Trabajo Práctico

Intérprete de Applesoft BASIC en Clojure

Las primeras computadoras Apple II salieron a la venta el 10 de junio de 1977, basadas en un microprocesador 6502 de MOS Technology funcionando a 1,023 MHz, con el lenguaje de programación *Integer BASIC* en ROM y una interfaz para grabadores de cassettes de audio, para que los usuarios pudieran guardar y recuperar programas y datos en cassettes. El controlador de video mostraba en la pantalla 24 líneas por 40 columnas de texto solamente en mayúsculas. El precio minorista original fue de 1298 dólares con 4 KiB de RAM y 2638 dólares con 48 KiB de RAM.

Aunque el *Integer BASIC* (una creación de Steve Wozniak) era adecuado para programar videojuegos, muchos clientes de Apple exigían una versión de BASIC que soportara números de punto flotante. Entonces Apple buscó a Microsoft y licenció una versión de BASIC denominada ***Applesoft***. La primera versión de *Applesoft* fue lanzada en 1977 solo en cinta de cassette. El *Applesoft II*, que fue lanzado en 1978 y pronto estuvo disponible en cassette, diskette y también en la ROM de la Apple II Plus y de varias de sus sucesoras, es la versión a que comúnmente hace referencia el término "Applesoft".

La familia de la Apple II estuvo formada por los modelos Apple II, Apple II Plus, Apple IIe, Apple IIc, Apple II GS y Apple IIc Plus, y estuvo en producción durante casi 17 años, desde 1977 hasta noviembre de 1993 (cuando se dejó de producir el modelo Apple IIe). En total se vendieron alrededor de 6 millones de unidades.

Si bien existen actualmente algunos emuladores de la Apple II (https://www.scullinsteel.com/apple2, https://porkrind.org/a2, etc.) escritos en JavaScript, su manejo es bastante complicado, y es por ello que, en este trabajo práctico, se pide desarrollar, en el lenguaje **Clojure**, no un emulador sino un **intérprete de Applesoft BASIC**, a fin de poder correr, en una ***Java Virtual Machine*** contemporánea, los siguientes programas escritos para aquellas computadoras de la familia de la Apple II:

Programa N° 1

STARS.BAS

```
10 INPUT "WHAT IS YOUR NAME? "; U$
20 PRINT "HELLO "; U$
30 INPUT "HOW MANY STARS DO YOU WANT? "; N
40 IF N < 1 THEN GOTO 90
50 S$ = "" : FOR I = 1 TO N
60 S$ = S$ + "*"
70 NEXT I
80 PRINT S$
90 INPUT "DO YOU WANT MORE STARS [Y/N]? "; A$
100 IF A$ = "N" THEN 130
110 IF A$ = "Y" GOTO 30
120 GOTO 90
130 PRINT "GOODBYE "; U$
140 END
```

Programa N° 2

NAME.BAS

```
10 INPUT "WHAT IS YOUR NAME? "; N$
20 L = LEN(N$): IF L = 0 THEN GOTO 10
30 PRINT "HELLO " N$: PRINT
40 FOR I = 1 TO L
50 PRINT MID$(N$, I)
60 NEXT I
70 FOR I = L TO 1 STEP -1
80 PRINT MID$(N$, I)
90 NEXT I
100 PRINT : PRINT "GOODBYE "; N$
```

Programa N° 3

SUM.BAS

```
30 LET N=1
40 LET S = S + N
50 PRINT N,S
60 LET N = N + 1
70 IF N <= 100 GOTO 40
80 PRINT : REM PRINT EMPTY LINE
90 PRINT "FINAL SUM: ";S
100 END
```

Programa N° 4

FIBO.BAS

```
05 PRINT "FIBONACCI NUMBERS"
10 LET M% = 5000 : LET C% = 0
20 LET X% = 0 : LET Y% = 1
30 IF X% > M% GOTO 100
40 PRINT "F(" C% ") = " X% : C% = C% + 1
50 X% = X% + Y%
60 IF Y% > M% GOTO 100
70 PRINT "F(" C% ") = " Y% : C% = C% + 1
80 Y% = X% + Y%
90 GOTO 30
100 END
```

Programa N° 5

GCD.BAS

```
10 REM FIND GREATEST COMMON DIVISOR (WITH USER SUPPLIED VALUES)
20 INPUT "ENTER A: ";A : INPUT "ENTER B: ";B
25 IF A<=0 OR B<=0 OR INT(A)<>A OR INT(B)<>B THEN GOTO 20 : REM RE-ENTER
30 IF A < B THEN C = A : A=B : B=C
40 PRINT A,B
50 LET C = A - INT(A/B)*B
60 LET A = B : B = C
70 IF B > 0 GOTO 40
80 PRINT "GCD IS ";A
90 END
```

Programa N° 6

PRIMES.BAS

```
10 INPUT "SEARCH PRIME NUMBERS UP TO: "; MAX
20 X = 1
30 LET P = .
40 IF X < 2 OR X <> INT(X) GOTO160
50 IF X=2 OR X = 3 OR X = 5THEN P=1 : GOTO 160
60 IF X/2 = INT(X/2) GOTO 160
70 IF X/3 = INT(X/3) GOTO 160
80 D = 5
90 Q = X/D : IF Q = INT(Q) GOTO 160
100 D = D + 2
110 IF D*D> X GOTO 150
120 Q = X/D : IF Q = INT(Q) THEN GOTO 160
130 D=D+4
140 IF D*D <= X GOTO 90
150 P = 1
160IFP=1THENPRINTX;" ";
170 X = X + 1
180 IF X < MAX THEN 30
190 PRINT
200 END
```

Programa N° 7

NATO.BAS

```
100 INPUT "ENTER A WORD: "; W$
110 T = LEN(W$): IF T = 0 THEN 100
120 PRINT "IT'S SPELLED AS FOLLOWS: ";
130 FOR I = 1 TO T
140 L = ASC(MID$(W$, I, 1)) - 64
150 IF L < 1 OR L > 26 THEN PRINT "??? "; : GOTO 190
160 FOR J = 1 TO L : READ S$ : NEXT J
170 PRINT S$; " ";
180 RESTORE
190 NEXT I
200 END
1000 DATA ALFA, BRAVO, CHARLIE, DELTA, ECHO, FOXTROT, GOLF, HOTEL
1010 DATA INDIA, JULIETT, KILO, LIMA, MIKE, NOVEMBER, OSCAR, PAPA
1020 DATA QUEBEC, ROMEO, SIERRA, TANGO, UNIFORM, VICTOR, WHISKEY, X-RAY
1030 DATA YANKEE: DATAZULU :REM EL ULTIMO VALOR ES ZULU
1040 DATA HOLA MUNDO
```

Programa N° 8

SINE.BAS

```
100 PRINT "X","SIN(X)"
110 PRINT "---","-------"
120 FOR I = 1 TO 19 : PRINT " "; : NEXT I
130 FOR A = 0 TO 8 * ATN(1) STEP 0.1
140 PRINT "*"
150 PRINT INT (A * 100) / 100, "  "; INT (SIN(A) * 100000) / 100000
160 FOR I = 1 TO 19 + SIN(A) * 20 : PRINT " "; : NEXT I,A
170 PRINT "*" : A = 8 * ATN(1)
180 PRINT INT (A * 100) / 100, "  "; INT (SIN(A) * 100000) / 100000
190 FOR I = 1 TO 19 : PRINT " "; : NEXT I : PRINT "*"
```

Programa N° 9

HEX.BAS

```
10 REM THIS PROGRAM PRINTS DECIMAL AND CORRESPONDING HEXADECIMAL NUMBERS.
15 REM IT SHOWS 16 NUMBERS PER PAGE. TO SHOW THE NEXT PAGE PRESS ANY KEY.
30 LET S = 0
40 REM NEXT SCREEN
50 PRINT "DECIMAL","HEXA"
60 PRINT "-------","----"
70 FOR I = S TO S + 15
80 LET A = I
90 GOSUB 200
100 PRINT I, HEX$
110 NEXT I
120 INPUT "PRESS RETURN OR TYPE 'QUIT' AND RETURN: "; V$ : REM WAIT FOR INPUT
130 IF V$ = "QUIT" GOTO 180
140 IF V$ <> "" THEN 120
150 S = S + 16
160 PRINT
170 GOTO 50
180 END
200 HEX$ = ""
210 LET B = A - INT (A/16) * 16 : REM B = A MOD 16
220 IF B < 10 THEN H$ = STR$(B)
230 IF B >= 10 AND B <= 15 THEN H$ = CHR$(65 + B - 10)
240 LET HEX$ = H$ + HEX$
250 LET A = (A - B) / 16
260 IF A > 0 THEN GOTO 210
270 RETURN
```

Programa N° 10

BIN-DEC.BAS

```
 80 PRINT "————————————————————————————————————————"
 90 PRINT " BIN -> DEC  /  DEC -> BIN  CONVERTER "
100 PRINT "————————————————————————————————————————"
110 PRINT
120 PRINT "1) BINARY -> DECIMAL"
130 PRINT "2) DECIMAL -> BINARY"
140 PRINT "3) QUIT"
150 INPUT "YOUR CHOICE: "; CH
160 IF CH < 0 THEN 180
170 ON CH GOTO 200, 340, 420
180 PRINT "WRONG CHOICE! RETRY..."
190 GOTO 110
200 LET P = 0 : LET S = 0
210 INPUT "ENTER BINARY NUMBER: ";N$
220 L = LEN (N$) : IF L=0 GOTO 310
230 FOR I=1 TO L
240 LET B$ = MID$(N$, L-I+1, 1)
250 IF B$ <> "0" AND B$ <> "1" GOTO 310
260 K = 0 : IF B$ = "1" THEN K = 1
270 IF K > 0 THEN S = S + 2 ^ P
280 LET P = P + 1
290 NEXT
300 GOTO 320
310 PRINT "ERROR, INVALID BINARY ENTERED" : GOTO 200
320 PRINT "AS DECIMAL: ";S
330 GOTO 110
340 X$ = "" :PRINT "ENTER AN INTEGER": INPUT "(GREATER OR EQUAL THAN ZERO): ";A
350 IF A < 0 OR A<>INT(A) GOTO 340
360 LET B = A - INT (A/2) * 2
370 LET X$ = STR$(B) + X$
380 LET A = (A - B) / 2
390 IF A > 0 GOTO 360
400 PRINT "AS BINARY: ";X$
410 GOTO 110
420 END
```

El intérprete a desarrollar en **Clojure** debe ofrecer los dos modos de ejecución de **Applesoft BASIC**: ejecución inmediata y ejecución diferida. Deberá estar basado en un REPL (*read-eval-print-loop*) que acepte, además de sentencias de Applesoft BASIC, dos comandos de **Apple DOS 3.3** (LOAD y SAVE). No será necesario utilizar espacios para separar los distintos símbolos del lenguaje. Soportará tres tipos de datos: números enteros, números de punto flotante y cadenas de caracteres.

Características de Applesoft BASIC y Apple DOS 3.3 a implementar en el intérprete

Comandos de Apple DOS 3.3

■ **Para la transferencia de archivos**

LOAD: carga un archivo.
SAVE: graba un archivo.

Sentencias de Applesoft BASIC

■ **Para la Entrada/Salida de datos**

INPUT: espera, con *prompt*, valor(es) por teclado.
PRINT: muestra valor(es) por pantalla.
?: lo mismo que PRINT.

■ **Para el uso de datos embebidos**

DATA: define datos embebidos en el código.
READ: lee variable(s) desde los datos embebidos.
REM: embebe un comentario en el código.
RESTORE: reinicia los datos embebidos.

■ **Para manipular el ambiente**

CLEAR: borra todas las variables.
LET/=: hace la asignación de un valor a una variable.
LIST: muestra el listado de sentencias del programa.
NEW: borra el programa y las variables.
RUN: ejecuta el programa.

■ **Para el control de flujo**

END: detiene la ejecución manteniendo el ambiente.
FOR/TO/NEXT/STEP: ciclo controlado por contador.
GOSUB/RETURN: va... (a una subrutina) y vuelve.
GOTO: va... (a determinada línea del programa).
IF/GOTO: si la condición es verdadera, va...
IF/THEN: si la condición es verdadera, ejecuta o va...
ON/GOSUB/RETURN: según un valor, va... y vuelve.
ON/GOTO: según un valor, va...

Sentencias del intérprete (ni en BASIC ni en DOS)

ENV: muestra el ambiente (*environment*).
EXIT: sale del intérprete y vuelve al REPL de Clojure.

Funciones de Applesoft BASIC

■ **Numéricas**

ATN: retorna la arcotangente de un número.
INT: retorna la parte entera de un número.
SIN: retorna el seno de un ángulo dado en radianes.

■ **Para cadenas de caracteres**

LEN: retorna la longitud de una cadena.
MID$: retorna una subcadena.

■ **Para la conversión de tipos**

ASC: retorna el cód. ASCII de la inicial de una cadena.
CHR$: retorna el carácter de un código ASCII dado.
STR$: retorna un número convertido en cadena.

Operadores de Applesoft BASIC

■ **Aritméticos**

+: calcula la suma de dos números.
-: calcula la resta de dos números.
***:** calcula la multiplicación de dos números.
/: calcula la división de dos números.
^: calcula la potencia de dos números.

■ **Para cadenas de caracteres**

+: realiza la concatenación de dos cadenas.

■ **Relacionales**

=: determina si dos valores son iguales.
<>: determina si dos valores son distintos.
<: determina si un valor es menor que otro.
<=: determina si un valor es menor o igual que otro.
>: determina si un valor es mayor que otro.
>=: determina si un valor es mayor o igual que otro.

■ **Lógicos**

AND: determina si ambos valores son verdaderos.
OR: determina si alguno de los valores es verdadero.

Para aprobar la cursada se deberá entregar el código fuente del intérprete completo.
Al rendir la evaluación final de la materia, se lo deberá modificar.