RetroMagazine

semplicemente retro

Numero 19 - Anno 3 - Dicembre 2019 - WWW.RETROMAGAZINE.NET - Pubblicazione gratuita

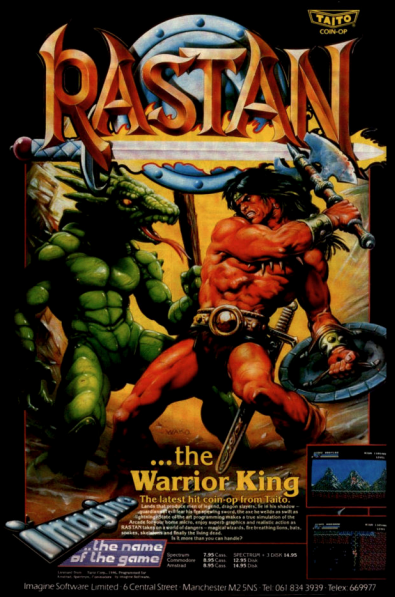
Auguri da tutta la redazione!

In questo numero:

MSX Games Report 2019

BASIC ed Interrupt sul Laser 500

The Chaos Engine



Rastan Saga

L'ineluttabile evolversi degli eventi...

Anche quest'anno siamo giunti all'ultimo numero dell'anno, pubblicato alla fine di dicembre e giusto in tempo per annoverare anche nel 2019 almeno 8 numeri di RetroMagazine.

Devo ammettere che riuscire a pubblicare 8 numeri all'anno è decisamente un impegno, a cui noi della redazione di RM però non riusciamo a rinunciare, spinti dalla passione e dalla voglia di realizzare un prodotto sempre migliore.

Si chiude un anno e come prassi è consuetudine fare un bilancio di tutto quello che è accaduto durante questo arco temporale. Non ho intenzione di menzionare numeri o snocciolare eventi, sarebbe troppo riduttivo e poco edificante. Quello che mi preme invece evidenziare è l'ineluttabile evolversi degli eventi.

Durante quest'anno diversi redattori si sono alternati all'interno della redazione, alcuni si sono allontanati, altri hanno deciso di unirsi a noi, altri ancora invece hanno preferito dare il loro contributo come redattori occasionali. La redazione cambia, cresce, si assottiglia per poi crescere nuovamente... E' l'ineluttabilie evolversi della vita che ci spinge a fare delle scelte, scelte che a volte sono necessarie per continuare ad evolversi.

E qual è l'ineluttabile evolversi della vita se non la morte? Purtroppo ogni anno che passa vediamo i protagonisti dell'epoca d'oro delle macchine a cui siamo legati passare a miglior vita. Ultimo, ma soltanto in ordine temporale è stato **Chuck Peddle** che ci ha lasciato il 15 dicembre. Non abbiamo avuto tempo di dedicargli un articolo completo, ma ci sembrava doveroso rendere omaggio, anche soltanto con una piccola menzione, a chi ha contribuito con il frutto del suo lavoro a rallegrare tante giornate della nostra giovinezza.

Concludo, come mia consuetudine, ringraziando tutti voi lettori per continuare a seguirci cosi' numerosi e per stimolarci a fare sempre meglio.

Buona lettura amici!

Francesco Fiorentini

SOMMARIO

Hanno collaborato alla stesura di questo numero di RetroMagazine

- Emanuele Bonin (RPI)
- Antonio Porcino
- Francesco Fiorentini
- Gianluca Girelli
- Alberto Apostolo
- Giuseppe Fedele
- Marco Pistorio
- David La Monaca (Cercamon)/Andrea Ferlito
- Daniele Brahimi
- Carlo Nithaiah del Mar Pirazzini
- Michele Ugolini
- Leonardo Vettori/Altri (FVB)
- Querino Ialongo
- **Copertina a cura di Flavio Soldani**

Dove dobbiamo andare per andare ... dove dobbiamo andare ?

Come scegliere il percorso migliore in poche mosse tramite l'algoritmo dei percorsi minimi di Dijkstra

di Emanuele Bonin (RPI)

Supponiamo di essere all'interno del gioco del gioco GhostBusters, stiamo guidando allegramente la nostra GhostMobile e all'improvviso abbiamo una chiamata di soccorso. Il nostro aiuto è richiesto per andare a disinfestare una casa all'altro capo della città, ma dobbiamo sbrigarci e quindi scegliere il percorso più veloce di tutti. In una cittadina come questa tutti gli isolati sono tutti dei quadrati, quindi la lunghezza di un qualsiasi percorso, che non vada nel senso opposto rispetto al punto dove dobbiamo arrivare, sarà sempre la stessa. Controllando l'immagine il concetto sarà più chiaro.

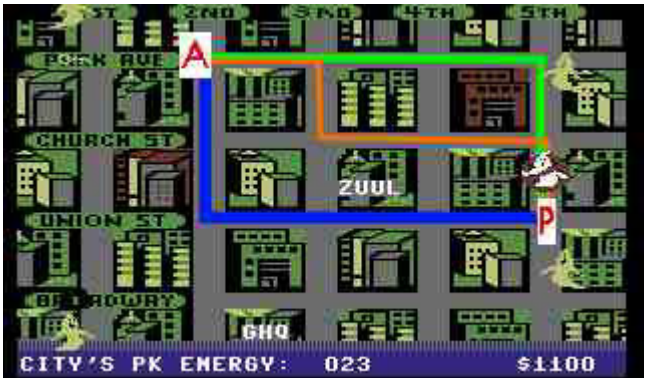


Figura 1 - Tutti i percorsi dalla Partenza all'Arrivo hanno la stessa lunghezza.

Supponiamo che alcune strade risultino accidentate o attraversate da ectoplasmi che ci rallenteranno la corsa, in questo caso, sebbene la lunghezza del percorso sia la stessa, alcuni percorsi risulteranno preferibili rispetto ad altri. Si potrebbe presentare il caso in cui le strade con il percorso più corto (nella figura le tre indicate hanno la lunghezza minima percorribile) siano talmente dissestate che ci conviene fare il giro più largo per risparmiare tempo.
Quindi, avendo a disposizione un computer con le indicazioni dei tempi di percorrenza delle strade cittadine, in quale maniera potremmo determinare quale sia il percorso più veloce in tempi rapidi senza dover provare tutti i percorsi possibili ?
Ci sono diversi algoritmi che ci vengono in aiuto per rispondere a questa domanda, uno dei più conosciuti e semplici è l'algoritmo inventato dall'olandese Dijkstra che ci permette di determinare il "miglior" percorso quando ci troviamo in casi simili a questo.
Lo scopo che mi propongo in questo articolo è appunto di cercare di illustrare questo semplice ma efficace algoritmo.

Studiamo la città

Come prima cosa, per applicare il nostro algoritmo, dobbiamo studiare l'ambiente in cui dobbiamo operare, nel nostro caso la cittadina dei Ghostbusters.
Facciamo corrispondere ad ogni incrocio che vediamo nella piantina una lettera distintiva e assumiamo di poter conoscere la condizione del tratto di strada che c'è tra due incroci.

Figura 2 - Ogni incrocio percorribile ora risulta distinto tramite una lettera. Gli incroci a sinistra non sono stati presi in considerazione solo per semplicità

Il passo successivo è quello di togliere tutto ciò che risulta superfluo ai fini dell'algoritmo, lasciando l'essenziale, cioè le strade di collegamento e le lettere che indicano gi incroci.

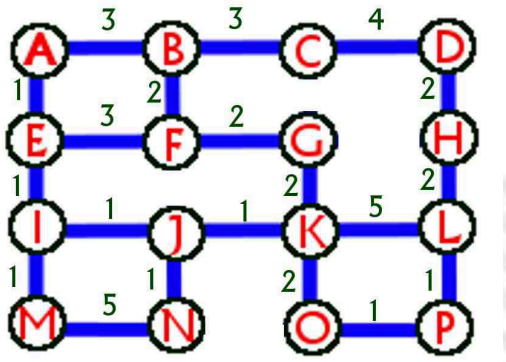


Figura 3 - La piantina "riscritta" all'interno di un grafo in cui le lettere corrispondono agli incroci (nodi) e le strade (collegamenti) sono state valutate secondo il loro tempo di percorrenza alcune strade sono state tolte per indicare che in realtà risultano non percorribili.

Le strade, come abbiamo già visto, non hanno tutte lo stesso tempo di percorrenza, ma noi, nel nostro esempio, abbiamo un metodo di valutazione che va da 1 (molto veloce) a 5 (molto lenta) e che utilizzeremo per dare i voti a tutte le strade percorribili, il risultato del nostro lavoro lo vediamo in figura. Inoltre per comprendere il caso di strade non percorribili, ne abbiamo tolta qualcuna.
Ciò che abbiamo ottenuto è un Grafo, dove i nodi corrispondono agli incroci e i collegamenti tra i nodi sono le strade.

Dopo aver trasportato tutto il nostro ambiente in un grafo, dobbiamo creare una tabellina con tutti i nodi e ad ogni riga verranno inseriti alcuni dati relativi al nodo stesso:

Nodo	Distanza	Nodi vicini da visitare	NodoP
L	0	H,K,P	
A	∞	B,E,F	
B	∞	A,C,E,F	
C	∞	B,D	
D	∞	C,H	
E	∞	A,B,F	
F	∞	E,B,G	
…	∞	…	

I significati delle colonne sono:

- Nodo: nome del nodo i cui dati sono nella riga della tabella;
- Distanza; indica la distanza del nodo dal nodo di partenza, inizialmente questo valore è "infinito" (verosimilmente si metterà un valore altissimo o un valore speciale [es. -1]) per tutte le righe tranne che per il nodo di partenza che ovviamente sarà 0;
- Nodi vicini da visitare: contiene la lista dei nodi raggiungibili;
- NodoP: conterrà il nodo dal quale raggiungere questo nodo alla fine dell'elaborazione.

Ora abbiamo tutto per iniziare, dobbiamo decidere solamente quale sarà il nodo di partenza e quale quello di arrivo, per questo esempio abbiamo il nodo L come partenza (da notare che è stato inserito come primo nodo appositamente nella tabellina ordinata per distanza) e A il nodo di arrivo.

Come lavora l'algoritmo

L'algoritmo si propone di determinare **la distanza minima di ogni nodo dal nodo partenza**, finché non viene raggiunto il nodo di arrivo. Naturalmente questo presuppone che il nodo di arrivo abbia almeno un percorso che gli permetta di essere raggiunto dal nodo di partenza, pena la conclusione dell'algoritmo senza una risposta valida, che poi è in realtà una risposta concreta.
In parole povere l'algoritmo selezionerà ad ogni ciclo di ricerca un nodo "corrente", il primo dei quali sarà il nodo di partenza. Una volta scelto il nodo corrente **vengono esaminati tutti i nodi a lui vicini raggiungibili**.
All'inizio tutti i nodi sono "**non visitati**", una volta che per un nodo sono stati analizzati tutti i suoi vicini, quel nodo viene marcato come nodo "**visitato**" e quel nodo non verrà più preso in considerazione in questa fase dell'analisi.
Il nodo corrente ha un valore della distanza dal nodo di partenza correttamente calcolato (diverso da infinito), quindi ogni suo vicino avrà **una distanza dalla partenza pari alla distanza del nodo corrente sommata alla distanza tra i due nodi**. Quindi vien da se che bisogna calcolare la distanza che ha ogni vicino calcolata rispetto alla distanza del nodo corrente.

Questo calcolo verrà assegnato al nodo vicino **solo se la distanza del nodo vicino è superiore alla distanza appena calcolata**. Infatti un nodo potrebbe essere stato preso in considerazione da altri suoi vicini che avranno calcolato una distanza a loro relativa.
Altra cosa molto importante, da fare nel caso in cui il nuovo calcolo risultasse inferiore al valore di distanza presente nel vicino, è quella di **registrare nella casella del nodo precedente (NodoP) il nodo corrente**.
Non appena è stato visitato completamente il nodo corrente, ammenoché non sia proprio il nodo di arrivo, dovremo scegliere il prossimo nodo da esaminare e per far ciò è sufficiente prendere come prossimo nodo il nodo (non visitato) che si trova in cima alla tabellina dei nodi con la distanza inferiore, quindi si ricomincerà ad analizzare i vicini.

Se invece il nodo corrente è proprio il nodo di arrivo allora abbiamo terminato la nostra ricerca, non ci resta che **ripercorrere a ritroso la catena dei nodi tramite i riferimenti ai nodi precedenti che ci siamo salvati durante l'elaborazione**.

L'utilizzo di un algoritmo del genere permette di velocizzare i tempi di ricerca.
Pensando di dover esaminare tutti i possibili percorsi, avremmo un numero di percorsi da provare che quadruplicano (nel nostro esempio) ad ogni nuovo nodo da esaminare, mentre in questa maniera ci ritroveremo al massimo a dover esaminare i nostri 16 nodi e i loro adiacenti, diminuendo di molto i tempi di elaborazione.

In un meta linguaggio potremmo scrivere l'algoritmo in questa maniera:

```
NodoCorrente = Nodo con Distanza = 0 (Nodo di partenza)
RicercaConclusa = No
Finchè RicercaConclusa == No {
        Finché ci sono nodi Adiacenti (non completamente visitati) {
                NodoAdiacente = ProssimoNodoAdiacente (non visitato)
                Somma=Distanza NodoCorrente + "distanza" fino a NodoAdiacente
                Se Somma < Distanza NodoAdiacente {
                        Distanza di NodoAdiacente = Somma
                        NodoP di NodoAdiacente = NodoCorrente
                        Togli dalla lista dei nodi da visitare del NodoCorrente
                        il NodoAdiacente
                }
        }
        Se NodoCorrente != NodoArrivo{
                NodoCorrente = Primo nodo della tabellina
        } Altrimenti {
                RicercaConclusa == No
        }
}
Se il NodoCorrente <> NodoDiArrivo{
        Ricerca non riuscita, i due nodi non sono collegati
} Altrimenti {
        Percorri a ritroso tutti i nodi a partire dal nodo corrente presente in tabella seguendo il campo NodoP
fino al raggiungimento del NodoDiPartenza
}
```

Programma in Basic V2 per C64

Quello che segue è un programma scritto in basic il cui scopo è di illustrare il funzionamento dell'algoritmo anche in maniera grafica.
Una volta avviato verrà chiesto il nodo di partenza e di arrivo, dopodiché partirà l'elaborazione secondo l'implementazione dell'algoritmo sopra descritto. Nel programma viene visualizzato il grafo con evinziatu i collegamenti tra i nodi e le "distanze assegnate" inoltre vengono evidenziati in rosso i nodi "attuali", cioè quelli per i quali si sta analizzando l'intorno, mentre in bianco i nodi dell'intorno esaminati. Viene anche mostrata la tabellina con i nodi da vistare (scritti in bianco) e già visitati (quelli scritti in nero). Al termine dell'elaborazione verrà mostrato il percorso "migliore" secondo le "distanze" assegnate.

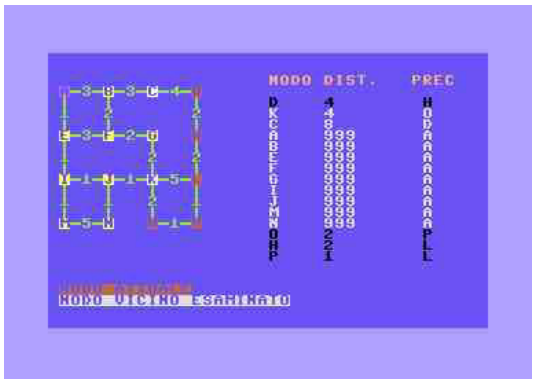


Figura 4 -Il programma durante la ricerca del percorso.

```
100 gosub 50000
105 gosub 50285
110 gosub 45000
115 x%=1:y%=21:gosub 49800: print
chr$(18);chr$(c%(2));"nodo attuale"
117 ? chr$(c%(15));" ";chr$(18);"nodo
vicino esaminato"
120 x%=1:y%=1:gosub 49800:rem locate
130 print chr$(c%(0));"nodo di partenza
a-p ?"
140 get a$: if a$<"a" or a$>"p" then goto
140
145 p%=asc(a$)-asc("a"):c%=4:n%=p%:gosub
45700:rem stp.nodo n% nel colore c
150 x%=1:y%=1:gosub 49800:rem locate
160 print "nodo di arrivo a-p ?  "
170 get a$: if a$<"a" or a$>"p" then goto
170
172 a%=asc(a$)-asc("a"):c%=4:n%=a%:gosub
45700:rem stp.nodo n% nel colore c%
175 x%=1:y%=1:gosub 49800:rem locate
180 print "                    "
185 n%(p%,0) = 0:i%(p%,1)=n%(p%,0): rem
marco la distanza per nodo part. a 0
190 n%=p%:gosub 44500:gosub 44800: rem
ordina n%-elemento e stampa tabella
200 np%=i%(0,0): rem nodo corrente
205 n%=np%:c%=2:gosub 45700: rem stampa
nodo n% nel colore c%
220 if n%(np%,1)=-1 then if np%=a% then
goto 330:rem ricerca conclusa
225 if n%(np%,1)=-1 then goto 300: rem
prossimo nodo
230 if n%(np%,4)<>-1 then nn%=n%(np%,4):n%
(np%,4)=-1:goto265
240 if n%(np%,3)<>-1 then nn%=n%(np%,3):n%
(np%,3)=-1:goto265
250 if n%(np%,2)<>-1 then nn%=n%(np%,2):n%
(np%,2)=-1:goto265
260 nn%=n%(np%,1):n%(np%,1)=-1
265 if n%(nn%, 1)<>-1 then
n%=nn%:c%=1:gosub 45700: rem evidenzia
nodo nn%
270 i%=nn%:gosub 44700: rem metto in n%
il puntatore a i%() del nodo i%=nn%
```

```
275 gosub 44200:rem d%=distanza tra np%
(nodo corrente) e nn% (adiacente)
280 d%=d%+n%(np%,0): if d% >= n%(nn%, 0)
then 290: rem se valore maggiore lascia
282 n%(nn%,0)=d%:n%(nn%,5)=np%: rem
cambio valore distanza aggiorno nodo prec
285 i%(n%,1)=d%:gosub 44500:gosub 44800:
rem ordina n%-elemento e stampa tabella
290 goto 220: rem visita il prossimo
vicino se c'e`
295 rem i nodi completamente visitati i
porto in fondo alla lista
298 rem in quanto non vanno piu` presi in
cosiderazione
300 i%=np%:gosub 44700:i%(n%,1)=10000:
rem n%(np%,0)=10000
310 gosub 44500:gosub 44800: rem ordina
n%-elemento e stampa tabella
320 goto 200: rem vai a prendere il
prossimo nodo in cima alla lista
325 rem uscita dal ciclo di ricerca,
disegna il percorso ottimale
330 gosub 5100
340 get a$:if a$="" goto 340
350 restore:goto 105
5000 end

5090 rem stampa percorso a ritroso
5100 ii% = 0:c%=4:np%=i%(ii%,0)
5110 nn%=n%(np%,5):gosub 44200: rem in s%
mette il n. di strada
5120 gosub 45500: rem evidenzia la strada
puntata da s% con il colore c%
5125 if nn% = p% goto 5150
5130 np%=nn%
5140 goto 5110
5150 return


44190 rem restituisce d% la distanza tra
np% (nodo corrente) e nn% (adiacente)
44195 rem in s% ci sara` il numero della
strada
44200 t%=np%*100+nn%:gosub 44250: if
d%<>-1 then return
44210 t%=nn%*100+np%:gosub 44250: if
d%<>-1 then return
44230 print "errore cfg:";chr$(np%+65);"
";chr$(nn%+65)
44240 end
44250 d%=-1
44260 for i = 0 to 19
44270 if d%(i,0)=t% then d% = d%(i,1):
s%=i: i = 20
44280 next
44290 return


44490 rem riposiziona il nodo n% su i% in
quanto modificato
44500 if n% > 0 then if i%(n%, 1) < i%
(n%-1, 1) then goto 44530: rem sposta su`
44510 if n% <15 then if i%(n%, 1) > i%(n%
+1, 1) then goto 44560: rem sposta giu`
44520 goto 44600
44525 rem Sposta su`
44530 t%=i%(n%, 0):i%(n%, 0)=i%(n%-1,0):i%
(n%-1,0)=t%
44540 t%=i%(n%, 1):i%(n%, 1)=i%(n%-1,1):i%
(n%-1,1)=t%
44550 n%=n%-1:goto 44500
44555 rem sposta giu`
44560 t%=i%(n%, 0):i%(n%, 0)=i%(n%+1,0):i%
(n%+1,0)=t%
44575 t%=i%(n%, 1):i%(n%, 1)=i%(n%+1,1):i%
(n%+1,1)=t%
44585 n%=n%+1:goto 44510
44600 return


44690 rem metti in n% la posizione su i%
() del nodo i%
44700 for i = 0 to 15
44710 if i%(i, 0) = i% then n%=i: i=16
44720 next
44730 return

44800 rem stampa tabella nodi
44820 print chr$(c%(10))
44822 x%=20:y%=2:gosub 49800:rem locate
44826 ? "nodo dist.   prec"
44830 for i = 0 to 15:pp%=i%(i,0)
44832 ? chr$(c%(15));
44834 if n%(pp%, 1)=-1 then ? chr$
(c%(0)): rem i nodi "visitati" stampati
neri
44835 x%=20:y%=i+4:gosub 49800:rem locate
44838 ? "                    "
44839 x%=20:y%=i+4:gosub 49800:rem locate
44840 ? chr$(asc("a") + pp%);"   ";n%
(pp%, 0);tab(34);chr$(n%(pp%, 5)+65)
44850 next
44860 return

44990 rem stampa del grafo
45000 print chr$(147)
45005 rem stampa dei nodi e assegnazione
delle coordinate a n%()
45007 for i = 0 to 3
45010 for j = 0 to 3
45012 n% = j*4 + i : c% = 7
45015 n%(n%, 6) = 121 + j*160 + (i*4):
rem assegno cella dello schermo
45020 gosub 45700: rem stampa nodo n% nel
colore c%
45030 next j
45040 next i
45050 rem stampa delle strade
45055 c%=5 : rem colore iniziale delle
strade
45060 for i = 0 to 19
45070 s%=i: gosub 45500: rem stampa
strada s%-esima
45080 next
45090 return

45490 rem stampa strada puntata da s% con
il colore c%
45495 rem se y% e` dispari, la strada e`
orizzontale
45498 rem altrimenti e` verticale
45500 x%=d%(s%,2):y%=d%(s%,3):d%=d%(s%,1)
45510 if y%/2 = int(y%/2) goto 45530:rem
strada verticale
45515 x%=(x%-1)*4+2:y%=0.5*y%+0.5:y%=(y%-
1)*4+3:gosub 49800:rem locate
45525 print chr$(c%(c%));chr$(96)+mid$
(str$(d%(s%,1)),2,1)+chr$(96)
45526 goto 45560
45527 rem stampa strada verticale
45530 y%=y%/2:n%=161 + (y%-1)*160 + (x%-
1)*4
45540 poke 1024 + n%, 66:poke 55296+n%,
c%
45545 poke 1024 + n% + 40, 48 + d%(s%,
1):poke 55296+n%+40, c%
45550 poke 1024 + n% + 80, 66:poke
55296+n%+80, c%
45560 return
```

```
45690 rem stampa nodo n% nel colore c%
45700 i%=n%(n%, 6)
45705 poke 1024 + i%, n%+129
45707 poke 55296 + i%, c%
45710 return

49010 rem vicini di ogni nodo quartetti
49020 data "b", "e", "", ""
49030 data "a", "c", "f", ""
49040 data "b", "d", "", ""
49050 data "c", "h", "", ""
49060 data "a", "i", "f", ""
49070 data "e", "b", "g", ""
49080 data "f", "k", "", ""
49090 data "d", "l", "", ""
49100 data "e", "j", "m", ""
49110 data "i", "k", "n", ""
49120 data "g", "j", "l", "o"
49130 data "h", "k", "p", ""
49140 data "i", "n", "", ""
49150 data "m", "j", "", ""
49160 data "k", "p", "", ""
49170 data "o", "l", "", ""

49790 rem posizionamento del cursore alle
coordinate x%, y%
49800 poke 780,0:poke 781,y%:poke
782,x%:sys 65520:return

49900 rem tabella delle distanze d%(19,3)
49910 rem Nodo1*100+Nodo2, distanza, x, y
49920 rem x e y sono solo indicative
rispetto alla griglia
50000 data "a", "b", 3, 1, 1
50010 data "a", "e", 1, 1, 2
50020 data "b", "f", 2, 2, 2
50030 data "b", "c", 3, 2, 1
50040 data "c", "d", 4, 3, 1
50045 data "d", "h", 2, 4, 2
50050 data "e", "f", 3, 1, 3
50055 data "e", "i", 1, 1, 4
50060 data "f", "g", 2, 2, 3
50070 data "g", "k", 2, 3, 4
50080 data "h", "l", 2, 4, 4
50090 data "i", "j", 1, 1, 5
50100 data "i", "m", 1, 1, 6
50110 data "j", "k", 1, 2, 5
50120 data "j", "n", 1, 2, 6
50130 data "k", "o", 2, 3, 6
50140 data "k", "l", 5, 3, 5
50150 data "l", "p", 1, 4, 6
50160 data "m", "n", 5, 1, 7
50170 data "o", "p", 1, 3, 7
50175 rem codici colori per chr$
50180 data
144,5,28,159,156,30,31,158,129,149,150,15
1,152,153,154,155

50200 rem Tabellina per memorizzare i
nodi
50205 rem matrice n%(16, 7)  0 based
50210 rem colonna
50215 rem indice   indica il cardinale
del nodo a=1 b=2 ecc...
50220 rem 0        indica la distanza dal
nodo di partenza
50225 rem 1        cardinali del primo
vicino (-1 = nodo gia' visitato)
50230 rem 2        cardinali del secondo
vicino
50235 rem 3        cardinali del terzo
vicino
50240 rem 4        cardinali del quarto
vicino
50245 rem 5        cardinale del nodo
precedente (per la ricerca del percorso)
50255 rem 6        posizione su schermo
(per la visualizzazione)
50265 rem la riga 0 serve per memorizzare
nella colonan 0 il cardinale del primo
50270 rem nodo in ordine di distanza
50272 rem c%() tabella colori i% indice
su n%
50275 rem inizializzazione
50280 dim n%(15,6), d%(19,3), c%(15),
i%(15,1)
50282 return

50285 for i = 0 to 15: read a$, b$, c$,
d$
50286 i%(i,1)=999:i%(i,0)=i
50288 n%(i,0) = 999: rem distanza
infinita
50290 n%(i,1) = asc(a$) - asc("a")
50295 n%(i,2) = asc(b$) - asc("a")
50300 if c$<> "" then n%(i,3) = asc(c$) -
asc("a"):goto50306
50305 if c$=  "" then n%(i,3) = -1
50306 if d$<> "" then n%(i,4) = asc(d$) -
asc("a"):goto50310
50307 if d$=  "" then n%(i,4) = -1
50310 next
50315 rem lettura delle distanze e le
posizioni delle strade
50320 for i = 0 to 19
50325 read a$, b$
50330 n1% = asc(a$) - asc("a"):n2% =
asc(b$) - asc("a")
50335 d%(i, 0) = n1%*100+n2%
50340 read d%(i, 1),d%(i, 2),d%(i, 3)
50345 next
50350 for i=0 to 15:read c%(i):next
50360 return
```

BASIC ed Interrupt sul Laser 500

di Antonino Porcino

La passione per il retrocomputing mi ha portato ad interessarmi del Laser 500 della Video Technology, una macchina praticamente sconosciuta che rischia di svanire nell'oblio.

Fortunatamente, di recente è stato recuperato anche il manuale operativo della macchina grazie al lavoro di Carlo Provetto che lo ha ritrovato e scansionato (il file pdf è scaricabile all'indirizzo riportato in calce all'articolo).

A differenza di molti altri manuali dei computer dell'epoca (mi riferisco ad esempio ai Commodore) quello del Laser 500 è estremamente dettagliato, contiene tutte le informazioni necessarie per poter sfruttare in pieno le potenzialità del computer senza aver bisogno di altre guide o manuali; c'è proprio tutto: dal BASIC all'hardware, compresi gli schemi elettrici per eventuali riparazioni. Insomma un manuale davvero ben fatto!

All'epoca in Italia circolava il manuale in lingua francese piuttosto che inglese, e ricordo che la ditta che distribuiva il Laser in Italia (la Scheidegger) ne commissionò una traduzione in italiano allo scopo di agevolare l'adozione del Laser 500 nei propri corsi di informatica. Questa però non fu mai portata a compimento (ed è facile capire perché, nella traduzione: "bit" era diventato "morso" !).

Il Laser 500 è un computer dall'architettura molto semplice: basato sullo Z80 dispone di 64K di RAM, 32 K ROM e un di chip video proprietario. Nella ROM ha un ottimo interprete BASIC, è siglato "Video Technology BASIC 3.0" ma in realtà non è che un altro classico Microsoft BASIC adattato alla macchina e concesso in licenza alla Video Technology. Addirittura alla fine del manuale si trova un form di "non-disclosure agreement" con il quale l'utente si doveva impegnare a custodire il Microsoft BASIC e a non farlo utilizzare da persone non autorizzate (!).

Sfogliando il suddetto manuale, a pagina 160 si trovano alcuni consigli su come rendere più efficienti i programmi. Fra questi ve n'è uno che mi ha insospettito:

7) Use simple expression in functions.
You may get the result from a function much faster if you simplify the argument of that function.

Example: `10 A = 3^2 + 17 – 22/3 : A = INT (A)`

will execute much faster than

Example: `10 A = INT (3^2 + 17 – 22/3)`

but occupy more space.

Secondo quanto consigliato, rendere più semplice l'argomento di una funzione ne aumenterebbe la velocità di esecuzione. Ossia

```
A=3^2+17-22/3:A=INT(A)
```

sarebbe più veloce di

```
A=INT(3^2+17-22/3)
```

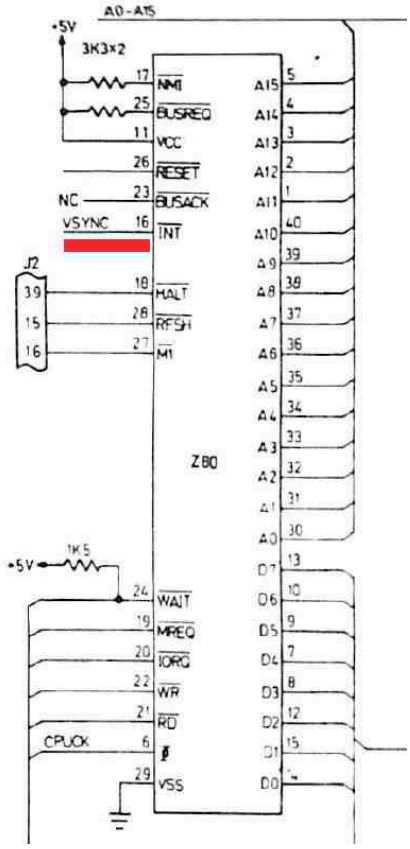
Ciò mi è sembrato piuttosto contro intuitivo, così ho voluto mettere alla prova quanto suggerito scrivendo un piccolo programma di test.
Dopotutto è sufficiente mettere le istruzioni incriminate a confronto dentro un ciclo FOR, allo scopo di rendere apprezzabile la differenza di tempo:

```
120 FOR I=1 TO 100
130 A=3^2+17-22/3:A=INT(A)
140 NEXT

190 FOR I=1 TO 100
200 A=INT(3^2+17-22/3)
210 NEXT
```

Adesso basterebbe semplicemente contare il tempo impiegato nei due FOR-NEXT. Vi è però un problema: il BASIC del Laser 500 non dispone di un meccanismo per misurare il trascorrere del tempo (l'equivalente della variabile "TI" o "TI$"). Possiamo però ricorrere ad uno stratagemma.

Come si evince dallo stesso manuale, nel Laser 500 il segnale VSYNC del chip video è collegato direttamente al pin /INT (interrupt) della CPU Z80 (vedi fig.). Questo fa sì che ad ogni inizio del ritracciamento della pagina video, cioè esattamente ogni 20 millisecondi, venga generato un interrupt sulla CPU; questo poi è utilizzato dal kernel in ROM per eseguire alcune funzioni di base, come ad esempio la scansione della tastiera, il lampeggiamento del cursore, eccetera.



Finita la routine di interrupt, il kernel dà la possibilità all'utente di eseguire eventualmente una sua routine in assembly. A tale scopo, completato l'interrupt, il kernel esegue un'istruzione jump in RAM alla locazione $8012, nella mappa di memoria indicata come il punto di uscita della routine di interrupt ("INTERUPT ROUTINE EXIT POINT"). In essa normalmente vi è contenuta una istruzione "RET"; modificandola in "JP indirizzo" diventa possibile eseguire del proprio codice ad ogni ciclo di interrupt.

SYSTEM VARIABLES

Address (HEX)	Description
8ØØØ	RST 8 jumps here
8ØØ3	RST 1Ø jumps here
8ØØ6	RST 18 jumps here
8ØØ9	RST 2Ø jumps here
8ØØC	RST 28 jumps here
8ØØF	RST 3Ø jumps here
8Ø12	INTERUPT ROUTINE EXIT POINT
8Ø15–8Ø1A	reserved
8Ø1B–8Ø2E	USR function addresses
8Ø2F	number of nulls
8Ø3Ø	store eaten char when not CTRL-C
8Ø31	save error number

Nel nostro caso, l'idea è quella di scrivere una piccola subroutine che incrementi un contatore in memoria, il quale diverrà un surrogato della variabile "TI" non presente nel BASIC del Laser.

In assembly Z80 sarà:

```
counter EQU $8650
timer:
  ld  hl,(counter)  ; carica il contatore in HL
  inc hl            ; incrementa HL di 1
  ld  (counter),hl  ; riscrive HL incrementato nel
contatore
  ret               ; esce dalla routine di
interrupt
```

dove $8650 è una word nella memoria bassa del Laser 500 normalmente non utilizzata. (La RAM inizia a $8000 e il BASIC a $8995).

Il nostro contatore è dunque a 16 bit, questo ci consente di contare fino a circa 21 minuti prima che raggiunga il massimo e riparta da zero (ossia 65536 x 20 ms), un tempo più che sufficiente per il nostro scopo di contare i ritardi nei cicli FOR.
Da BASIC potremo poi accedere al contatore semplicemente leggendolo con "PEEK":

```
T=PEEK(&H8650)+PEEK(&H8651)*256
```

Oltre alla routine di interrupt vera e propria che incrementa il contatore, ci serve anche la parte che installa tale routine modificando il vettore del kernel a $8012 sopra descritto. L'intero codice assembly diventa quindi:

```
org $d000

install:
  ; azzera il contatore
  ld hl, 0
  ld (counter), hl
  ; carica in HL l'indirizzo della routine "timer"
  ld hl, timer
```

```
  ; disabilita momentaneamente gli interrupt
  di
  ; scrive l'indirizzo della routine
  ld (0x8013), hl
  ; scrive "JP" (C3 è l'opcode per "JP")
  ld a, $c3
  ld (0x8012), a
  ; riabilita gli interrupt
  ei
  ret
counter EQU $8650
timer:
   ld  hl, (counter)
   inc hl
   ld  (counter), hl
   ret
```

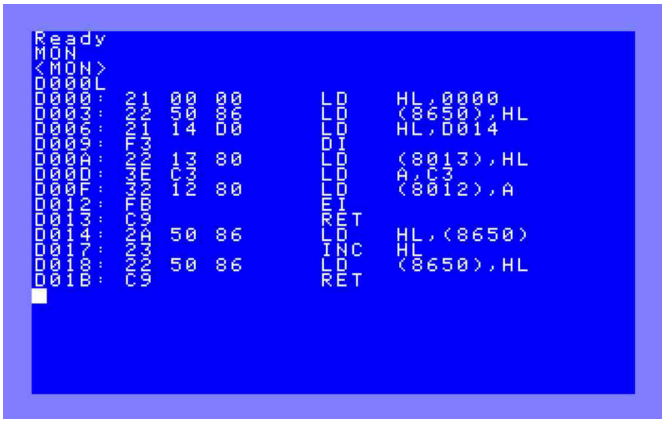
Il codice è posizionato all'indirizzo $D000, in una zona abbastanza alta nella RAM libera tale da non andare in conflitto con il programma BASIC che risiede più in basso a partire da $8995.
Per comodità scrivo il file sorgente sul PC e compilo con un normale assemblatore per Z80, ad esempio l'ottimo yaza:

```
$ yaza timer500.asm
Errors: 0 Warnings: 0

$ ls timer500.bin

28 timer500.bin
```

Alternativamente, potremmo digitare il codice direttamente sul Laser con il MONITOR integrato nel BASIC, il risultato sarebbe identico:



Per rendere tutto più omogeneo, inseriamo i byte del codice assembly nel listato BASIC con delle istruzioni DATA, proprio come si usava fare all'epoca:

```
20 FOR T=0 TO 27:READ B:POKE &HD000+T,B:NEXT
30 T=&HD000:CALL T
40 DATA
&H21,&H00,&H00,&H22,&H50,&H86,&H21,&H14,&HD0,&HF3
50 DATA
&H22,&H13,&H80,&H3E,&HC3,&H32,&H12,&H80,&HFB,&HC9
60 DATA &H2A,&H50,&H86,&H23,&H22,&H50,&H86,&HC9
```

In questo modo, all'avvio il programma BASIC installerà automaticamente il contatore timer in interrupt.

Possiamo adesso estendere il resto del programma BASIC leggendo dalla locazione del contatore che sfruttiamo a mo' di cronometro:

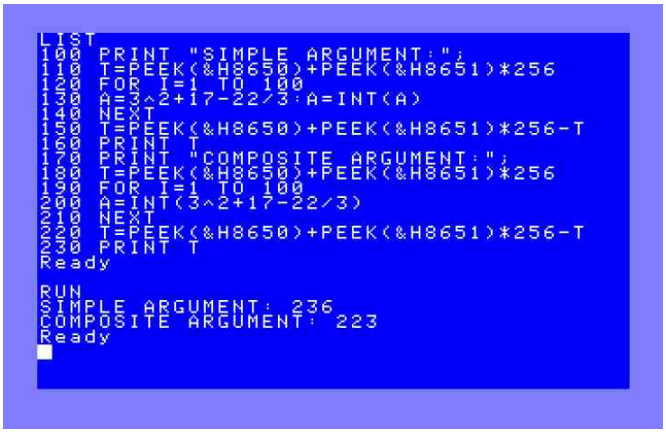
```
100 PRINT "SIMPLE ARGUMENT:";
110 T=PEEK(&H8650)+PEEK(&H8651)*256
120 FOR Z=1 TO 100
130 A=3^2+17-22/3:A=INT(A)
140 NEXT
150 PRINT PEEK(&H8650)+PEEK(&H8651)*256-T

170 PRINT "COMPOSITE ARGUMENT:";
180 T=PEEK(&H8650)+PEEK(&H8651)*256
190 FOR Z=1 TO 100
200 A=INT(3^2+17-22/3)
210 NEXT
220 PRINT PEEK(&H8650)+PEEK(&H8651)*256-T
```

Il programma di test adesso è pronto. Eseguiamo con RUN ed otteniamo il risultato a video:

```
SIMPLE ARGUMENT: 236
COMPOSITE ARGUMENT: 223
```

Troviamo così la conferma ai nostri sospetti: la versione "consigliata" dal manuale non è per nulla più veloce, anzi è leggermente più lenta: 236 ticks contro 223 (un tick corrisponde a 20 millisecondi).



Non ho verificato, ma credo che questo risultato sia estendibile anche agli altri interpreti BASIC (Microsoft e non); chi vuole può verificarlo sulla propria macchina vintage preferita, purché ovviamente questa disponga di un meccanismo per misurare il tempo.

Trovate i sorgenti e gli eseguibili di questo piccolo esperimento nel mio repository su GitHub nel link sotto riportato.

Links

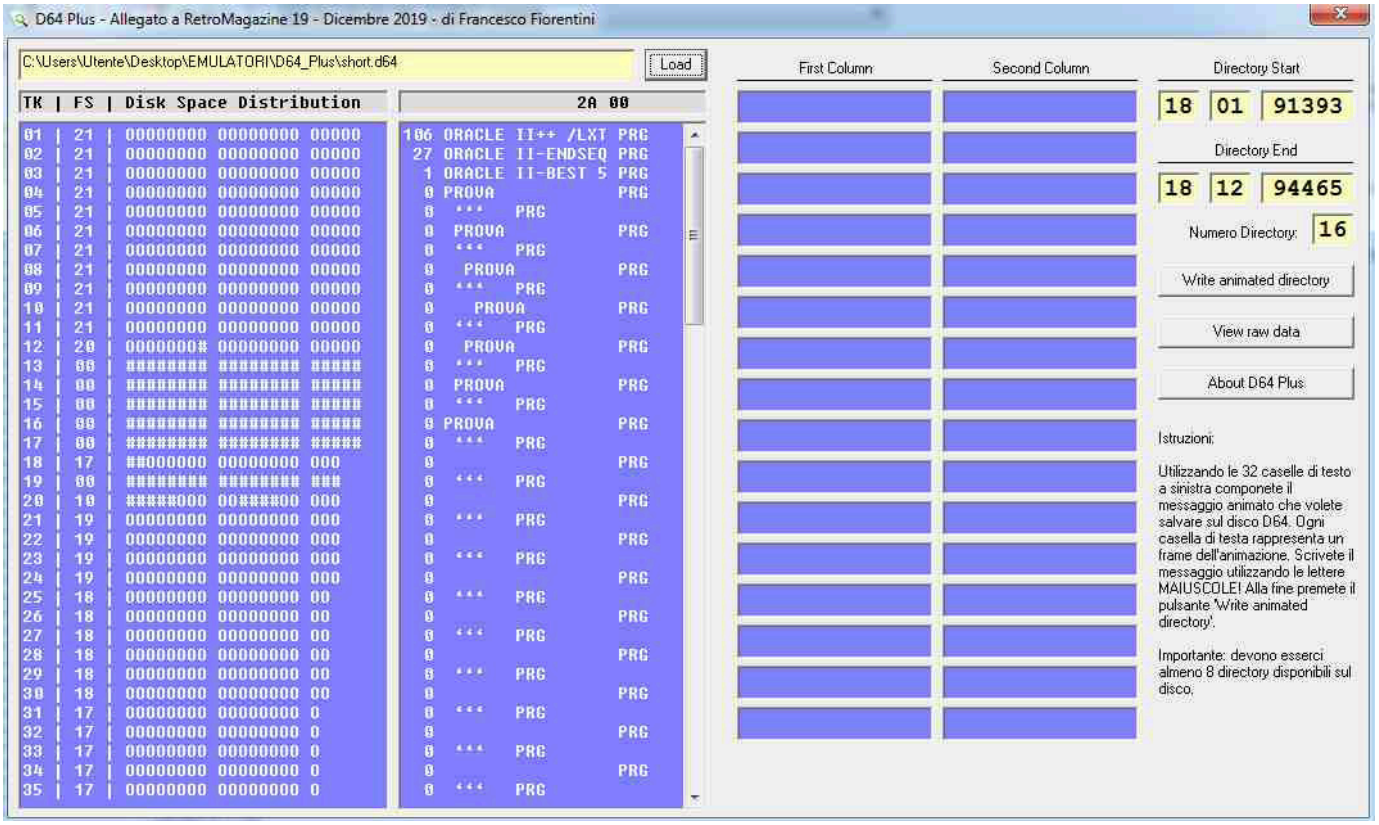
Manuale Laser 500: http://www.radioedintorni.it/Immagini/RetroComputer/RC-Manuali/Vtech-Laser-500.pdf

Repo su GitHub: https://github.com/nippur72/8-bit-projects/tree/master/laser500-simplify-argument

Emulatore online Laser 500: https://nippur72.github.io/laser500emu

Impara l'arte... E non metterla da parte!
Il formato D64 - quarta parte

di Francesco Fiorentini



Nel numero 14 abbiamo iniziato un viaggio affascinante che ci ha portato a scoprire alcuni dei segreti del formato D64 (la rappresentazione logica di un disco fisico del C64 su file). Abbiamo imparato a leggere il nome del disco, lo spazio disponibile e quello invece utilizzato, abbiamo imparato a seguire la catena delle directory, leggere i nomi dei file memorizzati e quanto spazio occupano sullo stesso. Abbiamo anche rilasciato un programmino in Visual Basic 5 per mettere in pratica quanto appreso.

Nel numero 17 abbiamo provato a manipolare il formato D64 per creare alcuni effetti per stupire amici e conoscenti.

Nel numero 18 Marco Pistorio ci ha fatto scoprire come hackerare il formato D64 per generare un effetto simpatico come le directory animate. Il trucchetto insegnatoci da Marco è efficace e facilmente riproducibile, ma decisamente noioso da mettere in pratica. Da qui l'idea di realizzare qualcosa per automatizzare il processo e rendere la creazione delle directory animate veloce ed immediata.
In questo numero ho quindi voluto combinare le nozioni apprese nei numeri 14 e 18 con il programma realizzato in Visual Basic 5 per rendere tutto ciò possibile.

Come abbiamo potuto apprendere dall'articolo di Marco, l'animazione viene realizzata tramite la sovrapposizione della scrittura del nome di un file fittizio contenuto nel disco D64.
In pratica l'animazione non è altro che un semplice nome di un file che viene riscritto velocemente sopra se stesso e crea così l'illusione di un'animazione.

Per raggiungere lo scopo di automatizzare questo processo, nel mio esempio in Visual Basic ho creato un array di 32 campi testo che l'utente può utilizzare per scrivere un messaggio animato a suo piacimento. Ovviamente, per creare l'illusione dell'animazione, il messaggio deve essere scritto simulando il movimento di una stringa all'interno dei 16 caratteri consentiti come lunghezza del nome del file su un disco in formato D64. Gli effetti che possono essere generati sono molteplici ed il limite è soltanto la fantasia dell'utilizzatore.

Per aiutarvi a comprendere meglio il funzionamento, potete immaginare i 16 caratteri del nome del file come il display di un mini proiettore led ed i 32 campi di testo come i frame di un immaginario messaggio da stampare sul proiettore.

Proviamo a fare un esempio molto semplice, cercando di creare l'animazione con il nome della nostra rivista. In questo caso simuleremo l'apparizione di una lettera alla volta, fino a comporre il nome RETROMAGAZINE e dopo aver spostato la parola dal lato sinistro fino all'estremità destra dello spazio disponibile, uno spostamento di soli 2 caratteri, ritorneremo sui nostri passi, cancellando le lettere, fino a lasciare visibile soltanto la lettera R...

```
01:'R               '
02:'RE              '
03:'RET             '
```

```
04:'RETR              '
05:'RETRO             '
06:'RETROM            '
07:'RETROMA           '
08:'RETROMAG          '
09:'RETROMAGA         '
11:'RETROMAGAZ        '
12:'RETROMAGAZI       '
13:'RETROMAGAZIN      '
14:'RETROMAGAZINE     '
15:' RETROMAGAZINE    '
16:'  RETROMAGAZINE   '
17:'   RETROMAGAZINE  '
18:'  RETROMAGAZINE   '
19:' RETROMAGAZINE    '
20:'RETROMAGAZINE     '
21:'RETROMAGAZIN      '
22:'RETROMAGAZI       '
23:'RETROMAGAZ        '
24:'RETROMAGA         '
25:'RETROMAG          '
26:'RETROMA           '
27:'RETROM            '
28:'RETRO             '
29:'RETR              '
30:'RET               '
31:'RE                '
32:'R                 '
```

Come potete facilmente intuire questo è soltanto uno dei possibili effetti che possono essere creati con la nostra piccola applicazione, ma piuttosto che dilungarmi sul suo utilizzo, vorrei spendere due parole per spiegare come è stato realizzato il programmino in Visual Basic.

Avendo già a disposizione il programma realizzato per l'articolo del numero 14, buona parte del codice necessario era già pronto allo scopo. Non dovevo fare altro che riprendere quel codice ed aggiungere una routine per simulare la scrittura di alcuni file su un disco D64.

Ovviamente la scrittura di un file D64 è più complessa della sua semplice lettura. Vi ricordo che i file sono scritti in blocchi di 8 e che le directory non sono consecutive, ma seguono generalmente un percorso ben precio per ottimizzare le prestazione del disco (per ulteriori informazioni vi rimando a leggere l'articolo sul numero 14), e che l'ultima directory deve contenere una sorta di terminatore per indicare che quella è l'ultima directory utile del disco.

Vi ricordo inoltre che l'animazione viene generata da due file, uno contenente il messaggio vero e proprio ed un file speciale che serve a riposizionare il cursore all'inizio della riga precedente; per scrivere tutti e 32 i frame dell'animazione ho quindi bisogno di ben 64 file che divisi per 8 necessitano di ben 8 directory libere sul disco...

Oltre all'array dei campi testo, ho popolato quindi due altri array: uno contenente la numerazione standard delle directory ed uno contenente i 16 caratteri da scrivere come caratteri per il file 'speciale'.

A questo punto non ho dovuto far altro che leggere l'ultima posizione dell'ultimo file, creare la directory successiva ed il suo relativo puntamento e scrivere le 8 directory successive contenenti ognuna 4 frame dell'animazione (4 campi testi) e 4 file speciali. facile vero?

Rappresentazione dei frame di animazione su disco

Effettivamente non è così difficile ed è più complesso da spiegare che da realizzare, una volta capita la logica del file D64. :-)

A questo punto avete tutte le informazioni in mano per comprendere la logica del programma VB che ho realizzato e che può essere modificato ancora ed ancora per fare ben altro... Chi ha detto estrarre i files e scriverne di nuovi?

Il programma VB, eseguibile con relativo sorgente, può essere scaricato dal seguente link: https://drive.google.com/file/d/178PnsdUUBftOK0PynyiDVVca7koH9Ijy/view?usp=sharing

Vi ricordo che lo scopo del programma è puramente didattico, il codice è volutamente discorsivo e non ottimizzato per ricercare la prestazione; il mio intento è dimostrare come con gli strumenti moderni sia possibile creare facilmente un ponte con le nostre retromacchine.

A questo punto non mi resta che augurarvi buone feste, anche se in ritardo, e buon anno nuovo!

Introduzione ad HOLLYWOOD - 3a parte

di Gianluca Girelli

1. Introduzione

Nel tutorial precedente abbiamo visto come costruire con semplicità un fondale e come animarlo usando la tecnica degli "Sprite".
L'obiettivo di queste pagine, invece, sarà quello di dimostrare come, con altrettanta semplicità, sia possibile trasformare il nostro "screensaver" animato in un visualizzatore di immagini (slideshow).

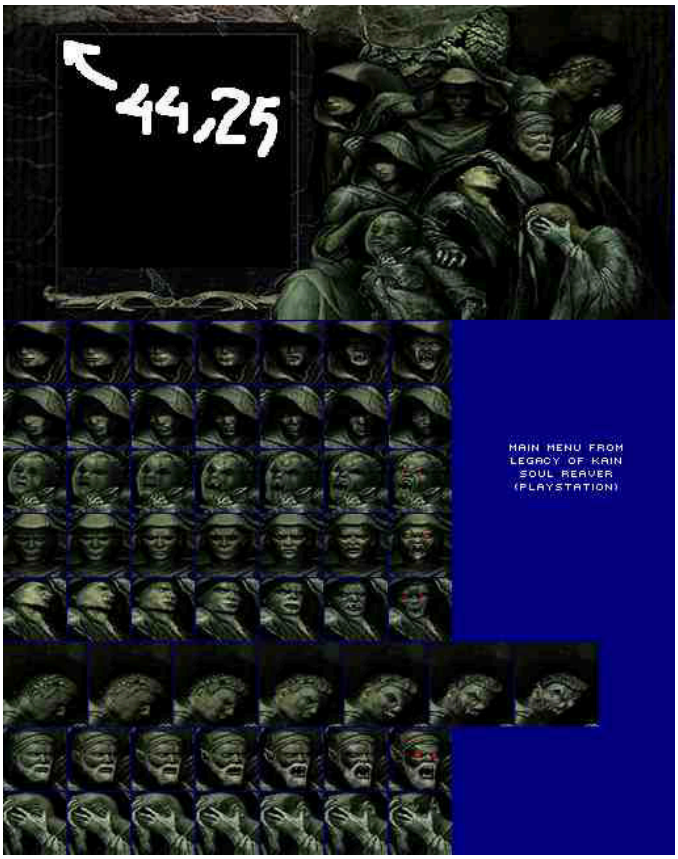


Figura 1

Poichè lo scopo di questi articoli è quello di dimostrare quanto Hollywood sia potente e flessibile, e non quello di scrivere il codice del miglior visualizzatore esistente, si faranno le seguenti assunzioni:
- il software agisce dall'interno di una directory di immagini. In tal modo non dovremo per il momento preoccuparci di imparare a gestire i "requester" per il cambio di percorso;
- si supporrà di avere un numero di foto (noto a priori e non minore di 1) in formato PNG chiamate "x.png" (con "X" che va da 1 al massimo numero di foto presenti nella directory). Non dovremo quindi per il momento preoccuparci di gestire problemi di validazione dell'input del programma (scansione dell'intera directory alla ricerca di tutte le possibili immagini, gestione dei diversi formati, verifica che esista almeno un'immagine etc....).

Inoltre, per meglio evidenziare le nuove sezioni aggiunte al codice dello screensaver/viewer che, per la maggior parte, è quello pubblicato sul numero scorso, non si è provveduto ad effettuare un'ottimizzazione dell'algoritmo che lo avrebbe sicuramente reso più compatto ed efficiente, a scapito però della facilità di lettura.

2. Analisi del codice

Come si può vedere, la parte iniziale del programma è virtualmente identica alla versione precedente. Nessuna modifica è stata effettuata nelle sezioni riservate ai "pre-processor commands" @DISPLAY e @SCREEN. Per quanto riguarda la procedura di inizializzazione variabili è stato solo aggiunto un contatore ("cnt") che verrà utilizzato come puntatore alla successiva immagine da visualizzare.

Analizzando la procedura "p_NextPic()", si evince come l'immagine corrente sia sempre caricata nel "Brush" numero 11 grazie all'istruzione "LoadBrush(11, cnt .. ".png")" e visualizzata con un effetto di transizione scelto randomicamente grazie all'istruzione "DisplayBrushFX(11, 44, 25, #RANDOMEFFECT)". "44" e "25" sono le coordinate dell'angolo in altro a sinistra della nostra "cornice", così come evidenziato in figura 1; i due punti ("..") riportati nell'istruzione LoadBrush, rappresentano invece l'operatore di concatenazione tra la stringa ".png" e quella contenuta nella variabile "cnt".

Poichè le foto contenute nella nostra directory possono però essere di diverse dimensioni non note a priori, è necessario riscalarle per adeguarle alla grandezza della nostra area di visualizzazione, costituita da un quadrato di 174x174 pixel.
Di questa operazione si occupa la nostra procedura "p_GetRatio(brush_num)" che, grazie all'istruzione "GetAttribute()", recupererà altezza e larghezza della foto. Successivamente, dopo aver determinato quale dei due lati è maggiore, si provvederà al "downscaling" mantenendo però inalterato il rapporto tra le dimensioni (istruzione "ScaleBrush(brush_num, #KEEPASPRAT, height)" o "ScaleBrush(brush_num, width, #KEEPASPRAT)"). E' da notare come non abbia nessuna importanza che poi il tutto venga effettivamente visualizzato in una finestra di 174x174 pixels reali o "virtuali", in quanto Hollywood si occuperà di fare tutte le conversioni del caso (anche "on-the-fly") grazie al motore di autoscaling interno ("ScaleMode=#SCALEMODE_AUTO")!

Riepilogando, il nostro "Main Loop" funziona ora in

questo modo (in pseudo codice):

```
inizio loop ; continua finchè l'utente non decide di uscire
  temporizza le animazioni degli sprite
      carica nuova foto
      calcola dimensioni foto ed effettua downscaling
      temporizza le transizioni delle foto
      visualizza la foto nel riquadro
      anima lo sprite "di turno"
      seleziona un nuovo sprite
  resetta i timer
  rinfresca il "background"
fine loop
```

Il risultato di tutto questo è ben visibile in figura 2, che mostra il nostro programma in azione.

A questo punto, sarebbe possible sbizzarrirsi per migliorare il codice arricchendolo di tante altre caratteristiche quali:
- la possibilità di inserire il percorso delle directory dalle quali caricare le immagini;
- il controllo dell'esistenza effettiva delle immagini e del loro formato (PNG, JPG, BMP, IFF, ILBM ..);
- una migliore temporizzazione degli sprite (movimento delle statue) e dell'alternanza delle foto;
- possibilità di selezionare la traccia sonora (o la playlist) da ascoltare (MP3, WAV, MOD ....);
- etc....

Lascio l'implementazione di queste caratteristiche al lettore volenteroso, ma sono comunque contattabile per chiarimenti attraverso i canali ufficiali della rivista.

3. Conclusioni

Come è ormai evidente, Hollywood costituisce veramente un ambiente MAL (Multimedia Application Layer) in grado di facilitare enormemente il compito del programmatore, sia esso occasionale o professionista. Con Hollywood non ci si deve preoccupare, oltre a scrivere il codice, anche di implementare tutti i plug-in necessari per l'uso delle diverse risorse in gioco, come invece solitamente accade con gli altri linguaggi di programmazione.
Inoltre, a partire dalla versione 5.2 (codename "Infinity"), Hollywood implementa anche l'uso del 3D grazie a librerie dedicate di cui parleremo in seguito. Attualmente il framework ha raggiunto la versione 8.0.

Arrivederci allora sul prossimo numero per esplorare qualche nuova caratteristica del linguaggio.

DISCLAIMER!

Figura 2

```
/*************************************************************
**                                                          **
** Name:        Gothic Screensaver and viewer               **
** Author:      Gianluca Girelli                            **
** Version:     1.0                                         **
** Date:        06.05.2013                                  **
** Interpreter: Hollywood 4.5                               **
** Licence:     Hollywood screensaver and viewer            **
** Function:    A screensaver-like demo code with gothic    **
**              background. The images are taken from PS1   **
**              game "Legacy of Kain: Soul Reaver".         **
**              Pictures viewer feature added to original   **
**              code (released on 17.07.2011).              **
**                                                          **
** History:                                                 **
**                                                          **
** 1.0: (06.05.13)                                          **
** - initial release                                        **
**                                                          **
*************************************************************/

@APPTITLE       "Gothic Screensaver and viewer"
@APPAUTHOR      "Gianluca Girelli"
@APPCOPYRIGHT   "Freeware - Graphics copyrights remain of original authors."
@APPVERSION     "$VER: 1.0 (06.05.2013)"
@APPDESCRIPTION "A picture viewer and screensaver-like demo done with Hollywood."

@VERSION 4,5
@DISPLAY 1, { Title = "'Gothic Screensaver' - Made on Amiga with Hollywood by gOblin",
              X=#CENTER, Y=#CENTER, Width=512, Height=240, HidePointer=True,
              Sizeable=True, ScaleMode=#SCALEMODE_AUTO }
@SCREEN {Mode = "Ask", Width = 640, Height = 480}


Function p_InitVars()
        ;from brush 1 to brush 8 we have facial animations ...
        @BRUSH 1, "mainmenu.png", {X=0, Y=241, Width=343, Height=48, Transparency = $000080 }
        @BRUSH 2, "mainmenu.png", {X=0, Y=289, Width=343, Height=48, Transparency = $000080 }
        @BRUSH 3, "mainmenu.png", {X=0, Y=338, Width=343, Height=48, Transparency = $000080 }
        @BRUSH 4, "mainmenu.png", {X=0, Y=387, Width=343, Height=48, Transparency = $000080 }
        @BRUSH 5, "mainmenu.png", {X=0, Y=437, Width=343, Height=48, Transparency = $000080 }
        @BRUSH 6, "mainmenu.png", {X=0, Y=486, Width=455, Height=64, Transparency = $000080 }
        @BRUSH 7, "mainmenu.png", {X=0, Y=551, Width=343, Height=48, Transparency = $000080 }
        @BRUSH 8, "mainmenu.png", {X=0, Y=600, Width=343, Height=48, Transparency = $000080 }
        ;... while brush 10 is background image
        @BRUSH 10, "mainmenu.png", {X=0, Y=0, Width=512, Height=240, Transparency = $000080 }

        ;create all the sprites extracting the animation frames from "mainmenu.png"
        CreateSprite(1, #BRUSH, 1, 49, 48, 7, 7)
        CreateSprite(2, #BRUSH, 2, 49, 48, 7, 7)
        CreateSprite(3, #BRUSH, 3, 49, 48, 7, 7)
        CreateSprite(4, #BRUSH, 4, 49, 48, 7, 7)
        CreateSprite(5, #BRUSH, 5, 49, 48, 7, 7)
        CreateSprite(6, #BRUSH, 6, 65, 64, 7, 7)
        CreateSprite(7, #BRUSH, 7, 49, 48, 7, 7)
        CreateSprite(8, #BRUSH, 8, 49, 48, 7, 7)

        ; starts with animation frame 1
        f = 1
        ; sprites position coordinates over brush 10
        X = {235, 265, 273, 325, 340, 409, 382, 399}
        Y = { 48,  96, 156,  49, 120,  14,  78, 150}

        ; starts with first photo inside directory
        cnt=0
EndFunction

Function p_GetRatio(brush_num)
        height=GetAttribute(#BRUSH,brush_num, ATTRHEIGHT)
        width=GetAttribute(#BRUSH,brush_num, ATTRHWIDTH)
        If height > width
                height =174
                ScaleBrush(brush_num, #KEEPASPRAT, height)
        Else
                width = 174
```

```
                ScaleBrush(brush_num, width, #KEEPASPRAT)
        EndIf
EndFunction

Function p_NextPic()
                ; switch photo every 1000ms
                If GetTimer(2) > 1000
                If cnt > 8
                        cnt=1
                        LoadBrush(11, cnt .. ".png")  ; load next picture
                        p_GetRatio(11)                ; calculate aspect ratio and perform picture
scaling
                        DisplayBrushFX(11, 44, 25, #RANDOMEFFECT)      ; display picture
                Else
                        cnt=cnt+1    ; starting from photo n.1
                EndIf
                StartTimer(2)
                LoadBrush(11, cnt .. ".png")
                p_GetRatio(11)
                DisplayBrushFX(11, 44, 25, #RANDOMEFFECT)
                EndIf
EndFunction

Function p_MainLoop()
    ; switch frames every 150ms
    If GetTimer(1) > 150
      If f > 6
           p_NextPic()
           f =1
           Wait(50)
           DisplaySprite(random, X[random-1], Y[random-1], f)
           ; the following loop avoids animating the same sprite twice in a row ...
           random1=Rnd(8)+1
           While random=random1 Do random1=Rnd(8)+1
           random=random1
           ; ... then it wait at least 3 seconds before changing sprite
           Wait(Rnd(150)+150)
       Else
           f = f + 1
       EndIf
      StartTimer(1)
      ; display the background ...
      DisplayBrush(10, 0, 0)
      ; animate !
      DisplaySprite(random, X[random-1], Y[random-1], f)
    EndIf
EndFunction

;=========== Begin ===========
EscapeQuit(True)   ; enable ESCape

p_InitVars()

SetInterval(1, p_MainLoop, 1000/25) ; 25 fps

StartTimer(1)
StartTimer(2)
random=Rnd(8)+1

; endless loop follows
Repeat
    WaitEvent()
Forever
;============ End ==============
```

Routine vintage per calcolare erf(x) e la sua inversa

di Alberto Apostolo

In Statistica e in Fisica ricorre spesso la **funzione errore**

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

e la sua inversa erfinv (in Figura 1 si trovano i rispettivi grafici).
Inoltre si dice **complementare della funzione errore**

$$\mathrm{erfc}(x) = 1 - \mathrm{erf}(x).$$

Alla funzione erf(x) è associata la **distribuzione standard normale cumulata di Gauss**:

$$\Phi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x e^{-\frac{t^2}{2}} dt.$$

Si dimostra che tale funzione è legata a erf(x) tramite le relazioni:

$$\Phi(x) = \frac{1}{2} + \frac{1}{2} \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)$$

$$\Phi^{-1}(y) = \sqrt{2}\,\mathrm{erfinv}(2y-1).$$

E' necessario avvisare che in alcuni testi è la funzione phi(x) ad essere chiamata funzione errore. Pertanto si invitano i lettori a verificare sempre la definizione, al fine di non incorrere in spiacevoli malintesi.
Le funzioni erf, phi (con le loro inverse) sono usate anche in ambito bancario e finanziario.
Esse appartengono al bagaglio matematico dell'**Ufficio Rischio di Credito** di un qualsiasi istituto bancario, per assolvere il compito di calcolare i parametri riguardanti la clientela, tra cui il **Rating** (ben noto nelle cronache di qualche anno fa). A causa della funzione integranda exp(-t**2) è possibile calcolare erf ed erfinv esclusivamente con metodi approssimati, tra i quali:
1) metodi per il calcolo degli integrali (per es. il metodo di Cavalieri-Simpson),
2) metodi basati sull'uso di polinomi,
3) metodi per calcolare erfinv risolvendo l'equazione

$$f(x) = \mathrm{erf}(x) - y = 0$$

(spesso con il metodo di Newton-Raphson o delle tangenti, mentre si sconsiglia il metodo di bisezione a causa dell'andamento del grafico di erf(x)).
In alcuni software e in alcuni ambienti di programmazione, lo sforzo di mettere a disposizione erf, phi e le loro inverse come funzioni di libreria è già stato compiuto.
Gli utenti del linguaggio **R** dispongono delle funzioni **pnorm** e **qnorm** che rappresentano rispettivamente phi e la sua inversa, mentre in **MATLAB** si hanno erf(x), erfc(x), erfinv(x), erfcinv(x) (in Figura 2 si trova una tabella di traduzione ricavata da [Hie15]).
In **Excel** sono disponibili le funzioni **ERF** ed **ERFC** mentre le funzioni **NORM.DIST** e **NORM.INV** calcolano rispettivamente phi e la sua inversa (in Excel 2007, le funzioni si chiamano NORMDIST e NORMINV).
Invece il linguaggio **Python** include erf(x) ed erfc(x) tra le funzioni di libreria ma non erfinv(x).
Negli ambienti dove mancano tali funzioni, si possono implementare parti di codice che eseguono il calcolo approssimato dei valori oppure accedere a un foglio Excel sul quale essi sono stati tabulati opportunamente. L'uso di un foglio Excel permette anche di importare questi valori sulla tabella di un DataBase e di leggerli con query scritte in linguaggio **SQL**.

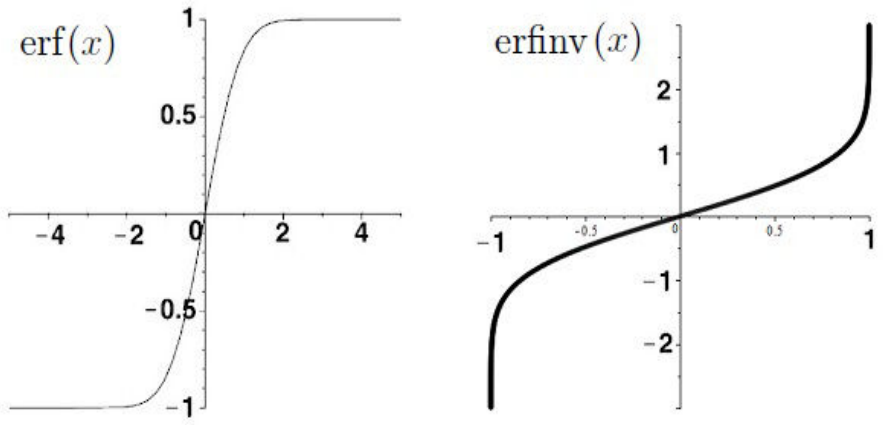


Figura 1

R	MATLAB
2*pnorm(x*sqrt(2))-1	erf(x)
2*pnorm(x*sqrt(2),lower=FALSE)	erfc(x)
qnorm((1+x)/2)/sqrt(2)	erfinv(x)
qnorm(x/2,lower=FALSE)/sqrt(2)	erfcinv(x)

Figura 2

Applicazione del metodo di Cavalieri-Simpson per calcolare erf(x)

Il metodo di Cavalieri-Simpson effettua l'approssimazione come riportato in Figura 3 .

Il programma **erf_simpson.py** (in Python 3.6) implementa la routine Fortran [Reg93] in Figura 4 e offre una stima grossolana dell'errore assoluto confrontandola con la funzione di libreria erf(x).

$$\int_a^b f(x)dx \approx \frac{b-a}{n}\left(y_0 + 4y_1 + 2y_2 + 4y_3 + \cdots + 2y_{n-2} + 4y_{n-1} + y_n\right),$$

dove $y_i = f(x_i)$ e $x_i = a + i\frac{b-a}{n}$.

Figura 3

```
# erf_simpson.py
"""Calcolo erf(x) con il metodo di Cavalieri-Simpson"""
# x argomento della funzione
# k fattore di scala per calcolare il numero di intervalli
# float in Python equivale per default a DOUBLE del Fortran
#
import math
#
def approx_erf(x):
    k = 0.2
    n = max(int(abs(x)/k)*2+1, 5)
    h = abs(x) / float(n-1)
    erf = 0.0
    test = 1
    for i in range(1,n+1):  #emulazione DO ... I = 1,N
        u = float(i-1)*h
        a = 2.0
        test = -test
        if test == 1:
            a = 4.0
        if (i == 1) or (i == n):
            a = 1.0
        erf = erf + a * math.exp(-u*u)
    erf = erf * abs(x) / (3.0 * float(n-1))
    erf = 2.0 * erf / math.sqrt(math.pi)
    if x < 0.0:
        erf = -erf
    return erf
#
if __name__=='__main__':
    n = 100000 + 1
    x0 = -10.0
    max_err_assoluto = -99999.
    h  = 2.0*abs(x0)/float(n-1)
    for i in range(n): #emulazione DO ... I = 0,N-1
        x = x0+h*float(i)
        y = math.erf(x)
        d = abs(round(approx_erf(x),6) - round(y,6))
        max_err_assoluto = max(max_err_assoluto,d)
    print("max err. assoluto: %.6f" % max_err_assoluto)
```

Lanciando il programma si ottiene nell'intervallo [-10,10]

```
max err. assoluto: 0.000013
```

```
C     ************************************************
      REAL FUNCTION ERF (XX)
C     NAME:        FUNCTION ERF
C     PURPOSE:     TO DETERMINE VALUE FOR ERROR FUNCTION
C                  (ERF)
C     INPUT
C     PARAMETERS:  NAME      DESCRIPTION
C                  X         COORDINATE FOR ERROR FUNCTION
C     OUTPUT
C     PARAMETERS:  NAME      DESCRIPTION
C                  ERF       FUNCTION ERF
C     COMMENTS:    ERROR FUNCTION IS DEFINED AS FOLLOWS:
C                      ERF = 2.0 * F / SQRT (PI)
C                  WHERE F IS THE INTEGRAL FROM 0 TO X OF
C                  EXP (- U * U)
C     DEFINE DISTRIBUTION FUNCTION
      F(U) = EXP (- U * U)
      X = ABS (XX)
      PI = 3.14159
      N = INT (X / 0.2) * 2 + 1
      IF (N .LT. 5) N = 5
      DELX = X / FLOAT (N - 1)
      ERF = 0,0
      DO 10 I = 1, N
         XF = FLOAT (I - 1) * DELX
         A = 2.0
         ITEST = (-1) ** I
         IF (ITEST .EQ. 1) A = 4.0
         IF (I .EQ. 1 .OR. I .EQ. N) A = 1.0
         ERF = ERF + A * F (XF)
   10 CONTINUE
         ERF = ERF * X / (3.0 * FLOAT (N - 1))
         ERF = 2.0 * ERF / SQRT (PI)
         IF (XX .LT. 0.0) ERF = - ERF
         RETURN
         END
```

Figura 4

Approssimazione con i polinomi

La funzione erf(x) ammette lo sviluppo in serie di Taylor:

$$\operatorname{erf}(x) = \frac{2}{\sqrt{\pi}} \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)n!}.$$

Tale serie converge per ogni x ma converge lentamente quando x è assai maggiore di 1.

Allora è utile una approssimazione asintotica per erfc(x) e, tramite la relazione erf(x) = 1 - erfc(x), calcolare la serie in Figura 5.

Per la funzione erfinv(x) è di interesse lo sviluppo in serie in Figura 6 dovuto a Carlitz (1963) e Strecok (1968) (cfr. [Mie14]).

Un'altra strategia è quella di ricorrere ad approssimazioni basate sui polinomi di Chebyshev.

La routine Fortran in Figura 7 realizza una approssimazione con tali polinomi dovuta ad Hastings [SS97] , ripresa dal programma **erf_hastings.py** con qualche modifica.

Per la sua semplicità, una utile approssimazione è anche quella in Figura 8 [Ale92],caratterizzata da un errore assoluto massimo di circa 0.0082 in [-10; +10].

Altre approssimazioni dovute ad Hastings si trovano sul celeberrimo formulario di Abramowitz e Stegun ("*Handbook of Mathematical Functions: With Formulas, Graphs, and Mathematical Tables*").
I lettori coraggiosi potranno cimentarsi con schemi di approssimazione più complessi se si consulterà il testo di Glassermann [Gla13] (cfr. schemi Beasley-Springer-Moro, Marsaglia, Zaman-Marsaglia).
Infine su Internet si possono scaricare gli articoli (in formato PDF) di Mike Giles ("*Approximating the erfinv function*") e di Sylvain Chevillard ("*The functions erf and erfc computed with arbitrary precision and explicit error bounds*").

$$\operatorname{erf}(x) = 1 - \frac{e^{x^2}}{\sqrt{\pi}\,x}\left(1 - \frac{1}{2x^2} + \frac{1\cdot 3}{2^2x^4} - \frac{1\cdot 3\cdot 5}{2^3x^6} + \cdots + (-1)^n \frac{(2n-1)!!}{2^n x^{2n}}\right)$$

Figura 5

$$\operatorname{erfinv}\left(\frac{2}{\sqrt{\pi}}z\right) = \left(z + \frac{z^3}{3} + \frac{7}{30}z^5 + \frac{127}{630}z^7 + \frac{4369}{22680}z^9 + \right.$$
$$\left. + \frac{34807}{178200}z^{11} + \frac{20036983}{97297200}z^{13} + O(z^{15})\right)$$

Figura 6

```
#erf_hastings.py
"""Calcolo erf(x) con la formula di Hastings"""
# x argomento della funzione
# float in Python equivale per default a DOUBLE del Fortran
import math

def approx2_erf(x):
    t = 1.0 / ( 1.0 + 0.3275911 * abs(x))
    a1 =  0.254829592
    a2 = -0.284496736
    a3 =  1.421413741
    a4 = -1.453152027
    a5 =  1.061405429
    erf = 1.0 - ((((a5*t+a4)*t+a3)*t+a2)*t+a1)*t*math.exp(-x*x)
    if x < 0.0:
        erf = -erf
    return erf

if __name__=='__main__':
    n = 100000 + 1
    x0 = -10.0
    max_err_assoluto = -99999.
    h  = 2.0*abs(x0)/float(n-1)
    for i in range(n): #emulazione DO ... I = 0,N-1
        x = x0+h*float(i)
        y = math.erf(x)
        d = abs(round(approx2_erf(x),7) - round(y,7))
        max_err_assoluto = max(max_err_assoluto,d)
    print("max err. assoluto: %.7f" % max_err_assoluto)
```

Lanciando il programma si ottiene nell'intervallo [-10;+10]

```
max err. assoluto: 0.0000002
```

```
      DOUBLE PRECISION FUNCTION ERF(X)
C...
C...  FUNCTION ERF COMPUTES THE ERROR FUNCTION ACCORDING TO
C...  THE HASTINGS APPROXIMATION [GREENBERG (1978, P11)]
C...
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C...
C...  THE FOLLOWING CODING IS A STRAIGHTFORWARD
C...  IMPLEMENTATION OF THE HASTINGS APPROXIMATION
C...  CITED ABOVE
      T=  1.0D0/(1.0D0+0.3275911D0*X)
      A1= 0.254829592D0
      A2=-0.284496736D0
      A3= 1.421413741D0
      A4=-1.453152027D0
      A5= 1.061405429D0
      ERF=1.0D0-(A1*T+A2*T**2+A3*T**3+A4*T**4+A5*T**5)
         *DEXP(-X**2)
      RETURN
      END
```

Figura 7

$$\operatorname{erf} z = \begin{cases} 1.128z\,, & 0 \le z \le 0.15 \\ -0.0198 + z(1.2911 - 0.4262z)\,, & 0.15 \le z \le 1.5 \\ 0.8814 + 0.0584z\,, & 1.5 \le z \le 2 \\ 1\,, & 2 \le z \end{cases}$$

Figura 8

Applicazione del metodo di Newton-Raphson per calcolare erfinv(x)

Data una equazione f(x) = 0 e un opportuno valore di "innesco" x0, il metodo di Newton-Raphson afferma che la successione

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

convergerà alla soluzione desiderata x.

Nel caso della equazione f(x) = erf(x)-y = 0 si avrà:

$$x_{n+1} = x_n - \frac{2}{\sqrt{\pi}}\left(\operatorname{erf}(x_n) - y\right)e^{x_n}$$

con valore x0 (Strecok,[Bee17]):

$$x_0 = \sqrt{-\log\left((1-y)(1+y)\right)}$$

L'argomento del logaritmo è stato fattorizzato per limitare i fenomeni di cancellazione nelle operazioni con i numeri floating-point.
Il programma **erfinv_nr.py** effettua il calcolo di erfinv(y).
La scelta di limitare il range di utilizzo ai valori di y appartenenti a (erf(-4.50), erf(+4.50)) e di

interrompere l'iterazione quando abs(x_n+1 - x_n) < 10**6 sono dovute agli errori di arrotondamento rilevati durante la fase di test.

Il programma **erfinv_nr_test.py** stima il numero massimo di iterazioni nel range (erf(-4.50), erf(+4.50)) per mostrare che la limitazione i<20 è sensata.

```
#erfinv_nr.py
"""Calcolo erfinv(y) con il metodo di Newton Raphson"""
# x argomento della funzione
# float in Python equivale per default a DOUBLE del Fortran
#
import math
#
def approx_erfinv(y):
    i = 0
    delta = 1.
    z = abs(y)
    c = 2.0/math.sqrt(math.pi)
    xn = math.sqrt(-math.log((1-z)*(1+z)))
    while abs(delta) > 0.000001 and i < 20:
        delta = (math.erf(xn)-z)* c * math.exp(xn*xn)
        xn = xn - delta
        i += 1
    if y<0:
        xn = -xn
    return xn
if __name__=='__main__':
    print (math.erf(-4.50),round(approx_erfinv(math.erf(-4.50)),6))
    print (math.erf(+4.50),round(approx_erfinv(math.erf(-4.50)),6))
```

Lanciando il programma si ottiene

```
-0.9999999998033839 -4.5
0.9999999998033839 -4.5
```

```
#erfinv_nr_test.py
"""Test numero iterazioni del calcolo di erfinv(x) con il metodo NR"""
# x argomento della funzione
# float in Python equivale per default a DOUBLE del Fortran
#
import math
#
def approx_erfinv2(y):
    i = 0
    delta = 1.
    z = abs(y)
    c = 2.0/math.sqrt(math.pi)
    xn = math.sqrt(-math.log((1-z)*(1+z)))
    while abs(delta) > 0.000001 and i < 20:
        delta = (math.erf(xn)-z)* c * math.exp(xn*xn)
        xn = xn - delta
        i += 1
    return i
if __name__=='__main__':
    n = 100000 + 1
    imax = -1
    x0 = -4.50
    h  = 2.0*abs(x0)/float(n-1)
    for j in range(n): #emulazione DO ... J = 0,N-1
        x = x0+h*float(j)
        imax = max(approx_erfinv2(math.erf(x)),imax)
    print ("max iterazioni:",imax)
```

Lanciando il programma si ottiene

```
max iterazioni: 12
```

Bibliografia

[Ale92] V. Alexiades, "Mathematical Modeling Of Melting And Freezing Processes", CRC Press, 1992.

[Bee17] N.H.F. Beebe, "The Mathematical-Function Computation Handbook: Programming Using the MathCW Portable Software Library", Springer, 2017.

[Con18] G. Conti, Matematica e rischio di credito (ultima consultazione: 5 marzo 2018). http://www.mathisintheair.org/wp/2016/10/matematica-e-rischio-di-credito-parte-prima/

[Har14] F.E. Harris, "Mathematics for Physical Science and Engineering: Symbolic Computing Applications in Maple and Mathematica", Academic Press, 2014.

[Gla13] P. Glassermann, "Monte Carlo Methods in Financial Engineering,cSpringer Science & Business Media", 2013.

[Hie15] D.E. Hiebeler, "R and MATLAB", CRC Press, 2015.

[Jel13] B. Jelen, "Excel 2013 In Depth", Que Publishing, 2013.

[Mie14] P. Van Mieghem, "Performance Analysis of Complex Networks and Systems", Cambridge University Press, 2014.

[Reg93] F.J. Regan, "Dynamics of Atmospheric Re-Entry", AIAA , 1993.

[SH12] S. Sirca, M. Horvat, "Computational Methods for Physicists: Compendium for Students Graduate Texts in Physics", Springer Science & Business Media, 2012.

[SS97] W.E. Schiesser, C.A. Silebi, "Computational Transport Phenomena: Numerical Methods for the Solution of Transport Problems", Cambridge University Press, 1997.

RetroMath: Il villaggio di Babbo Natale

di Giuseppe Fedele

Il viaggio è stato lungo, tre voli: Lamezia Terme-Milano Malpensa, Milano Malpensa-Helsinki ed Helsinki-Rovaniemi per un totale di dieci ore di aereo circa escluse le pause nei vari aeroporti; ma il desiderio e l'impazienza per quell'appuntamento, pianificato da circa sei mesi, fa sembrare lieve ogni fatica. E' un desiderio che ognuno porta con sé!!!
Rovaniemi è il capoluogo della regione della Lapponia.
Non troppo distante dal centro, in corrispondenza del Circolo Polare, si trova il villaggio di Babbo Natale che è considerato la sua residenza ufficiale e dove è possibile incontrarlo personalmente e visitare il suo studio o l'ufficio postale dove vengono recapitate le lettere dei bambini di tutto il mondo. Ma Rovaniemi è anche importante per le aurore boreali.
Due spettacoli in uno!!!
L'appuntamento è previsto per le 21:00. Ho tempo per passare dall'hotel, sistemarmi, fare una doccia e mangiare la zuppa di salmone che è uno dei piatti tipici.
Arrivo per le 20.40, sono troppo emozionato tanto che i 6 gradi sotto zero nemmeno si avvertono. Non c'è già nessuno nel villaggio, l'orario delle visite è fino alle 20.00. Faccio un giro e qualche foto (Figura 1).

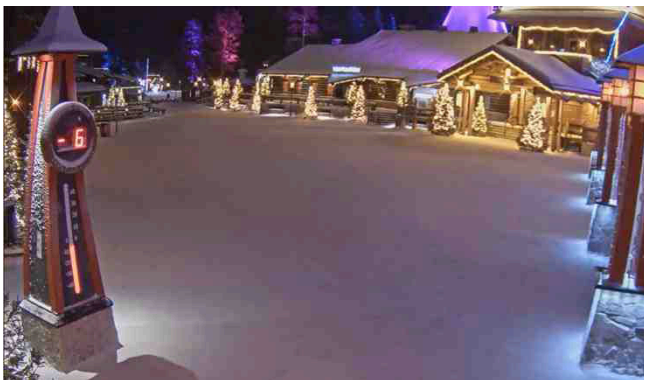


Figura 1 - Villaggio di Babbo Natale (04/12/2019)

Mi avvicino allo studio di Babbo Natale, so che farò fatica a ricordarmi tutte le cose che devo chiedergli. All'ingresso c'è un elfo, (non che sia sicuro, visto che non ne ho mai incontrato uno), si chiama Kilvo, mi dice che è il segretario speciale di Babbo Natale, è lui che gestisce tutte le richieste e che pianifica gli appuntamenti. Mi fa accomodare nella sala d'aspetto e se ne va. Dopo un pò rientra con un vassoio di "kampanisu", dei biscotti a pettine (Figura 2). Non ho fame, o meglio ho lo stomaco chiuso, ma per non sembrare scortese ringrazio e ne assaggio uno. Kilvo mi dice che Babbo Natale sarà lì a momenti.
Dopo meno di cinque minuti infatti Kilvo ritorna e mi invita a seguirlo. La stanza verso cui ci dirigiamo è

Figura 2 - Kampanisu

alla fine di un corridoio. Le gambe mi tremano e non per il freddo. Kilvo bussa alla porta, apre senza attendere risposta e annuncia la mia visita: "è venuto a trovarti Giuseppe dall'Italia"; mi fa entrare ed esce chiudendo la porta.

La stanza è grande, piena di quadri alle pareti in legno, un tavolo di cristallo al centro della stanza con delle sedie intorno. Il rosso è realmente il colore prevalente. Sono immobile vicino alla porta. Lui si alza, si avvicina verso di me, mi abbraccia forte. Non so quanto sia durato quell'abbraccio ma mi è sembrato simile a quello scambiato tra due amici che non si vedono da una vita. Inizia a parlarmi in italiano, ...Babbo Natale parla tutte le lingue del mondo.
Mi fa accomodare su una sedia, lui si siede dall'altra parte del tavolo e mi dice: "Trovo davvero singolare che tu sia venuto da così lontano senza portare i tuoi figli. Qui vengono sempre a trovarmi bambini, mai adulti da soli. A cosa devo il piacere della tua visita?"

Vagli a spiegare che è sempre stato il mio sogno quello di conoscerlo! Figurati se non lo sa già. Gli dico che sicuramente ritornerò a trovarlo e che in quell'occasione porterò anche i miei figli. Ma la mia curiosità sta diventando ossessiva: "sono venuto per chiederti una cosa".
"Dimmi pure".
Mi sento uno sciocco, la domanda è sempre la stessa: "come fai a consegnare i regali a tutti i bambini del mondo in una sola notte?"

"Questo è il mio mestiere" mi dice, "lo faccio ormai da duemila anni; nel tempo ho affinato le mie tecniche. In principio non riuscivo ad essere così efficiente".

Come per dire, ognuno fa il suo lavoro e non tutti possono fare tutto! Non mi basta, ho fatto dieci ore di aereo per sentirmi dire questo? Incalzo: "sai mi

piace molto cercare di capire e modellare i fenomeni che avvengono intorno a me; mi piacerebbe trovare una spiegazione a questa tua capacità; dimmi che è magia e la finiamo qui!"

Una risata rimbomba per la stanza, è una risata che mette allegria, è contagiosa: "ho, ho, ho…"

"Ma quale magia! Saprai benissimo che per un oggetto in moto, il tempo tende a rallentare tanto più la sua velocità si avvicina a quella di propagazione della luce nel vuoto; questo mi permette di prendere tempo."

Perché dovrei saperlo benissimo? Penso tra me e me…

"Non dovete pensare che la slitta sia in una determinata posizione spaziale in un determinato istante temporale. Come in un sistema quantistico, il comportamento della slitta va descritto in termini di funzione d'onda e densità di probabilità".

Inizio a manifestare cenni di cedimento! Ho letto un pò di fisica quantistica, di densità di probabilità, processi stocastici, ecc., ma…
"È inutile pensare che io sia in un determinato punto dello spazio", continua, "è più conveniente descrivere la mia posizione come una funzione della probabilità, in maniera analoga a quanto si fa per la posizione degli elettroni in un atomo".

Cerco di spiegargli i miei pensieri: "quindi stai dicendo che pur non sapendo dove tu sia ad ogni istante, possiamo dire grossolanamente dove è più probabile che ti trovi. Ciò vuol dire che preso un volume di riferimento potresti essere considerato come distribuito più o meno in ogni un punto di quel volume?"
"Più o meno è così, in questo modo riesco ad essere in più case contemporaneamente e a consegnare più pacchi velocizzando di molto il mio lavoro".
Boh, non sono un fisico, e tra il dire e il fare, cioè tra la teoria e la pratica, so che può esserci un abisso. "Mi risulta difficile comprendere quello che dici, mi verrebbe da chiederti come un oggetto con la massa pari a quella della slitta con le renne, tu di sopra e tutti i regali possano viaggiare alla velocità della luce".
"Cosa vorresti insinuare? Forse che mangio troppo! Oh, oh oh… Considera anche la possibilità di sfruttare l'effetto tunnel cioè la capacità di una particella con una massa e una determinata energia di attraversare una barriera energetica".

Mah, se corro contro un muro credo sia molto improbabile che riesca a oltrepassarlo e molto probabile che mi spacchi la testa. Mi sa di visionario…
"C'è poi un'altra cosa che mi piacerebbe sapere. Come pianifichi l'itinerario del tuo viaggio?"
Babbo Natale si alza e mi chiede di seguirlo. Nella stanza, quasi nascosta da un grosso albero alto fino al soffitto, c'è una porta. Da quella porta, una scala porta al piano di sotto: un'enorme openspace con tante stanze ricavate sfruttando delle pareti prefabbricate ad altezza d'uomo.
"Questo è il centro di calcolo del villaggio. Qui si trovano i supercomputer più potenti al mondo: due supercomputer IBM, Summit e Sierra, che si basano su CPU Power 9 e sulla GPU Nvidia V100; il supercomputer cinese Sunway TaihuLight che utilizza oltre 10 milioni di core del processore SW26010. Un Tianhe-2A (Milky Way-2A), della National University of Defense Technology (NUDT) cinese che utilizza una combinazione processori Intel Xeon e Matrix-2000. Frontera che monta 16.000 processori per un totale di quasi mezzo milione di core, integrando Xeon Platinum 8280 con sottosistemi Nvidia e IBM Power".

"Non è che assumi ingegneri? Chi non si innamorerebbe di questo posto?".

Ma un'altra cosa che attira la mia attenzione è un box al centro di questa openspace: un quadrato al centro di un rettangolo. Le pareti di questo box sono più alte e a differenza delle altre arrivano fino al soffitto. Sembra un corpo a sé isolato dal resto. Sarà una sala riunioni, penso.

"Qui al villaggio spendiamo molto in nuove tecnologie, ci serve avere supercomputer in grado di velocizzare tutte i calcoli che dobbiamo fare".

Ma che dovranno mai computare con tanta potenza di calcolo? Mi chiedo.

"In quella stanza che vedi al centro…", mi ha letto nel pensiero, "…conserviamo alcune macchine che non usiamo ormai da tempo. Lì abbiamo creato un piccolo museo di retrocomputing".

I miei occhi brillano, "posso visitarlo?" chiedo con voce sommessa. "Certo" mi dice e si avvia verso la porta.

Ecco il terzo spettacolo (oltre al villaggio e all'aurora boreale): uno stanzone di duecento metri quadrati con una serie di scaffali (tipo supermercato) pieni zeppi di retroPC tutti funzionanti.
Potrei rimanere qui fino a Natale prossimo fermandomi ad ammirare un Amstrad CPC 464, un Osborne 1, una serie di PET, tutta la serie di Commodore, un Epson HX-20, Olivetti di ogni tipo, ecc.

"Sono curioso di sapere che tipi di calcoli fate per avere necessità di tutte queste macchine", gli chiedo.

"Mi hai chiesto come pianifico l'itinerario del mio viaggio; vieni andiamo nella sala riunioni dove un mio collaboratore ti parlerà di uno dei problemi su cui

lavoriamo da tempo".

Nella sala riunioni mi viene presentato Mathin, un altro elfo che è a capo di una squadra di circa trenta persone che lavorano su algoritmi di ottimizzazione.
Mentre lui spiega scrivendo formule alla lavagna, io prendo appunti. Babbo Natale ne approfitta per fare un pisolino su una sedia.

Quello su cui lavorano ininterrottamente è il **problema del commesso viaggiatore**. Un viaggiatore deve far visita a tutte le città iniziando e terminando il viaggio in una stessa città. La difficoltà del problema sta nel fatto che il viaggiatore deve passare una e una sola volta da ciascuna città e deve minimizzare la lunghezza del viaggio.

"E' ovvia l'importanza di questo problema nel nostro lavoro!" continua Mathin, "... e la cosa si complica maggiormente poiché in ogni città dobbiamo anche pianificare il percorso per visitare tutte le case!"

Inizio ad intuire la necessità di un centro di calcolo nel villaggio di Babbo Natale.

Il problema può essere descritto come segue:

Dato un grafo completo G, simmetrico e pesato (un grafo di dice simmetrico quando ogni arco può essere percorso in entrambi i sensi; pesato quando ad ogni arco viene associato un peso; completo quando tutti i nodi sono collegati tra loro) con V insieme dei nodi e E insieme degli archi; si definisce **cammino hamiltoniano** un cammino che tocca una ed una sola volta tutti i nodi del grafo. Un ciclo hamiltoniano è un cammino hamiltoniano il cui nodo di arrivo coincide con il nodo di partenza.

$$TSP=\left\{\begin{array}{c}(G,f,t):G=(V,E)\text{ è un grafo completo},\\ f\text{ è una funzione } V\times V\to Z,\\ t\in Z,\\ G\text{ è il grafo che contiene il tragitto di costo non superiore a } t\end{array}\right\}$$

Dato il grafo mostrato in Figura 3, un ciclo Hamiltoniano è $P=\{A,B,C,D,E,A\}$.

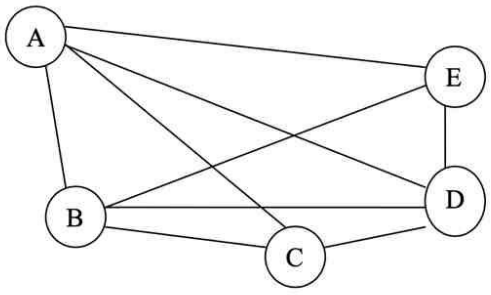


Figura 3 - Ciclo Hamiltoniano

Mentre Mathin mi parla di queste cose, mi ricordo di un articolo uscito qualche tempo fa su RetroMagazine su "Grafi, cammini e giochi" in cui venivano descritti alcuni di questi concetti. E ad un tratto la sua voce cambia tono "il problema è NP-completo e, poiché P è diverso da NP, o meglio non si è ancora dimostrato che P coincide con NP, non esiste alcun algoritmo risolutivo con complessità polinomiale".

Mi sa quasi di sfida e sono sicuro che per Mathin la cosa sia possibile e che ci stia lavorando da tempo insieme al suo team.

"Mi spieghi che soluzione avete trovato a questo problema?" gli chiedo. "Non possiamo svelare ancora i dettagli della nostra soluzione però posso farti vedere come implementare un algoritmo per il calcolo del ciclo hamiltoniano usando un programma logico tipo PROLOG".

Ma iniziamo con una struttura lista che può contenere elementi di qualsiasi tipo: [1,2,a,[5,6,b],c]. E' anche possibile esprimere le liste in termini di testa e coda. La testa è il primo elemento mentre la coda è costituita da tutti gli altri elementi della lista. L'operatore "|" separa la testa dalla coda: [X|Y].

Supponiamo di voler scrivere un predicato (una funzione in PROLOG) che verifichi se un elemento è membro di una lista. Occorre verificare se 1) l'elemento è la testa della lista; 2) l'elemento è membro della coda della lista. Quest'ultimo caso può essere risolto ricorsivamente dicendo che A è membro della lista [H|C] se è membro della sua coda C.

```
membro(A,[A|B]).
membro(A,[H|C]) :- membro(A,C).
```

Vogliamo adesso definire i collegamenti tra le città e le rispettive distanze; definiamo allora tanti predicati del tipo:

```
strada(A,B,100).
strada(B,C,150).
strada(C,D,85).
...
```

Il primo predicato sta a significare che la città A è collegata alla città B da una strada lunga 100, e così via. Poiché supponiamo che i collegamenti sono bidirezionali, dovremo prevedere un predicato del tipo:

```
coll(X,Y,C) :- strada(X,Y,C).
coll(X,Y,C) :- strada(Y,X,C).
```

A questo punto, definita una città di partenza Cp; una città di arrivo Ca ed una lista di città proibite Lp, occorre trovare un cammino Lc di costo C che

congiunge Cp e Ca e non contiene città proibite presenti in Lp.

Il caso base è quando Cp e Ca sono collegate direttamente ed entrambe non appartengono alla lista delle città proibite:

```
cammino(Cp,Ca,Lp,[Ca,Cp],C) :-
        coll(Ca,Cp,C),
        not(member(Ca,Lp)),
        not(member(Cp,Lp)).
```

Ricorsivamente possiamo adesso considerare una città intermedia Ci tra Cp e Ca, collegata direttamente a Cp, ed individuare un cammino tra Cp e la città finale Ca. Ovviamente Cp deve entrare nella lista proibita per evitare che venga inserita nuovamente nel cammino ed inserita in testa alla lista Lc delle città inserite nel cammino:

```
cammino(Cp,Ca,Lp,[Cp|LcTemp],C) :-
        coll(Ca,Ci,C1),
        not(member(Ca,Lp)),
        not(member(Ci,Lp)),
        cammino(Ci,Ca,[Cp|Lp],LcTemp,C2),
        C is C1+C2.
```

"E questo è tutto", continua Mathin, "... una soluzione certo elegante ma sempre inefficiente, del resto i programmi logici sono sempre molto espressivi!"

"Sai dove posso trovare un compilatore PROLOG per Commodore64?" gli chiedo.

"A questo link *https://www.lyonlabs.org/commodore/onrequest/collections.html* puoi trovare una serie di compilatori per Commodore 64 tra cui il PROLOG che cerchi".

Nel frattempo anche Babbo Natale si è svegliato.

"Grazie Mathin per le informazioni che mi hai dato e grazie per la piacevole serata passata in vostra compagnia. Vorrei chiedervi un'ultima cosa. Da circa un anno, con alcuni amici, abbiamo messo su una rivista di retrocomputing dal nome RetroMagazine. Se non vi dispiace mi piacerebbe scrivere un articolo sul problema che mi avete presentato".

"Nessun problema, anzi per i prossimi numeri anche noi potremmo contribuire!"

I miei elfi articolisti di retrocomputing, ho ho ho...

Fino ad arrivare là dove nessun C64 è mai giunto prima... Parte I

di Marco Pistorio

Sono un fan della serie fantascientifica "Star Trek" praticamente da sempre, come probabilmente tanti di voi che ci leggete. Da qualche tempo mi proponevo di realizzare una breve animazione che riuscisse a riprodurre, in qualche modo, delle scene oppure la sigla di apertura di questa serie, sfruttando un Commodore 64.
Ho recuperato su internet un video che lo riproduce, e mi rendo conto che "pesa" circa 2 Mbytes, una dimensione che risulta essere ben al di sopra della disponibilità del "commie", liscio, senza espansioni di memoria, SuperCPU e/o nuovi hardware et similia.
Estraggo dal video un certo numero di fotogrammi, ovvero i fotogrammi che potrebbero rappresentare la sigla che voglio riprodurre. Si tratta di circa 120 fotogrammi, a risoluzione 320x200 pixel.
Per memorizzare una schermata in modalità hi-res, monocolore, a risoluzione 320x200 pixel servono esattamente 8000 caratteri (cioè bytes).
Se si tratta di 120 fotogrammi occorrono 120x8000 bytes, ovvero all'incirca 960Kb.
Un floppy disk standard del Commodore 64 ha 664 settori, ognuno dei quali permette di memorizzare 256 bytes. Quindi un floppy disk può memorizzare fino ad un massimo di 664x256=166 Kbytes.
Ci vorrebbero quindi circa 5,6 floppy per la memorizzazione dei 120 fotogrammi.

E se, invece le 120 schermate hi-res si trasformassero in 120 schermate formate da caratteri ridefiniti, con 1000 caratteri per ciascuna schermata? Si tratterebbe allora di 120x1000 caratteri, cioè circa 120 Kb, con la limitazione che per ogni schermata si possa utilizzare un set con un massimo di 256 caratteri diversi l'uno dall'altro.
Inoltre, dal momento che i caratteri ridefiniti dovrebbero essere programmati per riprodurre tutte le 120 scene, si dovrebbero calcolare tutte le necessarie e diverse celle di dimensioni 8x8 pixel che formano i 120 fotogrammi. Dovremmo adoperare 4652 celle (tale numero è il risultato di un calcolo reale e, da tale risultato, si evince un fatto molto importante ovvero che molte celle in questi 120 fotogrammi si ripetono) e per ogni cella impiegare 8 bytes per la sua definizione, fino ad occupare altri 36,34 Kb per la ridefinizione dei caratteri necessari alla realizzazione di tutte le 120 scene. Siamo quindi a circa 120+37=157 Kb.
Si tratta anche stavolta di adoperare più di un intero floppy disk per contenere tutti i dati necessari.

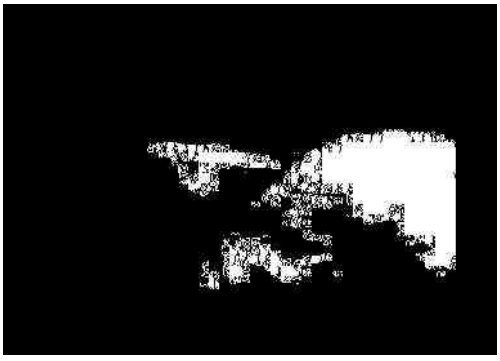


Figura 1 - Uno screenshot della animazione

In ambedue i casi abbiamo ragionato facendo una stima approssimativa, che non considera tra l'altro la memoria necessaria per contenere tutto il codice preposto al caricamento delle 120 schermate hi-res (oppure a quello preposto al caricamento delle 120 schermate carattere nonchè alla gestione del set di caratteri adoperato in ciascuna schermata), nè alla memoria necessaria per la generazione ed il controllo della traccia audio durante la riproduzione dell'intera animazione. Non dimentichiamoci che il C64 ha una RAM a bordo di "soli" 64 Kb, in ogni caso!

Sia la prima che la seconda strada sarebbero percorribili. Anche se la prima richiede più RAM per la memorizzazione dei 120 fotogrammi, potremmo risparmiare parecchio adoperando un sistema di compressione dei dati. La seconda sembra richiedere meno RAM ma necessita di una gestione accurata del set di caratteri che compone ciascuna delle 120 schermate.
E chiaro che entrambe le strade presentano qualche difficoltà da superare.

Ci sono strade più semplici, che ci potrebbero far ottenere una animazione di qualità quantomeno discreta, senza penare troppo, sfruttando solo la memoria a disposizione del Commodore 64, evitando caricamenti intermedi da dischetto? La risposta è sì. Come? Lo scopriremo nelle prossime puntate :)

Di seguito trovate il link grazie al quale potrete visionare l'animazione :
https://csdb.dk/release/?id=185478

Ringrazio **Gaetano Chiummo**, membro del gruppo "**Hokuto Force**", che ha realizzato la base audio presente all'interno di questa animazione, nonchè **Silvio Savarino**, che è il fondatore di questo gruppo di cracking/crunching, l'unico ancora attivo in Italia.

Non mi resta che augurare a tutti gli amici lettori di RetroMagazine buone feste e... restate sintonizzati!

MSX Games Report 2019

di David La Monaca

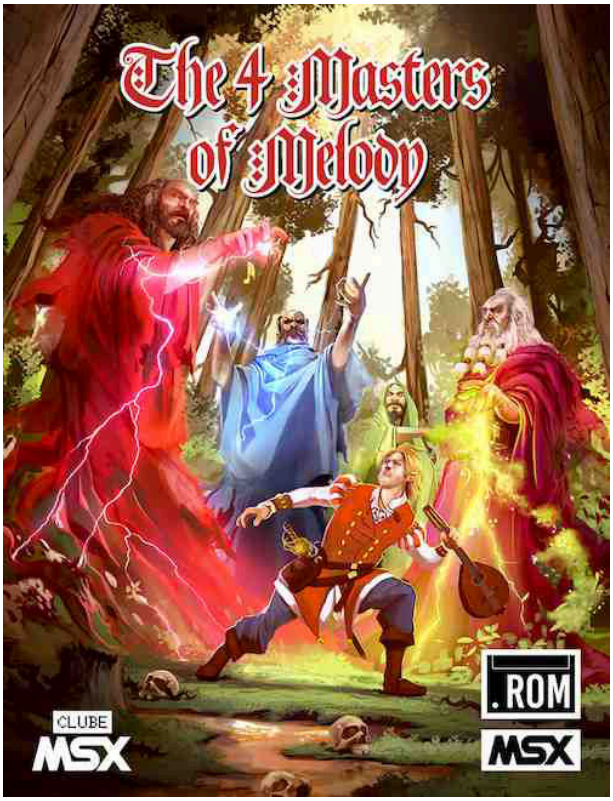
Siamo a fine dicembre 2019. Tempo di bilanci grandi e piccoli, personali e collettivi. Tempo di tirare le somme anche su dove sta andando il fenomeno del retrocomputing e sulla popolarità raggiunta dalle nostre amate piattaforme a 8-bit a circa 30-40 anni dalla loro comparsa sul mercato. La risposta, per sintetizzare al massimo, ce la fornisce una notizia che da pochi giorni è stata data dal popolare canale YouTube "The 8-Bit Guy" che ha raggiunto la cifra record di **un milione di iscritti** in tutto il mondo. Tutte le macchine godono di rinnovato interesse e sempre più appassionati di tutte le età, nostalgici e non, tornano ad accendere gli home computer che giacevano in qualche ripostiglio. Non mancano progetti di nuovo hardware e periferiche per emulare o riprodurre fedelmente il comportamento dei sistemi di un tempo e proliferano le idee e le realizzazioni di nuovi giochi o di porting da sempre attesi.

Anche nel mondo MSX si assiste ad una certa vivacità e le novità hardware/software si susseguono ad un buon ritmo. In questo breve articolo cerchiamo di ripercorrere l'anno appena trascorso segnalando alcune delle migliori produzioni software che riguardano il popolare standard a 8-bit. La totalità dei titoli che passeremo in rassegna gira infatti già a partire dagli MSX1. Senza la pretesa di essere esaustivi, ci basta mostrare quanta energia viene profusa e quanti sforzi vengono fatti in questo periodo per produrre giochi all'altezza dei bei tempi che furono.

The 4 Masters of Melody

Si tratta del primo gioco creato a partire da un progetto della popolare rivista "Clube MSX" e, al momento in cui scriviamo, sta per essere pubblicato nella versione cartuccia. Il lancio sul mercato è previsto per il 7 dicembre al MSX SP 2019 (http://msx.sampa.br), un evento riservato agli appassionati di MSX a San Paolo del Brasile. La versione in confezione boxata offre una grafica migliorata leggermente rispetto a quella del gioco, un manuale cartaceo in inglese e in portoghese, tutto ben confezionato in un package 16x20 cm, molto somigliante alle vecchie pubblicazioni dei titoli anni Ottanta.

Una copia verrà venduta al prezzo di 120 Real brasiliani, corrispondenti a circa 26,50 Euro, più spedizione. Inizialmente verrà prodotta in 30 esemplari soltanto, tutti numerati (una delizia per tutti i collezionisti!). Dopo la pubblicazione in occasione dell'evento MSX SP 2019, il gioco sarà disponibile presso il sito web della rivista Clube MSX



The 4 Masters of Melody

(www.clubemsx.com.br). Gli ordini internazionali verranno gestiti via email all'indirizzo contato@clubemsx.com.br ed il pagamento accettato è tramite Paypal.

Nel gioco impersonate Mirkell, un bardo di talento che si è perso nella leggendaria e spaventosa Foresta Melody. Soltanto per un caso fortuito, quattro entità note come i Quattro Signori di Melody riusciranno ad individuarvi. Gli enormi e potenti guardiani dall'aspetto di fantasmi decidono di aiutare Mirkell a lasciare la foresta, ma a una condizione: dovrà ripetere le venti melodie che essi suoneranno, dimostrando così di essere un coraggioso e brillante menestrello. Se Mirkell fallisce, però, i padroni scaricheranno su di lui tutta la loro furia! "The 4 Masters of Melody" è un gioco per MSX1 che funziona in modo simile al ben noto gioco "Simon", ma in un'ambientazione completamente fantasy. L'obiettivo è quello di far corrispondere la sequenza di note musicali suonate dai quattro maestri. Se si sbaglia anche una singola nota, è game over! Sviluppato da Mario Cavalcanti, il redattore capo di Clube MSX, il gioco funziona su qualsiasi MSX con almeno 16KB di RAM.

Uchūsen Gamma

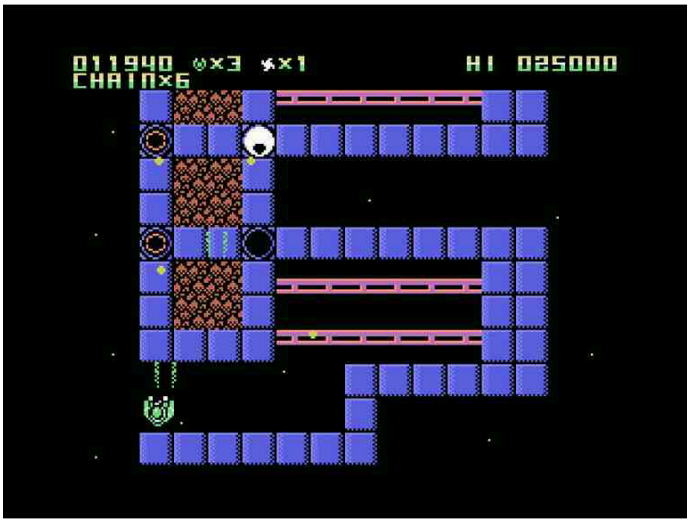
Uchūsen Gamma è uno sparatutto verticale pubblicato lo scorso 29 novembre 2019, sviluppato interamente da Juan J. Martinez (codice, grafica e suono). Juan è già noto alla comunità retrogaming per aver pubblicato altri titoli per sistemi a 8-bit come **The Dawn of Kernel** (Amstrad CPC), **Rescuing Orc** (Commodore 64), **Magica** (Amstrad CPC) ed il più recente **Night Knight**, un platform a schermo singolo per MSX (ne parliamo più avanti).

Il gioco offre la classica visual degli sparatutto ma con una particolarità: la ricarica appare solo quando 9 nemici vengono colpiti e distrutti in fila, in una "catena completa". Più lunga la catena, più punti saranno accumulati per ciascun nemico colpito e neutralizzato. C'è anche una componente di rischio/ricompensa in quanto potreste dover attendere di uccidere le navi nemiche fino a che ve ne saranno abbastanza sullo schermo per completare una catena intera.

Ecco le caratteristiche salienti del gioco:

- Azione classica da sparattutto, singolo giocatore
- Scorrimento verticale fluido
- 5 livelli con diversi tipi di nemici e boss di fine livello
- Controllo via tastiera o joystick
- Compatibile PAL ed NTSC
- Richiede un computer MSX computer con 16KB di RAM

Il gioco sarà disponibile via download libero da usebox.net e in seguito verrà realizzata una confezione "fisica" su cartuccia dalla Repro Factory (pre-ordini a partire da dicembre).



Uchūsen Gamma

Link: https://www.usebox.net/jjm/uchusen-gamma/
Repro Factory: http://www.repro-factory.com/

Nohzdyve

Giovanni Nunes ha pubblicato sul suo blog (in Portoghese) una versione di Nohzdyve per la piattaforma MSX. Il gioco è apparso nell'episodio chiamato Bandersnatch della serie Black Mirror, uno spettacolo TV interattivo ambientato negli anni Ottanta in cui il personaggio principale è uno sviluppatore di giochi per computer.
Nello show, il gioco Nohzdyve è uno di quelli pubblicati dalla **Tuckersoft**, una software house fittizia, per lo ZX Spectrum. Alla fine della puntata è apparso una sorta di easter egg (la cui spiegazione invito a cercare) che vi permette di scaricare una versione giocabile di Nohzdyve per ZX Spectrum. Il gioco ha una struttura molto semplice, ma il fatto che sia stato prodotto esclusivamente per la serie TV, lo rende istantaneamente un classico.
Giovanni cercava un gioco semplice per provare a se stesso di poterlo sviluppare su MSX usando qualcosa di simile all'architettura MVX. Il risultato della sua impresa è una perfetta versione del gioco originale.
Se volete sapere come è stato sviluppato o farci una partitina veloce, seguite il link qui sotto. Se non avete hardware MSX o un emulatore pronto all'uso, potete anche usare il formidabile WebMSX.

Nohzdyve

(Link Binary and Sources)
https://github.com/plainspooky/nohzdyve/releases

(Giovanni's Blog)
https://giovannireisnunes.wordpress.com/2019/06/28/nohzdyve-para-msx/

Night Knight

Juan J. Martínez ha annunciato la disponibilità del suo nuovo gioco per lo standard MSX, chiamato **Night Knight**.

La storia del gioco inizia quando Sir Bernard, un anziano cavaliere, viene colpito dalla maledizione di una strega che gli impedisce di dormire fino a che egli non avrà percorso un lunghissimo cammino. E allora il nostro eroe dovrà completare gli 80 livelli che lo aspettano e schivare orde di nemici che cercheranno in ogni modo di bloccargli la strada. Sir Bernard può andare a sinistra, destra e saltare sulle piattaforme più alte o sopra i suoi nemici per neutralizzarli.

La struttura del gioco ricorda molto quella di Miner 2049er, dove il protagonista deve muoversi attraverso molte piattaforme che cambiano colore al suo passaggio. Quando tutti i percorsi sono stati completati, apparirà una chiave di cui si ha bisogno per sbloccare l'uscita del livello e in questo modo completarlo.

In Night Knight è stata aggiunta una variabile tempo per rendere le cose un tantino più difficili e per proporre una svolta al genere platform: tutti i livelli devono essere completati in meno di 60 secondi. In alcuni dei livelli proposti, oltre ai nemici, appaiono oggetti che possono resettare il tempo o fermarlo per aiutarvi nella vostra missione finale. Non sono previsti oggetti che donano vite extra all'eroe del gioco.

Inoltre, come già detto in precedenza, il cavaliere può saltare soltanto verso le piattaforme poste più in alto. Per raggiungere quelle più in basso, deve saltare in basso in punti specifici dello schema di livello e cambiare direzione a mezz'aria.

Non solo il tocco dei nemici può uccidere il nostro eroe. In alcuni livelli compaiono nemici in grado di lanciarvi contro frecce, in altri ci saranno trappole piene di punte affilate ad attendervi. Quindi siate pronti ad evitare ogni pericolo, anche non evidente!

Night Knight è anche offerto da **Poly.play Software** in una speciale edizione in scatola che contiene:

- La cartuccia compatibile con PAL ed NTSC.
- Scheda MicroSD con l'immagine binaria della cartuccia, la Colonna Sonora in formato MP3 e molto altro
- Confezione in cartone
- Manuale di 8 pagine in format A5 (inglese)
- Adesivi del gioco
- Poster (formato A3)

Tutto al prezzo di 35 Euro più spedizione. Naturalmente, se desiderate solo giocare alla versione digitale, questa è gratuita e disponibile al link presente qui sotto. E' anche possibile giocare online col vostro browser usando lo strabiliante WebMSX (il link carica il gioco automaticamente).

Personalmente, ci ho giocato per qualche minuto ma ho solo raggiunto il livello 3! Se siete curiosi di provare voi stessi, mandateci i vostri punteggi!

Night Knight

(Download Night Knight gratis)
https://www.usebox.net/jjm/night-knight/

(Acquisto dell'edizione speciale per collezionisti)
https://www.polyplay.xyz/Night-Knight-Collectors-Edition-Cartridge

Prisoner Of War

Francisco Téllez de Meneses è il programmatore che già a giugno di quest'anno aveva annunciato il suo lavoro su un nuovo titolo per MSX. La buona notizia è che gli appassionati gamer non dovranno attendere molto a lungo per mettere le mani sul gioco finito. I pre-ordini per questo game ispirato a Metal Gear sono ora attivi, le edizioni in inglese e spagnolo sono disponibili online sullo web store di Matra.

Le cartucce sono vendute al prezzo di 35 Euro (spedizione a parte) ed i clienti possono scegliere fra la consegna al proprio indirizzo oppure il ritiro di persona al prossimo RUMSX 54 di Barcellona l'anno prossimo. Secondo quanto riportato dall'annuncio su msx.org, i fan del gioco che hanno effettuato l'ordine prima del 15 novembre riceveranno una misteriosa sorpresa con l'ordine.

Lo scenario del gioco vi porta in una località militare sconosciuta di alta sicurezza. Siete stati catturati e imprigionati ed ora dovete fare tutto quello che è nelle vostre possibilità per sfuggire vivo e vegeto da questa prigione.

Secondo quanto riportato dallo stesso sviluppatore il gioco è molto simile alle dinamiche del classico "stealth'em up" della Konami, Metal Gear, ed è

composto da 220 schermate differenti con tante caratteristiche e novità fra cui:

- 15 locazioni diverse (220 schermate)
- Diversi tipi di nemici, ciascuno con la propria personalità ed abilità
- Inventario di oggetti e armi
- Salvataggio dello stato corrente del gioco
- Finali differenti
- Colonna sonora PSG

Prisoner of War

Fonte: romhacking.ru

Revenge (Venganza)

Altro titolo particolare uscito per MSX. Particolare perché secondo quanto si legge sul sito web di riferimento, Revenge è un adventure game basato su testo, sviluppato interamente secondo lo stile classico in bianco e nero o a colori dei disegni a fumetti di un tempo. Tecnicamente riguardo al suo sviluppo software, il gioco è stato creato usando DAAD, un vecchio strumento professionale di programmazione specificatamente progettato per le avventure testuali. Si tratta di un tool talmente efficace che ancora oggi viene utilizzato per sviluppare storie interattive e Revenge ne è la prova.

In effetti le avventure testuali tornano a riscuotere un certo successo presso gli appassionati di retrogaming e per MSX la Physical Dreams ha lanciato sul mercato la traduzione in inglese di un titolo precedentemente disponibile solo in lingua spagnola. Il gioco si snoda attraverso un'intricata trama che, come il titolo suggerisce, conduce il protagonista da un letto d'ospedale fino alla completa vendetta contro chi aveva tramato alle sue spalle. Si gioca con la classica modalità a comandi dati da tastiera in base a ciò che succede nella storia, corredata da disegni grafici a colori o in bianco e nero. E' possibile mettere le mani su una copia di Revenge/Venganza in due formati diversi (il costo esclude la spedizione):

- Piccola scatola di cartone: 28€
- Grande confezione in plastica: 33€

Tutte le informazioni su Revenge/Venganza si ottengono contattando Physical Dreams all'indirizzo physicaldreamsgames@gmail.com.

Revenge (Venganza)

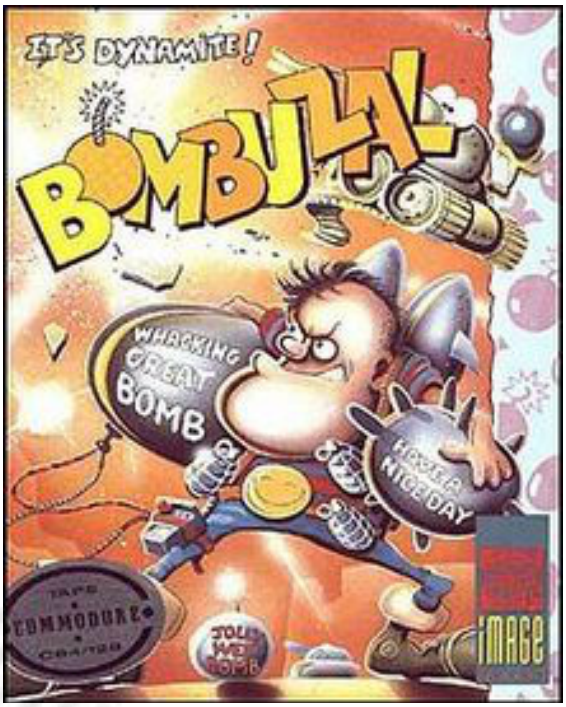
Conclusioni

Altri giochi sono stati annunciati per MSX durante il 2019 (Risky Rick, Barbarian The Duel, ecc.) e molti altri titoli in preparazione vedranno la luce solo nel 2020, per cui tutti gli appassionati videogamer, fan di questa piattaforma, possono stare tranquilli e godersi le molte novità che li attendono. Una cosa è certa: il retrogaming MSX è vivo e vegeto!

LE COPERTINE DEI VIDEOGIOCHI

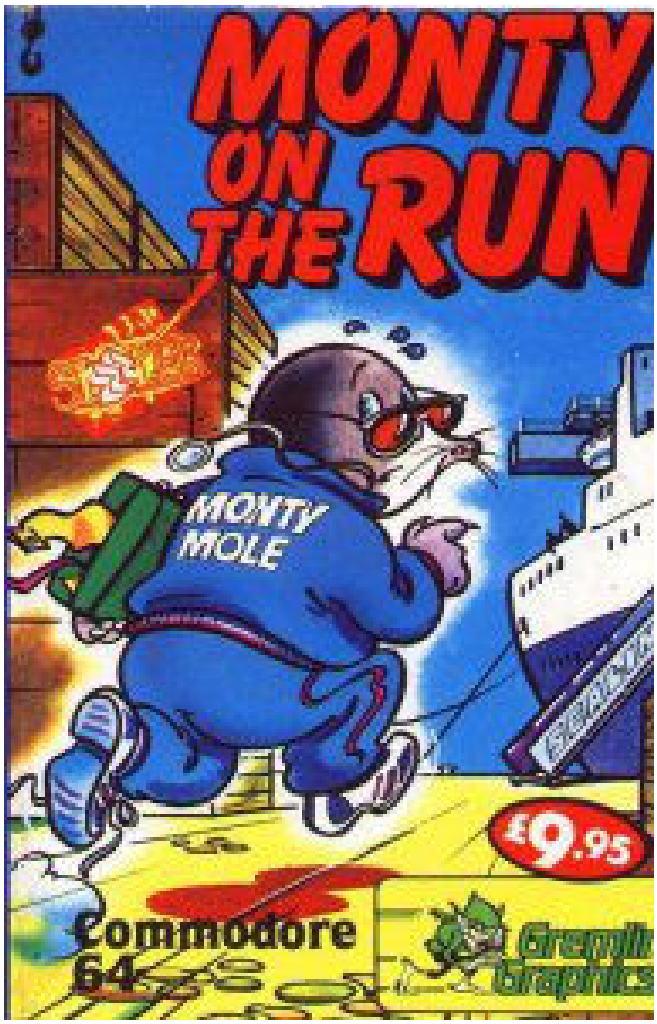
di Daniele Brahimi

Quanti di noi sono rimasti affascinati di fronte alle vetrine dei negozi di computer dell'epoca che esponevano videogiochi per Commodore 64, Spectrum e tutti gli altri home computer? Vetrine ricche di meravigliosi disegni sulle confezioni di videogiochi che ritraevano scene dello stesso.
Copertine fatte indubbiamente da bravissimi disegnatori che catturano i collezionisti e ai giorni nostri reperibili tramite un qualsiasi motore di ricerca, condivise su gruppi e forum facendoci rivivere momenti magici e al tempo stesso cariche di nostalgia. Ma arriviamo anche al punto sfavorevole; molti, anzi troppi sono rimasti delusi del fatto che la copertina fosse fatta benissimo, mentre il contenuto (il gioco stesso) avesse grafica e giocabilità scarse, senza contare magari anche la difficoltà. Anche se i giochi deludenti ci sono stati e sempre ci saranno.



Io sono il primo che si fa ammaliare da copertine come quelle dell'epoca anche per il solo gusto di averle nella propria collezione, però la sostanza andrebbe vista con altri occhi perché bene o male si sapeva e si sa il livello grafico e la giocabilità di quei computer... in più c'è anche il detto che dice che un libro non si giudica dalla copertina o che l'apparenza inganna.

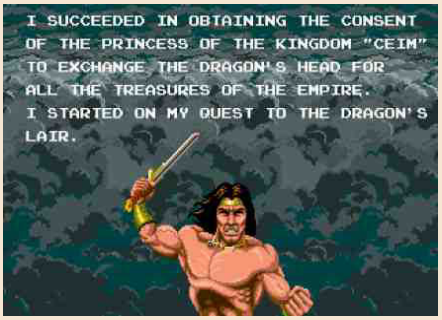
Forse è stata una strategia di mercato, chi può dirlo? (Sicuramente - NdR) Oppure i giochi erano destinati anche ai collezionisti, come avviene per quelli attuali. Forse i miei consigli al giorno d'oggi non hanno più alcun valore visti gli anni che sono passati o forse sì perché qualcuno che si soffermi alle apparenze è rimasto (senza puntare il dito contro nessuno perché lo feci anch'io quell'errore in passato).



Prima di procedere all'acquisto di un gioco, leggete recensioni da più fonti sia da parte di italiani che da stranieri, questi ultimi con punti di vista a volte differenti dai nostri riguardo le caratteristiche di un videogioco, guardate video di presentazione, giocate alle versioni dimostrative, infine pensateci prima di dormire se i soldi saranno spesi bene... Se poi anche la copertina è da mostra meglio, due piccioni con una fava! Con questo articolo voglio augurarvi buone feste con i vostri retrocomputer e retrogame, giocate molto e ricordate i bei tempi che furono quando per tutto il periodo natalizio questi videogiochi ci accompagnavano sia come decorazioni natalizie, come divertimento e come motivazione per godersi meglio i giorni in cui si stava a casa da scuola.

RASTAN SAGA

Publisher: Taito
Anno: 1987
Piattaforma: arcade
Genere: Platform



Il tema fantasy ha da sempre ispirato tanti videogiochi sia moderni che del passato. Sarà per l'ambientazione ricca di creature mitologiche e fantastiche, sarà per la presenza di magia e incantesimi o per l'uso massiccio di spade ed armature, fatto sta che questo genere era molto caro ai platform e beat'em up dell'epoca d'oro della sala giochi e delle console casalinghe.

Tra i tanti titoli prodotti vi voglio parlare di **Rastan Saga**, conosciuto in Europa e Stati Uniti semplicemente con il nome di Rastan.

Lanciato dalla Taito nel 1987 per il mercato arcade, Rastan può essere riconducibile al mondo degli Hack'n Slash e dei platform a scorrimento laterale.

Protagonista è un muscoloso barbaro che, armato della sua fedele spada, deve attraversare lande e castelli infestati da ogni tipo di mostro.

Sulla reale storia del gioco si sa davvero poco. Nella versione demo mode compare una schermata in cui il nostro eroe, ormai vecchio e seduto su un trono, ci invita ad ascoltare la sua storia senza nascondere di essere stato da giovane un ladro e un assassino. Invece nella versione giapponese compare un'introduzione nella quale si racconta che la principessa del regno di Ceim ricompenserà con ogni ricchezza l'eroe che le porterà la testa di un terribile e spietato drago.

Dal punto di vista grafico, Rastan colpisce subito per i dettagli e i colori perfettamente combinati negli sprites dei vari personaggi e negli sfondi. Ma anche la giocabilità costituisce un ulteriore punto di forza di questo titolo visto che muovere il nostro eroe, sia nella fase di esplorazione che in quella di combattimento, è davvero divertente.

Inizialmente abbiamo come arma

una spada che possiamo usare per gli scontri ravvicinati, ma nel corso della nostra avventura possiamo raccogliere asce, mazze ferrate, martelli ed una magica spada che lancia palle infuocate con la quale è possibile colpire anche a distanza.

Spesso i nostri nemici rilasciano degli items come scudi, mantelli, pozioni magiche e gioielli i cui benefici sono descritti in modo chiaro in una schermata prima dell'inizio del gioco. I mostri che incontriamo nel nostro cammino appartengono soprattutto alla mitologia occidentale (uomini rettili, arpie, druidi, pipistrelli giganti) mentre i classici boss che ci aspettano ad ogni fine livello sono presi dal mondo medievale (guerrieri, stregoni e, come ci anticipava l'introduzione nipponica, il temibile drago finale). Diversi anche gli ostacoli da superare come fiumi di lava, cascate, spuntoni e precipizi che possiamo evitare solo saltando da una corda all'altra.

Diversi gli adattamenti per le console e computer del periodo tra i quali la conversione per il Sega Master System, per qualità grafica e narrazione, è quella che si avvicina maggiormente al titolo originale.

Dopo un solo anno dall'uscita nelle sale giochi, la Taito ha cercato di bissare il successo con **Rastan Saga II**, conosciuto semplicemente con il nome di Nastar (chiaro l'anagramma con il nome originale), ma discostandosi forse troppo dal gameplay originale e a causa anche di un bug nel quarto livello, il titolo non ha fatto breccia nel cuore dei videogiocatori come il suo predecessore.

Nel 1991 la Taito chiude la trilogia facendo uscire **Warrior Blade: Rastan Saga III**. Si tratta questa volta di un beat'em up a scorrimento laterale molto simile a Golden Axe. Questa volta, oltre al solito Rastan, possiamo scegliere anche il mercenario Dewey o la ladra Sophie, ognuno con proprie caratteristiche e armi.
La particolarità di questo titolo sta nel fatto che necessita di due schermi per seguire tutta l'azione dei nostri eroi e del fatto che è possibile scegliere il proprio percorso sulla mappa e quindi avere un finale ogni volta diverso. Purtroppo questo gioco, seppur tradotto in lingua inglese, non ha varcato i confini nipponici ed oggi è possibile giocarlo solo grazie all'emulazione.

Tornando al capostipite della saga, chiudo al recensione con un paio di curiosità. Innanzitutto è fin troppo chiaro che i suoi autori si sono ispirati al film Conan il Barbaro, interpretato da Arnold Schwarzenegger nel 1982, a cui il nostro eroe somiglia nell'aspetto fisico, nell'abbigliamento e nel modo di impugnare la spada. Infine la Taito ha inserito Rastan come cameo nel titolo Champions wrestler, gioco di lotta uscito del 1989, dove troviamo tra i vari personaggi anche un certo Miracle Rastan che, sia nel nome che nell'aspetto fisico, ricorda il mitico Rastan originale.

di **Querino Ialongo**

GIUDIZIO FINALE

» Giocabilità 80%

Un tasto per saltare e uno per colpire con la spada, Rastan non chiede di più.
Forse qualche mossa speciale avrebbe reso questo titolo ancora più giocabile.

» Longevità 80%

Niente da dire sul gameplay, Rastan è bello da giocare ancora oggi, infatti i suoi successori non sono riusciti a replicare il suo successo iniziale.

BLASTAWAY

Developer: Simone Bevilacqua.
Anno: 2019
Piattaforma: Windows e AOS4
Genere: Labirinto

Dopo sei mesi di intenso lavoro Simone Bevilacqua, fondatore e fact-totum di RETREAM, ha ufficialmente rilasciato la sua ultima fatica, Blastaway, un frenetico arena-shooter dal "look and feel" squisitamente retro.

Ispirato da classici quali "Wizard of Wor" (per il gameplay) e "The Chaos Engine" (per la parte grafica), Blastaway pone il giocatore all'interno di 5 diverse arene, ognuna composta da 10 livelli, in cerca della via di uscita.
Ogni livello è popolato da diversi tipi di mostri, rappresentanti diversi tipi di minaccia, ed il passaggio di livello è subordinato al reperimento di chiavi in grado di dare accesso alle distinte sezioni del labirinto ed alla conseguente eliminazione dei Wor.

Il gioco si presenta come un tipico labirinto con visuale dall'alto, lievemente inclinato in prospettiva per dare l'impressione della profondità. Gli imponenti muri che lo costituiscono possono essere resi parzialmente trasparenti raccogliendo il relativo power-up.
Una volta eliminati tutti i Wor del livello, due velocissimi miniboss faranno la loro comparsa in sequenza: corretto posizionamento nel labirinto e velocità di riflessi per sparare al boss saranno la chiave del successo. Stage dopo stage i nemici si susseguiranno senza sosta, in una sorta di affascianante scalata che, anche se apparentemente ripetitiva, in realtà pone diversi livelli di sfida ad ogni nuovo nemico che entra nell'arena.

Come tutti i giochi di Simone, anche Blastaway è caratterizzato dell'estrema cura posta nei particolari: dalla ricerca spasmodica della giusta palette di colori per dare al gioco un look retro, al bilanciamento del livello di difficoltà per fornire al giocatore l'adeguato livello di sfida senza frustrarlo all'eccesso.

Negli anni Simone si è particolarmente distinto per i suoi lavori sviluppati principalmente per Commodore 64, Amiga ("Classic") e CD32. Nel caso di Blastaway ha invece deciso di usare Windows come piattaforma di riferimento, anche se il gioco è stato reso disponibile per i sistemi Amiga "next gen", funzionanti con sistema operativo AmigaOS4. Nonostante il target, come detto, siano le moderne piattaforme Windows, Simone ha deciso di

Requisiti

Minimo: 500 MHz CPU, 32 bit graphics card, 16 bit sound card, 24 MB RAM, 15 MB disk, 32 bit binaries support, AmigaOS 4.1 or Windows XP. Opzionale: joypad with a digital pad and a digital button.

confezionare in tutto e per tutto un prodotto che sembra appena uscito dalle splendide riviste della microinformatica del passato: vediamo come ha fatto.

La grafica è stata sviluppata usando Gimp, completamente a mano, pixel per pixel, usando un pennello dalle dimensioni 1x1.
Normalmente i giochi di RETREAM vengono sviluppati usando Personal Paint su AmigaOS, ma alla fine anche Gimp si è rivelato un utile strumento nonostante il controllo della palette colori sia su Gimp un po' troppo limitato.

L'intero gioco utilizza una palette fissa a 32 colori, con le ombre che aggiungono 2 bit di profondità. I colori non sono stati scelti scientificamente, ma si è partiti da una piccola selezione di tonalità vibranti per raggi ed esplosioni, assicurandosi poi che alcune di esse fossero utilizzabili anche per il personaggio principale. Successivamente sono state aggiunte alcune delle tonalità pastello per le tessere (tiles) che compongono gli sfondi.
Infine, sono stati aggiunti altri colori a poco a poco, come richiesto dai nemici e dalle altre impostazioni, cercando di introdurre sfumature al di fuori dall'insieme dei colori già esistenti, ma comunque in un modo che si fondessero bene.
Si tratta di un approccio completamente iterativo, ed ogni volta che un nuovo colore viene aggiunto la grafica preesistente viene ritoccata qualora tale operazione apporti un miglioramento (ad esempio, sull'antialias o per rendere le cose più colorate).

Per quanto riguarda i suoni, parlato e rumori sono stati registrati dalla bocca di Simone, altri suoni dalla sua chitarra, il rumore delle chiavi battendone un mazzo con le dita e cose del genere. Il parlato è poi stato sintetizzato con Software Automatic Mouth per il C64, in passato già usato per registrare il parlato di Huenison, altro grande gioco di RETREAM. I suoni sono invece stati campionati con Audacity. Come si vede, Simone è dunque un creativo a tutto tondo, che nel passato si è anche cimentato con la scrittura ed ha pubblicato un interessante libro.

A differenza di tutti i suoi titoli precedenti, Simone ha deciso di distribuire gratuitamente Blastaway.

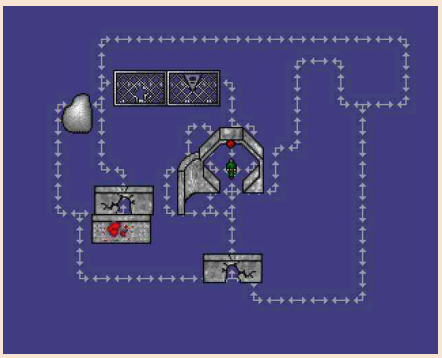
I link di download, per Windows e AOS4, si trovano nella pagina ufficiale del gioco all'indirizzo https://retream.itch.io/blastaway , mentre tutto il resto della produzione di RETREAM si trova all'indirizzo https://retream.itch.io/ .

Come detto in apertura di articolo l'autore non si ferma mai e, una volta conclusi i lavori su Blastaway, Simone è immediatamente ritornato su alcuni suoi giochi precedenti per effettuare il backport del nuovo sistema grafico che introduce nuovi filtri e nuovi livelli di zoom. In un Paese spesso martoriato dal pressapochismo, Simone costituisce un indubbio esempio di dedizione e professionalità.

Al momento in cui scrivo non ho ancora avuto la possibilità di finire completamente il gioco, per cui non mi è possibile espimere un giudizio finale tuttavia, basandomi sui livelli sin qui esplorati, azzarderei un 85% sia per la giocabilità che per la longevità.

Assicuratevi di dare un'occhiata al sito web di riferimento e, possibilmente, spargete la voce in giro: l'Italia torna a flettere i muscoli ed a mostrare programmatori di eccezione.

di **Gianluca Girelli**



GIUDIZIO FINALE

» Giocabilità 85%
» Longevità 85%

Al momento in cui scrivo non ho ancora avuto la possibilità di finire completamente il gioco, per cui non mi è possibile espimere un giudizio finale tuttavia, basandomi sui livelli sin qui esplorati, azzarderei un 85% sia per la giocabilità che per la longevità.

MAYHEM IN MONSTERLAND

Piattaforma: Commodore 64, poi Wii Virtual Console.
Data di pubblicazione: 1993
Pubblicazione: Apex Computer
Sviluppo: Apex Computer
Versione recensita: Commodore 64

Nel 1993 il Commodore 64 era sulla via del tramonto, da tempo imperversavano altri sistemi più potenti (console Sega e Nintendo) e il cugino Amiga lo aveva scalzato dai cuori dei commodoriani più incalliti.

Fu però un anno di grandi giochi e grandi produzioni, piccoli gioielli così curati da diventare indimenticabili.

Questo platform game dal classico scorrimento orizzontale, il cui protagonista è un draghetto ciccione è uno degli esempi di quanto ancora si potesse ottenere dal piccolo 64 bit se sfruttato a dovere.

La Apex ha realizzato quello che, per quanto mi riguarda, è uno dei migliori prodotti dal punto di vista grafico per la macchina Commodoro, raggiungendo un livello di dettaglio che faceva impallidire console più giovani.

La colorazione, il dettaglio dello schermo, la grandezza degli sprites e la velocità sono a livelli altissimi e sfruttano alcuni accorgimenti grafici che per l'epoca non erano per nulla scontati.

Le animazioni dei personaggi, degli avversari... tutto gira con una fluidità impressionante e tutto gira su una macchina che già nel 1993 era "alla fine" della corsa.

Le magie di chi sapeva sfruttare a dovere un prodotto, cosa che non sempre si è vista ai tempi e che molte volte anche al giorno d'oggi viene poco applicata.

Un gioiello di design e di gioco. Anche l'accompagnamento musicale è ben strutturato con musiche in crescendo durante l'azione di gioco e simpatici effetti sonori.

I livelli del gioco devono essere percorsi in entrambe le direzioni,

ovvero sia da sinistra a destra che da destra a sinistra. L'avventura è composta da cinque livelli, ognuno di questo deve essere percorso due volte: una di notte (Sad) e una di giorno (Happy). Il risultato è quello di ben 10 stage che si differenziano non solo dalle colorazioni dello schermo, ma anche dalla diversa tipologia di nemici presenti e dalle azioni da compiere per portarli a termine.

Durante la notte bisogna recuperare dei sacchi in possesso degli avversari, mentre durante il giorno è necessario raccogliere le stelline gialle sparse (ovunque) per tutto il livello. Ogni nemico che incontreremo nell'avventura presenta un diverso stile offensivo e punto debole, entrambi ben caratterizzati.

Il Drago Ciccione può vantare la classica "culata" alla Super Mario per schiaccaire gli avversari e, subito dopo il secondo stage, la "carica" alla Sonic. Due modalità di attacco differenti molto gradevoli e adatte alle esigenze del giocatore nelle circostanze e che garantirà un approccio "strategico" da scegliere in ogni diversa situazione.

Sistema di controllo perfetto, il protagonista si muove senza problemi con estrema precisione. Muoversi da una piattaforma all'altra saltanto "in su" sul joystick in questo prodotto non risulta snervante ed è ben calibrato.

Nota dolente... il livello di difficoltà altino e la longevità.. pur con le modalità giorno e notte.. è pur sempre un platform.

Se avete voglia di qualcosa di davvero bello da guardare e giocare sul vostro C64. Qualcosa che vi dimostra che attraverso un ottimo lavoro di programmazione e di game design si possono raggiungere vette incredibili, Mayhem è il prodotto che fa per voi. Una grafica così colorata, ben fatta e un gameplay così ben strutturato difficilmente si trova e ne vale la pena.

Un po' Mario... Un po' Sonic... Molto Commodore!!

di **Carlo Nithaiah Del Mar Pirazzini**

GIUDIZIO FINALE

» Giocabilità 98%

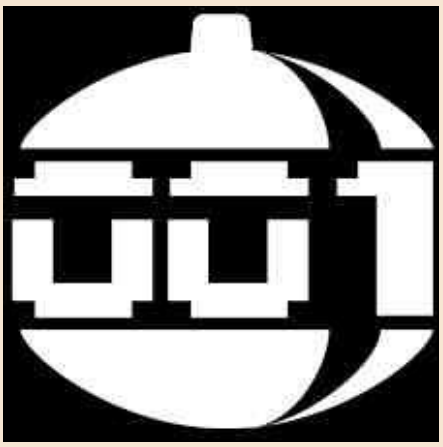
Controlli perfetti, ottimo gameplay, livelli strutturati con cura. Il sistema di gioco a due modalità giorno e notte è davvero geniale.

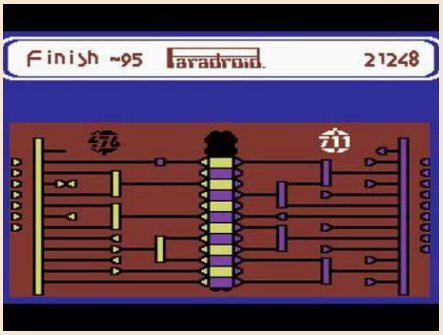
» Longevità 75%

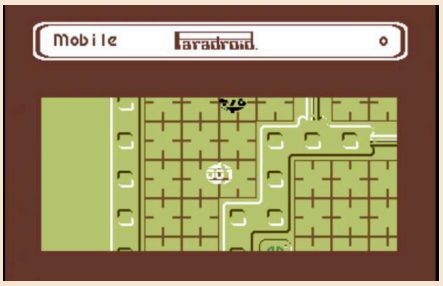
Difficilino per essere un platform puccettoso.. e, quando lo terminerete, purtroppo non lo rigiocherete subito.

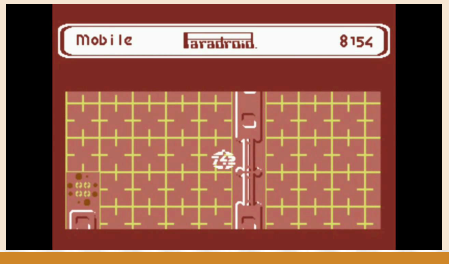
PARADROID

Piattaforma: Commodore 64, in seguito Wii Virtual Console.
Data di pubblicazione: 1985
Pubblicazione: Hewson
Sviluppo: Graftogld
Versione recensita: Commodore 64









Paradroid... un po sparatutto, un po rompicapo.. di genere fantascientifico. Grafica minimalista, sonoro a tratti angosciante ma ben fatto... geniale parto di Andrew Braybrook, programmatore di Uridium e Gribbly'S Day out.. pietre miliari dei videogames degli anni 80.

Paradroid, la storia del piccolo 001... considerato uno dei più grandi capolavori realizzati per il C64, nonostante la sua disarmante semplicità.
Leggevo che The Commodore 64 book: 30th Anniversary Special lo mette ai primi posti tra i migliori 25 giochi per l'8bit commodore.

Uno dei giochi che dal 1986 ad oggi rigioco annualmente. Senza mai stancarmi.

Ambientato su una nave spaziale con una vista dall'alto. La nave è formata da diverse stanze e livelli, ognuno popolato da droidi impazziti ed ostili. Il giocatore controlla l'unità 001 "Influence Device", che deve distruggere tutti gli altri droidi per riportare l'ordine.

Ogni droide, compreso quello controllato dal giocatore, è rappresentato da una sfera che gira intorno a un numero di tre cifre. Il numero corrisponde allo status del droide, più questo è alto più il droide sarà difficile da distruggere. Il numero del droide controllato dal giocatore è il più basso.

Oltre che distruggere i droidi nemici con il laser in dotazione, 001 deve evolversi assorbendo i poteri e lo status degli avversari. Questo avviene collegandosi con gli altri droidi e sconfiggendoli in un duello, una sorta di minigioco dove 001 deve appropriarsi di un maggiore numero di porte logiche e rubare l'identità dell'avversario. Dopo un duello, quale che ne sia l'esito, il droide eventualmente occupato da 001 viene distrutto.

Se 001 prevale su un altro droide, si appropria del numero e dei poteri del nemico appena sconfitto. Nel caso sia il droide del giocatore a perdere sarà esso a essere distrutto, oppure regredirà al livello base nel caso avesse già assimilato altri nemici. Ho sempre trovato questa modalità di gioco incredibile, da perdere la testa.

Cosa dire di più sul gioco. Graficamente non è importante, è funzionale. Il sonoro è caratterizzato dai suoni riprodotti dai droidi nella nave e da alcuni rumori di fondo.

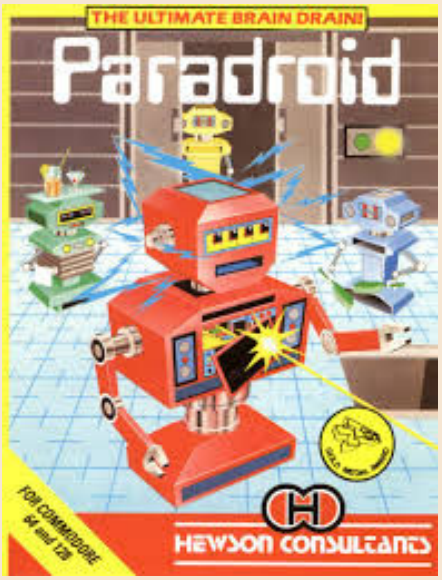
Il punto forte, la struttura di gameplay. L'approccio che si dovrà ottenere per conquistare gli avversari o per distruggerli è sempre differenti. L'Ai nemica è ben strutturata. Non saranno poche le volte che perderete "le staffe" durante la partita. Il gioco di per se non è difficile, ma punisce le disattenzioni e le "facilonerie".

Questo sistema di difficoltà non vi farà spegnere il gioco ma vi terrà incollati al vostro joystick per tanto tempo.. e con me lo fa ancora dopo tutti questi anni.

Hanno fatto anche una versione "heavy" per Amiga e Atari St anni dopo... ma questa per C64 è ancora un passo avanti.

Che dire di più... dovete giocarci... se lo avete già fatto... rifatelo!! E' un piccolo gioiello.

di **Carlo Nithaiah Del Mar Pirazzini**



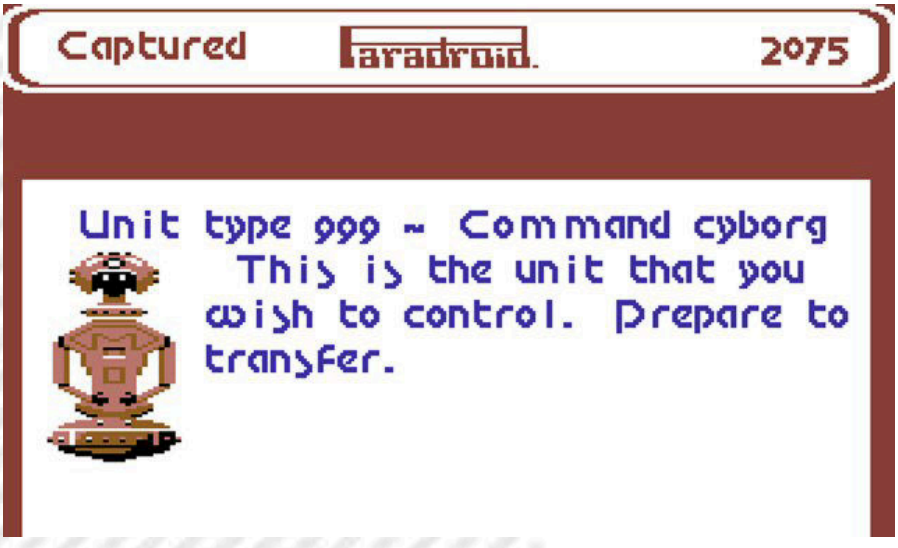
GIUDIZIO FINALE

» Giocabilità 95%

Difficile se si gioca senza pensare, ma è un ibrido tra shoot'em up e rompicapo e bisogna sapersi dosare.
Anche se si perde una vita, si procederà ad una nuova partita fino ad esplorare tutta quanta quella dannata nave.

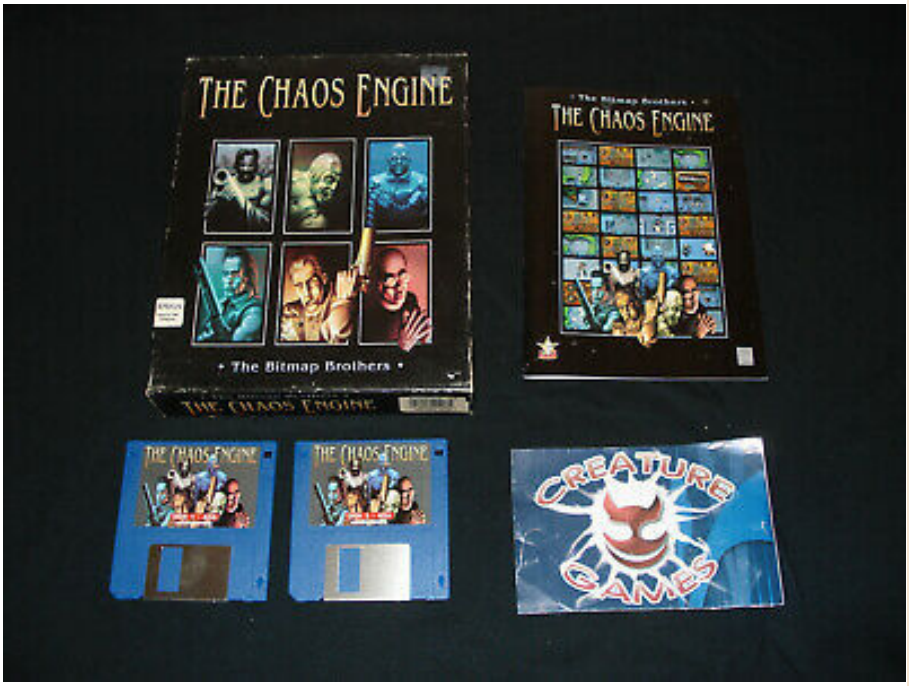
» Longevità ETERNO

La nave spaziale è "enorme" con tante sfida da fare, nemici da distruggere e esplorare ogni singolo punto.
Dal 1985 ad oggi è sempre una scoperta ed è piacevole da giocare... Eterno...
Semplicemente eterno.



THE CHAOS ENGINE

Piattaforma: Amiga, Atari ST, Amiga CD32, MS-DOS, RISC OS, Sega Mega Drive, Super Nintendo, Telefono cellulare
Data di pubblicazione: 1993
Pubblicazione: Renegade Software
Sviluppo: The Bitmal Brothers
Versione recensita: Amiga 1200



Nel 1993 eravamo in pieno periodo Steampunk nel mondo dei videogiochi, lo dimostrano le numerose uscite sia per i sistemi casalinghi.

Per Amiga saltò fuori questa serie e ci prese in mano in un mondo fatto di strana tecnologia, vapore e mix tra vittoriano e moderno.

Chaos Engine è uno sparatutto con elementi di RPG, pubblicato dall Renegade Software nel 1993 e realizzato dai Bitmap Brothers (un gruppo che ci faceva battere il cuore con i propri giochi).

Fu sviluppato con successo su Amiga nel 1993 in tutte le sue versioni e successivamente convertito per tutti i sistemi dell'epoca con discreto successo.

La versione che andiamo ad analizzare è proprio quella per Amiga in versione AGA, che girà sul mio fidato 1200.

Lo scenario del gioco è l'Inghilterra vittoriana totalmente diversa da quella che conosciamo. Un mondo dove la tecnologia non è basata sull'elettricità ma sul vapore. A causa di un errore di un brillante scienziato, il mondo è scosso dal Chaos Engine, apparecchiatura creata per sbalzi temporali che però sta causando fortissimi danni. Molto presto questi effetti malevoli appaiono ovunque e mostri di qualsiasi genere minacciano la popolazione. L'intero Regno Unito viene tagliato dal resto del mondo, la famiglia reale viene evacuata oltre oceano e molta della popolazione riesce a fuggire dalle isole maladette. Questo scenario apocalittico, richiama in Inghilterra bande di avventurieri e mercenari, decisi a cercare di sistemare la faccenda, dietro compenso Reale.

Saremo noi ad impersonare questi mercenari e questi sono sei, ognuno con caratteristiche differenti in armamento ed equipaggiamento.

Avremo due personaggi "HEAVY", armati di micidiali cannoni a grosso impulso come il Marinaio e il Teppista, più lenti e impacciati ma con maggior resistenza e potenza. Poi le classi tuttofare come il Brigante e il Macellaio con statistiche e armamenti nella media e infine le classi "VELOCI" con armamento più leggero e con equipaggiamento curativo come il Gentleman e il Prete (il mio preferito).

Il gioco consiste nel farsi strada attraverso livelli con visuale isometrica blastando qualsiasi cosa vi venga contro e risolvendo semplici enigmi (solitamente per reperire le chiavi oppure alcuni passaggi segreti).

Il gioco non sarà mai in solitario. Che si giochi singolarmente o in coppia con un amico, sullo schermo i personaggi saranno sempre due.

Nella modalità singola il compagno d'avventura sarà gestito dalla AI e non si limiterà ad essere un semplice "POD", ma vi sarà di aiuto in alcuni momenti di "CHAOS" durante la partita.

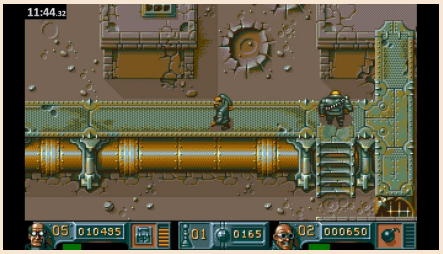
Alla fine del livello, attraverso i soldi recuperati e gli oggetti sparsi per i livelli sarà possibile fare l'upgrade del vostro personaggio e del suo compagno, aumentando le statistiche al meglio per poter proseguire con maggior sicurezza.

Giocabile da singolo e molto longevo in coppia, il gioco presenta una pulizia grafica notevole per il periodo. Con il classico stile pulito e "metallico" dei BITMAP BROTHERS che rende gradevole l'avventura.

Piuttosto difficile da giocare con il classico joystick, consiglio vivamente un buon pad o un arcade stick ad otto direzioni per evitare continue imprecazioni contro il divino.

A distanza di tutti questi anni THE CHAOS ENGINE è ancora un ottimo prodotto, invecchiato bene.

di **Carlo Nithaiah Del Mar Pirazzini**



GIUDIZIO FINALE

» Giocabilità 90%

Modalità in doppio molto divertente (anche se si rischia la lite con l'amico), interessante la modalità singola con l'AI di supporto. Ben strutturati gli enigmi nei livelli e in crescendo la difficoltà. Un po legnosi i controlli se non disponete di un joystick funzionale a 8 direzioni o un pad.

» Longevità 85%

La modalità in doppio meriterebbe di più, in singolo un po' meno. Alcuni dubbi sulla lunghezza di alcuni stage (eterni) e di altri (durano due secondi).

DRAGON'S LAIR

Anno: 1986
Sviluppatore: Software Projects Ltd.
Piattaforma: C64
Genere: Azione

In questo numero natalizio ho voluto occuparmi non del solito gioco dimenticato e sottovalutato, bensì di un gioco che tutti conoscono, anche i non videogiocatori e che per me è stato importante non solo a livello di divertimento e di passione ma anche perché è stato il primo gioco che ho messo nel datassete appena ho collegato il Commodore 64 la mattina del Natale 1987.

Faceva parte di dieci cassette che il negoziante mi aveva regalato come bundle di acquisto e tra loro c'era anche il secondo capitolo, non quello vero e proprio, bensì la seconda parte chiamata Dragon's Lair Part II: Escape from Singe's Castle di cui parlerò sempre in questo articolo (non è una recensione di due giochi in uno, ma come ben sappiamo all'epoca alcune versioni furono suddivise in due cassete e/o dischi per questioni di memoria). Il caricamento inziò molto bene con una stupenda musica e l'imponente castello... ma una volta finito mi resi conto di quanto il gioco fosse diverso dalla versione arcade alla quale ero abituato, non solo a livello grafico, ma anche di giocabilità. Inoltre alcuni tratti avevano una difficoltà estrema.

Il gioco comincia con il livello della piattaforma che cade e presenta Dirk liberodi muoversi e non legato alle sequenze di mosse esatte come avverrà in alcuni livelli successivi e nella versione arcade ed in Singe's Castle. Avendoli finiti entrambi posso consigliarvi di studiare bene ciascun livello imparando le mosse e i movimenti dei nemici, tanto non lo abbandonerete facilmente questo capolavoro nonostante le parecchie critiche mosse.

Il primo livello di Singes castle è invece quello della barca che evita scogli e mulinelli, uno dei livelli più facili della versione arcade, in più nella versione home computer sono stati aggiunti segnali indicatori per facilitare le cose anche se a mio avviso la barchetta un pare po' improvvisata e va parecchio veloce. Anche nel secondo capitolo vale tutto lo stesso discorso del primo: imparare a memoria i livelli.

Se qualcuno di voi non lo avesse mai giocato prima d'ora oppure volesse rispolverarlo per riprenderlo da dove lo aveva lasciato, il web offre un infinità di soluzioni anti esasperazione, compresi i video cheats con le mosse giuste al momento giusto. Nonostante all'epoca non sia mai riuscito a superare il secondo livello, l'ho amato come nella versione arcade e quando l'ho ripreso ho giocato a qualsiasi ora del giorno libera per finalmente trionfare dopo aver sconfitto il drago Singe (primo capitolo) ed essere scappato dal castello inseguito dal re lucertola che vuole fracassarci la testa (secondo capitolo). Con questo articolo voglio auugurare buone feste a tutti voi lettori e alla redazione: divertitevi con i retrogame e se vi capiterà, gustatevi questo meraviglioso gioco con la sua coinvolgente presentazione!

di **Daniele Brahimi**

Immagine evocativa

GIUDIZIO FINALE

» Giocabilità 70%
Non difficile familiarizzare con i comandi...

» Longevità 85%
Ore ed ore incollati allo schermo finchè non ci prenderete gusto!

Imparare le mosse a memoria...

Giappone 7^puntata: Il Mugen PuchiPuchi

di Michele Ugolini

Pikk, pokk, pffff, pfì, pokk, pokk, pikk, boing, pikk, pokk.
Purtroppo, cari lettori, non posso iniziare l'articolo senza una attenta analisi delle caratteristiche principali del gioco che sto per descrivere ed incredibilmente, le ho appena descritte, credetemi.
Quindi continuiamo, pfff, pokk, pikk, pfì!
La popolazione giapponese
(vi avevo già avvertiti abbondantemente)
rende particolare tutto ciò che apparentemente sembrerebbe normale.
Utilizza delle modalità di approccio realmente insolite ed originali.
Complice la loro propensione ad analizzare il mondo in maniera filosoficamente opposta alla nostra astrazione culturale.
Complice, anche e soprattutto, una fervida immaginazione collaudata da un isolamento geografico nel periodo medioevale, seguito da una incommensurabile esplosione tecnologica post bellica, poi riversata su tutto l'Occidente.
Oggi vi parlerò di un gioco, poi diventato anche videogioco, molto particolare, appartenente alla categoria "scaccia pensieri" che, nella versione originale, è diventato abbastanza raro da reperire.

Identikit del gioco:
Copyright: Bandai
Produttore: Asovision
Nome: Mugen Puchipuchi
(cfr. figura 1)
Titolo per Nintendo Wii
(cfr. figura 2) : Ouchi de Mugen Puchi Puchi Wii
Traduzione semplificata del nome:
Pluriball infinito,
anche detto Pucipuci
In parole semplici: Infinite bollicine di plastica degli imballaggi
Data di rilascio: 22 Settembre 2007
Colorazioni: 7 colori
Suoni prodotti: pikk, dingdong, bau, boing, prrrr, etc..
Categoria: Scacciapensieri

Cos'è, insomma, questo gioco?
Come è nata l'idea di questo gioco?
Perchè è stato creato questo gioco? Quale aggettivo di immediata comprensione potrei citare per spiegarvi questo gioco, cari lettori?
No, non riesco neppure io a fornire un aggettivo.
Le teorie che potrebbero spiegare la nascita di questo scaccia pensieri sono numerose.
Si narra nella notte dei tempi, quando ormai non c'erano più le mezze stagioni e quando dopotutto il domani sarebbe stato un altro giorno... che in Giappone si lavorava alacremente!
Fin qui nessuna novità.

Esatto, nessuna novità, il laborioso popolo giapponese è sempre stato dedito al lavoro ed ancor più al perfezionismo lavorativo.
Immaginiamo il loro boom economico: le industrie si stanno moltiplicando a ritmi vertiginosi e l'export verso i continenti attigui è lanciato al galoppo.
In quel preciso momento nasceva però un problema : come trasportare i prodigi elettronici via nave e via aereo senza danni, vista l'esistenza di tantissimi km che separano le isole dello stato giapponese, dal resto del mondo?
Non era un problema di facile soluzione.
L'imballaggio giapponese e l'origami si sono uniti in matrimonio ed è nato un miracolo straordinario riguardo la progettazione delle scatole!
Ricordate quando avete aperto l'ultimo prodotto Sony o Panasonic, etc? Gli spazi progettati per attutire e scaricare le pressioni e le vibrazioni esercitate durante il trasporto rasentano la maniacale filosofia di vita della popolazione nipponica.
Le scatole di cartone ondulato e la progettazione di tali incredibili oggetti assemblabili in un incastro poliedrico hanno realmente "fatto scuola" alle attuali ditte come Ikea ed Amazon. Oggi il sapere di questa progettazione degli imballaggi è pressoché ubiquitario.
Però negli anni 70, 80, l'incastro perfetto nasceva là, dove turbolenze aeree, moti ondosi impressionanti, cicloni, tifoni e Godzilla vari, tenevano sulle spine i produttori di tecnologia giapponesi, i quali pregavano tutti i loro Santi, sperando di far arrivare la merce sana e salva in tutto il mondo.
Preghiere, ingegneria, imballaggi e pluriball hanno portato alla vittoria finale.
Il Giappone fortunatamente non ha invaso il mondo tramite la guerra, bensì attraverso qualcosa per cui ammirarli: l'inventiva, la perfezione ed infiniti gioielli meccanici ed elettronici.
Il pluriball è stato inventato nel 1957 da due ingegneri statunitensi. Nonostante ciò in Giappone la produzione di queste bollicine si è sviluppata attraverso proporzioni straordinarie.
I turni di lavoro e lo stress lavorativo nipponici sono sempre stati elementi presenti, dai tempi della loro rivoluzione industriale post bellica. In Giappone, l'appartenenza del singolo ad un meccanismo sincronizzato collettivo era ormai diventato status obbligatorio, sia sociale, che morale. Quindi per forza di cose, ben presto, stress e pluriball... diventarono segretamente amanti.
Immaginiamo di vestire la divisa di un operaio presso una ditta di pluriball giapponese. Turni massacranti, controllo maniacale del prodotto, orari di distribuzione ai vari fattorini precisi al secondo,

supervisori severissimi che girano in continuazione e ci fanno sentire il loro fiato sul collo...
Poi improvvisamente il supervisore si allontana, oppure abbiamo terminato di controllare la macchina che ha prodotto la bobina, oppure addirittura siamo in pausa pranzo, dopo una giornata stressante, siamo soli, davanti a questa infinita quantità di bollicine e nessuno ci sta guardando...
Io, personalmente parlando, senza alcun rimorso nel proseguire, perderei volentieri ogni dignità pur di poter scoppiettare una intera bobina di pluriball!
Ecco che però nasce un problema sostanzioso: le bobine devono essere perfette e i futuri acquirenti controlleranno con i propri occhi la qualità di queste bobine.
Così fu: qualcuno scoprì il misfatto.
Le numerose ditte di pluriball ricevettero altrettante numerose lamentele dai loro clienti poiché, misteriosamente, tantissime bobine riportavano ingenti danni: spesso e volentieri le bollicine erano sgonfie, forse scoppiate o peggio ancora fatte scoppiare.
Un topo cattivo dentro la ditta avrà rosicchiato le bollicine? Un'anima maligna che solo i Ghostbusters avrebbero potuto acchiappare avrà tolto la pressione da ciascuna bollicina, una per una?
Vista la frequenza dei terremoti in Giappone, ci saranno stati problemi

Figura 1

di urti o vibrazioni durante il trasporto di tali bobine?
I nastri trasportatori avranno forse avuto dei problemi meccanici che hanno portato allo scoppio di queste bollicine?
La verità era molto più elementare: stress lavorativo e pluriball erano amanti segreti!
I titolari delle ditte cercarono di porre rimedio aumentando i supervisori, ma ahimè, non ottennero risultati soddisfacenti, anzi, andava sempre peggio.
I titolari allora pensarono di far indossare un guanto metallico che non potesse portare a fine corsa l'atto del pollice opponibile, ma ahimè i dipendenti segretamente forse facevano scoppiare le bollicine tra i denti, o forse tra le dita dei piedi, chissà, nessuno potrà mai saperlo:
la dignità spesso muore in preda ad atroci ed oscure sofferenze disposte dall'anima! Far amputare le dita dei dipendenti non era fattibile: i sindacati avrebbero "tuonato fuoco e fiamme" ed oltretutto, i poveri dipendenti amputati avrebbero rallentato considerevolmente il lavoro!

Figura 2

Da questo stallo nacque la geniale idea di creare e regalare ad ogni dipendente il Pucipuci!
Per Natale ogni membro della famiglia di ciascun dipendente avrebbe ricevuto il proprio Pucipuci. Così gli operai stressati potevano massacrare questo gioco e salvare le bobine di pluriball.
Da quel giorno gli amanti segreti diventarono, serenamente, buoni amici.
Chissà se realmente la storia è veritiera, forse potrebbe esserci un fondo di verità.

Tecnicamente parlando, questo Pucipuci è composto da una scatolina di plastica semi trasparente che contiene
un piccolo apparato elettronico con alcuni tastini, è inoltre dotato di un piccolissimo speaker nel retro, il

Figura 3

tutto alimentato da una batteria a bottone.
Sulla superficie della scatolina ci sono esattamente otto tastini che, se premuti singolarmente, simulano la consistenza di scoppio delle bollicine del pluriball e allo stesso tempo comandano lo speaker, di emettere un suono che solitamente corrisponde a "pikk".
Ogni cento "pikk" viene emesso casualmente un suono particolare: il campanello "ding dong", la molla "boing", il cane "bau" etc...
Se non ricordo male ci dovrebbe essere un bonus, cioè un suono aggiuntivo particolare, in ciascun Pucipuci.
Ricordiamo infatti che la tecnica dell'omaggio esclusivo è una metodica tanto amata dai giapponesi poichè alimenta notevolmente le oscure trame nipponiche del collezionismo.
La scatolina appena descritta appartiene al Pucipuci classico.
Ovviamente esistono anche le versioni limited edition.
Queste emettono suoni particolari ed alcune varianti possiedono addirittura l'audio di alcune voci femminili specifiche , inerenti ad alcuni personaggi famosi dei loro media.
Quest'ultimo è un fenomeno che

Figura 4

richiama la logica delle "Idol", ma preferisco non approfondire questa particolarità, poiché è un fenomeno profondamente intimo della loro cultura moderna: non possiedo alcun attrezzo del mestiere per addentrarmi né potervi spiegare obiettivamente tale peculiarità tutta loro.
Ci sono numerose varianti riguardo le colorazioni del Pucipuci.
Ci sono anche versioni che, allorquando i tasti non vengono pigiati per più di 15 secondi, il gioco emette un suono di richiamo insistente al fine di essere nuovamente utilizzato!
Nel Web troverete spesso la dicitura "Edamame Puti Puti", poichè in alcune versioni recenti del gioco, la sensazione di pressione sul tasto, è simile alla pressione che viene esercitata sul baccello dell'edamame (un delizioso legume giapponese simile al fagiolo frequentemente apprezzato in Italia come aperitivo nonchè in numerose altre ricette orientali) per far uscire il fagiolino sintetico di edamame dall'interno del Puti Puti a forma, appunto, di edamame!
Non un edamame qualunque, un edamame con una faccia disegnata sopra.
Non una faccia qualunque, ovvio, una faccia da collezione!
(cfr. figura 3)

Quante varianti esistono del Pucipuci?
Tantissime, alcune esclusive e limited edition. Un'orda di ossessivi compulsivi giapponesi ha decretato il sostentamento delle ditte che li producono.
Vogliamo approfondire il discorso del Peri Peri (cfr. figura 4)?
No, per questa volta vi risparmio.
Se per caso, invece, state scalpitando per saperne di più sul Pucipuci e magari siete anche curiosi di ascoltare il Jingle dedicato a questo gioco,
eccovi accontentati:

https://en.wikipedia.org/wiki/Mugen_Puchipuchi

http://b.bngi-channel.jp/putiputi/

https://www.animenewsnetwork.com/news/2008-02-07/bandais-bubble-wrap-popping-keychain-now-with-moe

https://www.asovision.com/periperi/cp.html

https://www.asovision.com/edamame/

https://youtu.be/hoywfUJNU0c

Attenzione ad "Adobe Flash Player", è ostico con Chrome: alcuni link funzioneranno meglio con Explorer.

Eccoci arrivati alla fine dell'articolo. Cari lettori, vi avevo promesso una nippo follia: effettivamente questo scacciapensieri appartiene a tale categoria.

Nel prossimo viaggio in terra nipponica, fatene incetta, prendete soprattutto quelli più fantasiosi oppure "limited edition", saranno un simpatico regalo per il fortunato "scoppiettatore seriale italiano"!

Mentre faccio pikk, pikk, boing, al mio Pucipuci, approfitto anche per fare gli Auguri di Sante Feste a tutti voi, un abbraccio, pikk!

Firenze Vintage Bit 2019 - Testimonianze

Leonardo Vettori, Massimo Belardi, Leonardo Miliani, Fabrizio Corpetti, Luca Carraffiello.

LEONARDO VETTORI – Vicepresidente dell'Associazione Firenze Vintage Bit Onlus.

Contrariamente a quanto si potrebbe pensare contribuire all'organizzazione di un evento di retrocomputer non ha niente a che vedere con il retrocomputer.
I compiti a cui un buon vicepresidente è tenuto ad assolvere sono i seguenti:
1) Indicare dove è posizionata la macchina del caffè di Robert Swiderski e verificare che ognuno paghi il suo caffè, fino a quando il Cusani per errore non mette l'acqua al posto del caffè.
2) Portare gli espositori alla propria postazione e farli sentire a casa propria.
3) Urlare in continuazione che la conferenza del tal dei tali sta per iniziare fino a perdere la voce.
4) Fare un sorriso a tutti e cercare cavi, cavetti, cavettini per gli espositori.
5) Spiegare ai curiosi perchè lo facciamo.
6) Dare supporto morale al Gori durante la conferenza e premere il pulsante al tempo giusto per far scorrere le slide.
7) Tenersi informato sulle previsioni meteo e rassicurare espositori e visitatori che, nonostante la settimana prima Lastra a Signa si fosse allagata, il freddo e la pioggia non sarebbero arrivati.
8) Ultimo, ma più importante di tutti, spiegare dove si trova il bagno pubblico.
Alla fine di tutto questo lavoro è bello leggere le testimonianze degli amici che hanno reso speciale il Firenze Vintage Bit 2019.

MASSIMO BELARDI del MUSEO del CALCOLO DAGOMARI

Da cinque anni ho la fortuna di partecipare al Firenze Vintage Bit.
Doppia fortuna per un doppio ruolo, come associato del FVB e come espositore, tramite il Museo del Calcolatore 'Laura Tellini'.
Ogni anno scopro nuovi mondi.
Nell'ultima edizione, ho avuto modo di approfondire la conoscenza del museo delle 'TexasInstruments', ed ho conosciuto Leonardo Miliani.
Leonardo è un bel tipo.
Ha un progetto, creare un computer da zero, from scratch come dicono gli americans. E' un ideasogno seria, educativa, istruttiva, da portare nelle scuole.
Ai tempi del recupero della nostra storia informatica, da Faggin un poi, l'esperienza di Leonardo è una pietra miliare del cos'era prima e di cosa potrà essere domani.

L'affluenza è stata alta. Noi del museo siamo stati accolti da domande e tante curiosità; specie i bambini e relativi genitori, i nostri interlocutori principi, il nostro pubblico ideale.
Talmente tante sono state le persone, che diversamente dagli altri anni, non abbiamo avuto modo di seguire le conferenze della domenica. E siamo contenti così...

Le nostre vecchie signore hanno concluso un altro giro di danza. I nostri basici sistemi di calcolo, hanno ancora stupito tutti. Chissà e non mi pare vero, l'anno prossimo sarà il tempo per crescere ancora, per sfruttare qualche presa di corrente, chissà, per una bella Divisumma 14?

LEONARDO MILIANI – Inventore dell'LM80C

Sono da sempre appassionato dei computer ad 8 bit e qualche tempo fa mi è venuta la pazza idea di

costruirne uno partendo da zero!
I motivi che mi hanno spinto a fare ciò sono presto detti: si poteva replicare quello che fece a suo tempo Steve Wozniak, quando progettò il primo computer Apple? Una sfida personale che, per ora, penso di aver vinto: sulla mia scrivania c'è infatti un piccolo computer assemblato su basette millefori tenuto insieme da mille cavi che però funziona: lo Zilog Z80 è il cuore di questa macchina, affiancata da memorie RAM e ROM, nonché da un chip grafico ed un chip audio che, collegato ad un computer, mi fanno rivivere quelle emozioni che da ragazzino si provavano quando si tirava fuori il "Commodore" e lo si collegava al televisore di casa.

Ero così orgoglioso di questa macchina che mi sono unito al gruppo Facebook del Firenze Vintage Bit per presentarla ad altri appassionati. Ed alla fine ho attirato l'attenzione di qualcuno tanto che mi hanno concesso uno spazio espositivo ma, soprattutto, la possibilità di un intervento all'evento del 24 novembre 2019 in cui parlare agli altri della mia creazione. Ma, soprattutto, ho avuto vero orgoglio quando durante la mostra diverse persone si sono presentate dicendomi che erano venute apposta per vedere il mio computer LM80C e chiedevano come era assemblato e di cosa era capace. Alla fine della giornata la voce rimasta era poca ma la soddisfazione personale era molta.

Oltre a questo ho conosciuto un sacco di persone fantastiche unite dalla comune passione dei "vecchi sistemi": una giornata fantastica. Un'esperienza che consiglio vivamente a chi si è fatto sfuggire questo evento di provare quanto prima.

FABRIZIO CORPETTI – L'esperto di Texas Instrument.

Vorrei dare anch'io un piccolo contributo al primo evento a cui ho partecipato in Toscana il FVB. Si sente un calore di famiglia, in un luogo ricco di storia. Un Grazie per questo a tutti i protagonisti. Si conoscono nuovi Amici, si incontrano altri che già conosci, si condividono le esperienze di un hobby che per me deve essere divulgazione, conoscenza sopratutto verso i giovani. Far vedere da dove siamo partiti dagli anni 70, fino ai giorni nostri, con tutti oggetti della Nostra Storia.....

LUCA CARRAFFIELLO , esperto di macchine commodore 264.

L'occhio del "non addetto ai lavori" non è abbastanza allenato per riconoscere i diversi campi d'interesse specifici, che si usa genericamente presentare a grandi linee sotto la voce comune detta "retrocomputing" (o "archeologia informatica", o "retrocoding", "retrogaming", chissà cos'altro). Realisticamente, la comunità mondiale degli appassionati d'informatica d'antan è suddivisa per obbiettivi: collezionisti dello hardware d'epoca, didatti del calcolo computazionale, videogiocatori incalliti, archeologi alla ricerca del software perduto, programmatori della sfarzosa cosiddetta "scena demo", sviluppatori di giochi su computer vetusti...non si finirebbe più a volerli definire tutti! Per le singole sottocomunità, certo non mancano piccoli ritrovi qui e là, ovvio... Ma poi, c'è il Firenze Vintage Bit!

Personalmente, non ho mai incrociato in precedenza una realtà come l'Associazione Firenze Vintage Bit Onlus, crocevia nel quale confluiscono tutte, ma proprio tutte, le varie anime dell'universo Retrocomputing.
Tenendosi ben distanti dall'autoconfinarsi in uno specifico argomento, tra gli associati è nitidamente percettibile la voglia di scambio e condivisione delle singole esperienze d'ambito, che vanno poi tutte a confluire in un'unica, massiva, cultura di genere. Ed

anche nel 2019, i frutti di tale cultura sviluppati durante l'anno, sono stati esposti al pubblico nell'ambito della manifestazione omonima.

È proprio la folta audience, come sempre, l'anima della festa. I visitatori hanno potuto ritrovare i giochi dell'adolescenza o scoprirne di nuovi, hanno assistito a brevi talk tematici e presentazioni di macchine davvero esotiche, hanno rimesso le mani su computer conosciuti, testato altri provenienti da chissà quale oscuro meandro dell'evoluzione informatica, e tanto altro. Famiglie, gruppi di ragazzini, seriosi esperti del campo: tutti loro si sono immersi nel caleidoscopio di luci colori e chipmusic che li accoglieva nelle sale dello Spedale di Sant'Antonio, in un'esposizione da toccare, da giocare, non certo da guardare passivamente.

Il pubblico guarda noi, noi cosa vediamo nel pubblico? Io ci vedo l'affermazione culturale della "Macchina". Dal primo videogioco in quanto tale è passata una sessantina d'anni, ed una manciata di lustri ci dividono ormai dall'ingresso dei computer nelle nostre case. L'informatizzazione popolare è un dato di fatto, ed infila passato presente e futuro dei computer in un unico calderone dal quale tutti prendono qualcosa per farne qualcosa di più. E come spesso accade, basta osservare i bambini ed i ragazzini, intenti a giocare a tutto il giocabile al FVB 2019: non c'è il gioco vecchio o quello nuovo, c'è solo il gioco, la sfida, il divertimento, sia esso chiassoso e colorato che pacato e riflessivo.

La dimensione di questo carnevale a base di silicio quarantenne? Umana, una dimensione a misura d'uomo, l'impegno divertito di appassionati che rubano tempo al lavoro, alla famiglia, agli impegni, per regalarsi e regalare un tuffo nelle loro passioni, iniziate un tempo fa e giunte orgogliosamente intatte sino ad oggi. E qualcuno la famiglia ce la impegna in toto, dall'allestimento al cucinare!

Si poteva pensare che il passato sbiadisse col tempo. Invece il Firenze Vintage Bit si afferma ogni anno di più, ogni edizione più riccamente, ogni volta con più novità, dimostrandosi pronto a fare il suo ingresso in un nuovo decennio.

Arrivederci allo FVB 2020 quindi, ad maiora!

UN RETROCAPODANNO CON I BOTTI!!!

È quasi Natale...così cantavano Jovanotti e Luca Carboni nel 1993 e ogni anno, in questo periodo, mi ritornano come per magia in mente le parole di questa canzone.
A dire il vero per noi amanti di retrocomputing e retrogames dicembre è un mese ricco soprattutto dei classici mercatini natalizi, dove, se siamo fortunati, è possibile reperire anche qualche tesoro della nostra infanzia.
Meno numerosi i retroeventi, ma ce ne è uno davvero originale che non possiamo non segnalarvi e recensire, stiamo parlando del mitico **Capodanno Nerd Party** che si svolgerà il 31 dicembre, a partire dal primo pomeriggio, presso il fantastico locale America Graffiti nella città di Forlimpopoli, in provincia di Forlì e Cesena, in via emilia per Forlì, accanto al locale Cineflash Multiplex.

Capodanno Nerd Party è la festa di Capodanno che tutti gli appassionati e nostalgici stavano aspettando.
L'evento è aperto a nerd e cosplayers, ed è nata da un'idea di A.G. Cosplay & BHC Fiere del Fumetto e Animazione.
La festa si terrà in un locale dedicato e allestito a tema per un capodanno davvero unico.
Durante la serata il divertimento è assicurato con la lotteria gratuita nerd con premi a tema, photoset in un uno spazio unico, karaoke con le sigle dei cartoni animati, area videogames e retrogaming, giochi da tavolo, just dance, deejay set con in console STAB FLOW con musica dance anni 90 e rock.
L'evento è dedicato a chi vuole divertirsi davvero e si potrà partecipare soltanto tramite prenotazioni che saranno valutate dallo staff.
Molto ricca l'area giochi da tavolo aperta per tutta la notte. Ci si potrà così sfidare a giochi come Gloom, Bandido, Dungeon Wc, Exploding Kittens, Vudù, Funky Gallo, Pictomania, Bananagrams Indovina chi, Mercante in Fiera, Doubble, Risiko, Monopoly fallout, Harry potter e la pietra filosofale, Jenga, Uno, Domino, Reverse, Super Cluedo, Scarabeo Marco Polo, Pocahontas, Domino, Partini, Communication, Rummikub, e Il gioco dell'Oca.
Ma il punto di attrazione principale sarà ancora una volta l'area retrogames che chiuderà non prima delle 5:00 del mattino. Ecco solo alcuni dei titoli che sono stati scelti dagli organizzatori: Bobble Bubble, Crash CTR, Metal Slug, Street Fighter II, 1942, Final Fight, Super Puzzle Fighter II Turbo, Bomb Jack, Wonder Boy, The Simpson, Tetris, Sailor Moon, Golden Axe, Super Pang, Mortal Kombat, Ghouls'n Ghosts, Cadillacs and Dinosaurs.
Come ogni capodanno che si rispetti non mancherà il tradizionale cenone che, per accontentare davvero tutti i palati, comprenderà anche menù vegetariani, vegani e celiaci con piatti a loro dedicati.

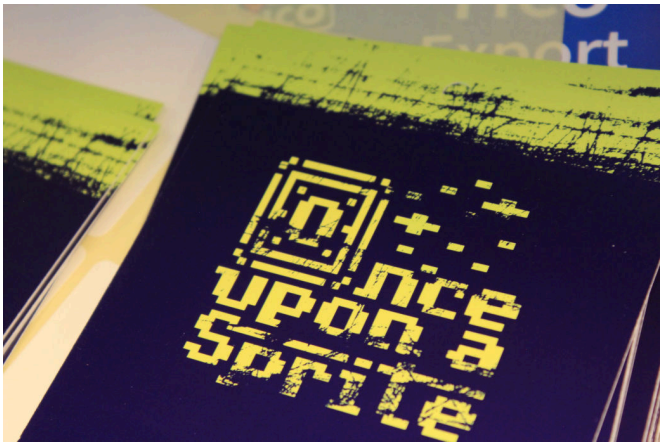
di Querino Ialongo

EVENTI DICEMBRE 2019

FORLIMPOPOLI (FC)
31 DICEMBRE

ONCE UPON A SPRITE - L'evento principe del retrocomputing in Italia

di David La Monaca (con la collaborazione di Andrea Ferlito)



Il retrocomputing per noi di RetroMagazine non è solo un hobby nostalgico. Chi ci segue da qualche numero avrà notato come nei nostri redazionali spesso e volentieri incoraggiamo tutti gli appassionati a rendere la propria esperienza pubblica e attiva, coinvolgendo altri fan ed esperti dei diversi sistemi a commentare, intervenire e a dare forma alle tantissime idee che possono scaturire solo dall'incontro di più persone. Condividere le proprie esperienze e presentarle alla comunità di riferimento costituisce un passo essenziale per rendere sempre vivace e attuale la scena del retrocomputing ovunque nel mondo.
E parlando di condivisione, non possiamo non menzionare l'evento annuale "Once Upon A Sprite" (di cui abbiamo già parlato in un breve resoconto nel n.ro 18 di RM) come esempio più alto di confronto, di incontro fra i più accreditati esperti (vecchi e nuovi) delle piattaforme a 8/16 bit di un tempo, per mantenere viva l'impronta storica che gli home computer hanno donato alle generazioni passate, oggi spesso professionisti della programmazione, del giornalismo specializzato, dell'IT in genere.
Ne parliamo grazie alla collaborazione di Andrea Ferlito, uno dei fondatori e promotori del meeting tutto italiano che si tiene ogni anno in autunno in una grande città e che raccoglie un copioso numero di "addetti ai lavori" oltre all'interesse marcato di pubblico giovane e meno giovane presso la sede dell'evento e sui forum e social network dedicati al retrocomputing. Senza tema di smentita, possiamo affermare che OUAS, con la maturità raggiunta, rappresenti il momento d'incontro del retrocomputing più curato e professionale che viene organizzato sul territorio nazionale, per la qualità dei contenuti, il profilo dei conferenzieri e degli ospiti, il tenore generale e l'interesse suscitato dalle sue tavole rotonde e dagli interventi tecnici.

LE ORIGINI

Once Upon A Sprite nasce dal fortunato incontro di due appassionati del settore della programmazione di videogiochi che provengono da percorsi professionali simili. A Roma, nel marzo 2017, Massimiliano Agostinelli e Roberto De Gregorio cercano di organizzare una piccola conferenza/raduno di programmatori di videogiochi italiani degli anni '80/'90. Decidono di chiamarla in maniera informale "Retrochiacchiere e dintorni".

Parallelamente e nello stesso periodo, Andrea Ferlito, mentre lavora alla preparazione di un talk che ha come argomento principale il retroprogramming per il C64 da presentare alla conferenza "Codemotion Roma", rivela al suo amico Andrea Pompili (sviluppatore di Catalypse su C64 per Genias, uscito nel 1992) l'idea di organizzare un evento tutto dedicato a quegli anni gloriosi e pionieristici di produzione dei videogame. Un meeting in cui si possa parlare di tutta la "filiera" del gaming, dall'idea iniziale, allo sviluppo del codice, dalla creazione della grafica e del commento sonoro, fino alla distribuzione (più o meno legale) dei titoli. Andrea Pompili è anche amico di Agostinelli ed è già coinvolto nell'altro evento, per cui invita il suo omonimo a partecipare a Retrochiacchiere. L'evento, sebbene di dimensioni limitate e quasi improvvisato, propone speaker e argomenti che riscuotono un grande interesse. Così, dopo questo primo esperimento, Massimiliano, Roberto ed Andrea decidono di lavorare insieme ad un nuovo evento mettendo a frutto e condividendo tutte le loro personali esperienze.
Massimiliano è stato un protagonista degli anni d'oro degli home computer e nella sua esperienza c'è un po' di tutto: programmatore, cracker, intro maker e persino un po' pirata. Roberto è invece il collezionista per eccellenza, raccoglie e cataloga macchine, giochi originali e memorabilia dei tempi che furono. Andrea, infine, avrebbe sempre voluto creare giochi negli anni '80, ma si è dedicato per vocazione alla consulenza informatica ed ha ripreso in mano l'assembly del 6510 in età avanzata e con un certo rammarico. Porta con sé l'esperienza pluriennale nell'organizzazione di conferenze, in qualità di CTO di Codemotion e curatore dei contenuti relativi al game development.
Il nome del meeting annuale "Once Upon A Sprite" proviene da un'idea di Raffaele Valensise (storico artist per Simulmondo, Genias e Team 17 – ha inoltre lavorato per diverso tempo nella distribuzione). Come data di riferimento viene scelto l'ultimo sabato di otto-

bre e come luogo la selezione ricade quasi automaticamente su Roma, comoda per i promotori dell'iniziativa e centrale rispetto alla platea di riferimento: l'intera penisola.

COS'E' OUAS?

Il focus principale dell'evento è da sempre la storia della programmazione di videogiochi su macchine anni '80 e '90, narrata al pubblico attraverso esperienze dirette, curiosità e aneddoti divertenti dei protagonisti di quegli anni. Si trattava di un mondo e di un settore di produzione agli albori, le informazioni (tecniche e non solo) circolavano con il contagocce, chi voleva emergere doveva stupire gli altri e per farlo doveva barcamenarsi tra le limitazioni intrinseche di quelle macchine e cercare di superarle. Per raggiungere quest'obiettivo, il programmatore doveva conoscere bene la macchina su cui stava scrivendo il proprio gioco, doveva avere nozioni di elettronica ed essere capace di pensare in modo laterale. Intorno al duro lavoro del programmatore che produceva un titolo, si muoveva poi una "filiera" fatta di software house, di distribuzione, di editoria, di cracking crew, demoscene e pirateria.

I giovani programmatori di allora sono oggi tra i migliori professionisti IT e lavorano in ambiti particolarmente delicati e complessi come ad esempio la cyber security, programmazione di firmware e hardware, software realtime, ecc. Tutto questo sistema, a suo modo virtuoso e in auge circa 30-35 anni fa, non esiste più e questo crea un gap di formazione importante. I giovani sviluppatori di oggi sono per lo più programmatori "dedicati" (ad una piattaforma di sviluppo, ad un linguaggio di alto livello) o specialisti certificati su un framework, posizionato molto lontano dall'hardware e dall'elettronica sottostante, che ormai consta di pochissime piattaforme/architetture costruite attorno a 2-3 CPU prodotte.

OUAS prova a raccontare le storie di quel passato soprattutto ai giovani sviluppatori di oggi, per incuriosirli e far capire loro un po' meglio da dove viene il lavoro che fanno. E la cosa sta funzionando: nelle varie edizioni dell'evento sin qui tenute, la fetta di pubblico più grande fa riferimento ad un range di età tra i 20 e i 30 anni. All'interno del meeting vengono a volte organizzati anche workshop maggiormente tecnici, dedicati a chi vuole approfondire la programmazione delle vecchie macchine.

Chi ha vissuto in prima persona il periodo che va dalla fine degli anni '70 fino al 2000, ha avuto la fortuna di veder nascere il fenomeno degli home computer e dei personal computer: un fenomeno che ha portato tanti giovani ad avvicinarsi all'informatica professionale. Ma spesso è stato proprio il videogioco il grimaldello che ha fatto atterrare il computer sulla scrivania dei ragazzi degli anni '80 e '90. C'è chi si è fermato al tentativo di battere i record utilizzando il suo joystick e c'è chi invece è voluto andare oltre lo schermo colorato e gli sprite, cominciando a guardare dietro le quinte del videogame e cercare di comprendere come avvenisse tutta quella magia che donava ore e ore di spensierato divertimento. Per le piattaforme di allora era obbligatorio imparare l'assembly (il BASIC era lo standard della programmazione di allora, ma troppo lento per creare produzioni videoludiche di un qualche interesse). Di qui la necessità di approfondire e di imparare le tecniche per gestire al meglio la CPU del computer, la sua memoria RAM ed i chip dedicati alla grafica ed al sonoro. Non esattamente una passeggiata, ma alla fine i programmatori esperti potevano veramente disporre di ogni singolo bit e ciclo di elaborazione hardware ed ottenere in questo modo risultati sorprendenti e inimmaginabili date le dotazioni spesso limitate delle macchine. Il background e l'esperienza complessiva che un programmatore poteva ricavarne rappresentava senza alcun dubbio un percorso formativo di prim'ordine per il futuro professionale nel settore dell'informatica professionale.

LE VARIE EDIZIONI

Dopo il primo Retrochiacchiere del 2017, tre edizioni successive si sono tenute, due delle quali a Roma (ottobre 2017 e 2018) e l'ultima, nello scorso ottobre 2019, per la prima volta a Milano. Ne abbiamo parlato brevemente nello scorso numero di RetroMagazine e più avanti approfondiremo alcuni talk che sono stati fra i protagonisti della giornata meneghina. La prima edizione fuori da Roma è stato anche una sorta di esperimento: incontrare la comunità di retro-appassionati dell'area attorno a Milano, che aveva mostrato grande interesse nel format dell'evento. Per l'occasione il team di OUAS ha chiesto ed ottenuto il supporto degli "autoctoni" Carlo Santagostino (storico redattore delle riviste ZZapp! e The Games Machine), di Francesco Sblendorio (programmatore e grande appassionato di retrocomputing) e Fabrizio Lodi (programmatore, giornalista e fondatore di Retrocampus). E l'esperimento è

riuscito perfettamente: sala piena tutto il giorno, molti speaker, interventi di ottimo livello e grande soddisfazione del pubblico

IL TEAM

Attorno a "Once Upon A Sprite" si è creato un gruppo di amici che ha vissuto attivamente l'epoca d'oro degli home e personal computer e che tuttora regala la propria esperienza e i suoi preziosi consigli ai tre organizzatori. Oltre ai già citati Andrea Pompili e Raffaele Valensise, ricordiamo Valerio Lupi (xOANINO di UCF), Antonio Mazzanti (Randall Flagg di Razor 1911), Giuliano Peritore (sviluppatore Amiga ed esperto di hardware) e Francesco Raimondo (fondatore di Yodas Crew).

OUAS 2019 – MILANO

L'agenda dell'edizione milanese di OUAS è stata suddivisa in tre blocchi ben distinti: il primo tutto incentrato sul ruolo dell'editoria videoludica italiana, il secondo relativo ad hacking, cracking e pirateria, ed il terzo maggiormente tecnico ha presentato al pubblico progetti personali concreti degli speaker a tema retrogaming. Per dare un'idea dei contenuti trattati in questa edizione di OUAS, vi riproponiamo una rapida scorsa del programma della manifestazione con qualche commento.

Ad apertura lavori, Andrea Ferlito ha dato il consueto benvenuto ad una sala praticamente gremita di pubblico e di addetti ai lavori. La scelta azzeccatissima della location, a due passi dalla Stazione Centrale, ha favorito l'afflusso di appassionati da ogni parte d'Italia.

Subito dopo è stata la volta di RetroMagazine con una breve presentazione del progetto editoriale nato dall'idea del nostro direttore Francesco Fiorentini. Lo spirito che ci anima, i risultati raggiunti e le idee progettuali per il futuro prossimo sono stati i punti cardine del talk di Cercamon/David La Monaca, che assieme alla fotografa d'occasione Vovo, ha documentato il resto dell'evento per RM.

Parlando sempre di editoria ma stavolta al passato, Carlo Santagostino ha introdotto e moderato il tema della discussione della Tavola Rotonda "L'editoria videoludica italiana, origini e sviluppo" cui hanno partecipato lo stesso Santagostino (Zzap!, TGM), Bonaventura Di Bello (Zzap!, via Skype), Alessandro. De Simone (Commodore Computer Club), Paolo Besser (Zzap!, TGM), Claudio Todeschini (TGM), Davide Corrado (Zzap, Bovabyte) e Fabrizio Lodi (Giornalista e Developer).

A seguire un interessante excursus sulla "Nascita del mercato italiano dei videogiochi" a cura di Carlo Santagostino. All'inizio degli anni '70 è nato e si è diffuso il mercato del videogioco. Questo nuovo media, sebbene già "inventato" in esperimenti accademici e ricerche industriali, ha iniziato a divenire popolare e soprattutto venduto ovunque come forma di intrattenimento. La storia a livello mondiale è ormai nota e codificata, ma cos'è successo in particolare in Italia, quali sono state le prime aziende a credere ed investire in questa novità? Alcune di queste stavano seguendo un percorso di sviluppo del tutto in linea con quanto stava succedendo in altri paesi europei e in USA. Non mancavano casi d'eccellenza e progetti di assoluta avanguardia portati avanti da aziende italiane, ma perché in Italia il videogioco non è diventato un generatore di business industriale come invece è successo in altri paesi? Le cause, come ha magistralmente dimostrato Carlo Santagostino, sono state molteplici ma la principale è stata la mancanza di una legge chiara sul copyright e sulla pubblicazione di software videoludico e non, che ha lasciato il campo ad una pirateria dilagante (molto più che in altri paesi dove pure era presente) che ha drenato e distratto capitali che avrebbero potuto finanziare la nascita di un'industria del videogioco prospera e volano di uno sviluppo ben più consistente dell'informatica personale nel nostro paese, con una conseguente crescita complessiva del digitale (nei diversi settori dall'educational.all'entertainment).

Andrea Pachetti ci ha poi portato all'interno della sua felice esperienza del Blog QuattroBit, fonte inesauribile di aneddoti, curiosità ed interviste coi protagonisti dei videogame anni '80. In oltre dieci anni di vita, Quattro Bit ha raccolto un gran numero di contenuti utili a ricostruire la storia dei primi videogiochi, dai trafiletti sui quotidiani agli approfondimenti sulle riviste d'epoca, fino a foto e video: tra tutti i contributi, i più apprezzati e letti sono stati le interviste realizzate con i protagonisti di questa prima "invasione". Da qui è nato un progetto più ambizioso, cioè contattare i programmatori e i produttori di videogiochi per Commodore 64 del 1982-85: scoprirne gli aneddoti e i segreti, per presentarli in un futuro volume cartaceo.

Nella seconda parte di OUAS 2019, ha preso la parola in questo mondo storicamente più maschile (non maschilista) una graditissima speaker femminile, Stefania

Calcagno, conosciuta nel mondo Amiga e dell'hacking in generale con l'interessante talk intitolato "Phreaking, Stamping, Hacking: Come una generazione ha cambiato le regole del gioco". Una generazione di giovani ha approfittato della tecnologia informatica e telematica durante la sua nascita e i primi anni del suo sviluppo, anche a causa di norme che non erano certamente al passo coi tempi. Non tutti, ad esempio, sanno che la Blue Box fu venduta da Jobs e costruita da Wozniak per aiutarli a finanziare l'Apple 1. Negli anni 70, 80 e 90 fu possibile rivoluzionare il modo di comunicare a grandi distanze ed a costo zero e gli sceners furono i protagonisti di quella rivoluzione. Stefania ha ripercorso per il pubblico presente le "scoperte" di tutti quegli anni, passando per milestones, ricordi personali e momenti rivelatori.

A metà fra retrocoding e la presentazione di un progetto tecnico pratico, l'intervento di Francesco Sblendorio: "Creare e mantenere una BBS per C64 nel 2019". Un modo divertente per rendere il più possibile attuale l'utilizzo di un Commodore 64 connesso a Internet, implementando facilmente servizi di news, messaggistica e altro attraverso lo sviluppo di un moderno framework Java. Di facile implementazione anche per chi è digiuno di BBS, modem e programmi terminale, Francesco ci ha stupiti portando servizi utili come news, messaggistica e-mal–like, feed RSS, download di software via XModem e semplici giochi online sui 64 Kbyte di un Commodore 64!

Antonio Mazzanti ha poi trascinato il pubblico nei ricordi e nelle divertenti avventure del suo alter ego di un tempo, quando con il nickname di Randall Flagg militava nei gruppi Razor 1911, Eclipse ed altri, tutti protagonisti della cracking scene mondiale. "Le cracking crew anni 90 tra mito e leggenda" il titolo del suo talk, infarcito di aneddoti, curiosità e storie mai rivelate prima! Un vero e proprio viaggio nella scena cracking PC degli anni '90. Finalmente oggi sappiamo grazie a chi abbiamo potuto giocare gratis agli adventure game "Monkey Island" e "Indiana Jones" su PC/VGA!

Tornando a parlare di Commodore 64, indiscusso protagonista negli anni '80, vero e proprio simbolo degli home computer in Italia, Silvio Savarino ha illustrato a tutti la nascita e l'evoluzione della scena C64, mostrando poi tramite la sua personale esperienza che questa è estremamente attiva anche dopo il Duemila. "Hokuto Force, la scena C64 nel nuovo millennio" il titolo dell'intervento di Silvio aka Overkiller, membro per l'appunto di Hokuto Force, uno dei gruppi più attivi e prolifici delle due ultime decadi. Con esempi pratici di come avviene al giorno d'oggi la produzione di un nuovo crack o di un nuovo videogame originale per C64, Silvio ci ha dimostrato come la scena di questo amatissimo 8-bit sia più viva che mai.

Nella terza parte di OUAS 2019 si è dato spazio ai progetti tecnici ispirati dal retrogaming e fra questi grandissimo successo (e non poteva essere diversamente) hanno riscosso Giuliano Peritore, Vittorio Signorelli e Maurizio Damiani Chersoni con il loro talk "Space ARDUIN.VADERS: Reverse ed emulazione di Space Invaders su Arduino Nano". 42 anni fa, l'anno del colore in TV, quando i personal computer disponibili erano l'Apple II, il TRS-80 e il CBM PET Tomohiro Nishikado viene incaricato di progettare un videogioco da bar. Nasce così una pietra miliare dei videogiochi, Space Invaders, primo vero gioco bitmap, con colonna sonora continua, che introduce il concetto di "vita" e la tabella High Score. L'hardware del gioco si basa su Intel 8080, chip audio Texas SN76477, 8 KByte di RAM, 8 Kbyte di ROM, nulla rispetto all'elettronica di oggi. I tre autori del progetto avevano allora fra i 16 ed i 18 anni e rimangono affascinati dal gioco e se ne ricordano molti anni dopo quando collaborano a diversi progetti digitali. In occasione di OUAS 2018 decidono di porsi una sfida: per OUAS 2019 devono far girare il byte code originale di Space Invaders su hardware molto semplice ed economico e con meno componenti a contorno possibile. Per la sfida viene scelto Arduino Nano 3.x con display Ilitek 9341 (costo totale di circa 7 dollari!). Si tratta di portare avanti un lavoro raffinatissimo necessario per emulare l'Intel 8080 e far entrare tutto nella RAM di Arduino e del suo display. Ma occorre anche emulare via software l'hardware dedicato al suono, il tutto per riuscire infine a far girare anche la ROM originale di test del gioco. Difficile condensare in poche righe tutti i dettagli tecnici e le difficoltà incontrate nel percorso, ma per chi desidera approfondire rimandiamo alle slide che verranno pubblicate sul sito www.x-nor.it. Una sfida intrigante che ha suscitato l'interesse di tanti amici del pubblico. Il loro desiderio di provare l'emulatore ha ripagato in pieno tutte le fatiche della sfida pienamente vinta dai tre autori.

A seguire Valerio Lupi ha illustrato il suo lavoro che consiste nello "Scrivere un emulatore di C64" con gli strumenti dei moderni OS di oggi. VC64-emu è il nome del progetto che Valerio ha portato avanti per qualche mese (iniziato a gennaio 2019, ma sviluppato quasi totalmente durante l'estate). Lavorare a questo progetto

gli ha permesso di capire cosa significa creare un emulatore di hardware partendo totalmente da zero, dopo che per anni si è sempre e solo occupato di software di sistema senza nessuna interazione con l'utente e lo schermo. L'obiettivo principale che si era prefisso, a causa della sua mania per Jeff Minter, era quello di fermarsi con lo sviluppo non appena fosse riuscito a far girare nell'emulatore il gioco "Gridrunner" di Llamasoft (per il quale basta soltanto emulare il text mode standard, in quanto la grafica è composta totalmente da caratteri ridefiniti). Alla fine è riuscito ad implementare anche gli sprite e gli altri modi grafici 'base' (text e bitmap, standard e multicolor) e rendere il tutto 'abbastanza' decente da essere presentato: al momento l'emulatore carica soltanto i .PRG, la compatibilità è molto variabile (molti giochi hanno bug grafici dovuti all'implementazione del VIC, che col senno di poi Valerio effettuerebbe diversamente), ma più o meno la maggioranza dei giochi parte. Chissà se un giorno ne riprenderà lo sviluppo... anche perché ora la sua sfida è quella di realizzare un emulatore di Atari 2600. Sicuramente un altro modo, molto creativo, di imparare l'architettura di queste splendide macchine anni '80. Ottimo lavoro, Valerio! (Potete trovare le slide ed i sorgenti di VC64-Emu su https://github.com/valerino/vc64-emu).

La giornata è terminata con la premiazione della coding challenge indetta da Retrocampus - sys64738, che consisteva nello sviluppo a tema di un videogame su piattaforme retro a 8/16bit. Con grande originalità il tema prescelto era "Milano" ed il vincitore è emerso nel nome di Paolo Cattaneo con il titolo "RossoNero", un'avventura poliziesca scritta per PC128 Olivetti Prodest, ambientata nella Milano degli anni Settanta, attraversata dal clima degli scontri in piazza, in cui il giocatore è chiamato a risolvere un misterioso caso di omicidio. Il gioco è per lo più improntato all'uso del testo ma presenta 10 schermate grafiche che appaiono mentre il giocatore riesce ad avanzare nella storia. L'interazione si divide in due fasi: nelle scene di ispezione dovrete "cliccare" in giro per trovare indizi utili a progredire nell'indagine, altrimenti vi sarà solo richiesto di selezionare delle opzioni dal menù che si presenterà di volta in volta. Più indizi scoprite, più possibilità avete di risolvere il caso. Ci sono diversi finali, ma solo uno è quello davvero buono. E' anche possibile rimanere bloccati senza possibilità di risolvere il caso, quindi se pensate di essere arrivati su un binario morto potrete sempre rinunciare e ricominciare da capo. Tutto sommato un gioco con una rispettabile trama ed un gameplay complicato quanto basta per attrarre la vostra attenzione. Ben realizzata la parte dedicata alla ricerca di indizi con le schermate 160x100x2 di buona fattura che aiutano il giocatore a immedesimarsi nella parte e a respirare il clima dell'ambientazione. Ottimo lavoro e complimenti al vincitore della sfida OUAS2019!

In foto nell'ordine: A. Ferlito, M. Agostinelli, R. De Gregorio, C. Santagostino, F. Sblendorio, A. De Simone, S. Calcagno, F. Lodi, A. Pachetti, A. Mazzanti, V. Lupi, G. Peritore.

Riferimenti

Pagina Facebook di OUAS:
https://www.facebook.com/onceuponasprite

Once Upon a Sprite 2017:
https://www.youtube.com/playlist?list=PLq2-o3pBTowf5O6f3f2Y5J_t6bpgmPLBu

Slide di Once Upon a Sprite 2017:
https://www.slideshare.net/Codemotion/clipboards/once-upon-a-sprite-2017

Slide di Once Upon a Sprite 2018:
https://www.slideshare.net/Codemotion/clipboards/once-upon-a-sprite-2018

Finalisti della Coding Challenge di OUAS 2019:
http://www.retromagazine.net/download/ouas2019-CC.zip

Cercamon e Vovo in missione per RetroMagazine

To be, or not to be, that is the question!

Essere o non essere... è una frase dell'Amleto di William Shakespeare che ho preso in prestito come titolo per questo articolo di chiusura dell'ultimo numero dell'anno di RetroMagazine.

Il dubbio che attanagliava Amleto e che era alla radice dell'indecisione che gli impediva di agire, è similmente ascrivibile anche a noi di RetroMagazine.
Ma qual è il dubbio che attanaglia noi di RM? Considerato il fatto che ho preso a prestito uno dei più famosi versi della lingua inglese, è proprio legato all'idioma di Albione.

Sempre più lettori stranieri ci chiedono a gran voce una versione inglese della nostra fanzine ed arrivati alle soglie del numero 20, dopo aver promesso di cercare una soluzione, credo proprio che dovremo smettere di pensare e cominciare ad agire.

Portare avanti due versioni di RM, una in italiano ed una in inglese, è francamente utopia... Abbandonare la lingua italiana per la più globale lingua inglese, ci sembra francamente uno sgarbo che l'italico idioma non merita affatto...
Come fare dunque per avere la botte piena e la moglie ubriaca? La tecnologia potrebbe venire in nostro aiuto. Abbiamo preso contatto con una realtà italiana che potrebbe aiutarci a velocizzare il processo di traduzione tramite l'utilizzo di algoritmi di Machine Translation.

I presupposti per fare qualcosa di buono e di innovativo ci sono tutti, non ci resta altro che cominciare a mettere a frutto questa collaborazione e toccare con mano i primi risultati.

Ovviamente occorrerà del tempo e delle energie dovranno essere dirottate verso questo fronte... Il che potrebbe tradursi in un rallentamento delle uscite di RM. Vedremo come all'interno della redazione saremo in grado di gestire questa nuova sfida.
Restate sintonizzati su questi canali per vedere come la cosa si evolverà. :-)

Prima di lasciarvi per godere del cenone di Capodanno mi preme ancora una volta ricordarvi i nostri canali di comunicazione ufficiali.

Per proposte di collaborazione scrivete pure all'indirizzo:
retromagazine.redazione@gmail.com
Il da fare non manca, credeteci sulla parola!

E' il momento di salutarvi, ma non temete, la vostra fanzine preferita tornerà presto (o quantomeno nel giro di un paio di mesi), come sempre.

Un saluto da parte mia e di tutta la redazione di RetroMagazine. Buona lettura e... che il nuovo anno possa portarvi tutte le soddisfazioni di cui il 2019 è stato avaro. Buon 2020!

Francesco Fiorentini

Disclaimers

RetroMagazine
Anno 3 - Numero 19

Direttore Responsabile
Francesco Fiorentini

Vice Direttore
Marco Pistorio

Dicembre 2019