



**Universidad Nacional Autónoma de México  
Facultad de Estudios Superiores Aragón**



Ingeniería en Computación  
**COMPILADORES**  
Grupo: 2608


**Profesor:** Pérez Medel Marcelo

## **TAREA 5**

### **Máquinas virtuales**

**Alumna:** Cruz Cervantes Guadalupe Sugeily

**a) Revise el programa y explique brevemente cómo funciona**

 vm.py > ...

```

1 #Tenemos 2 listas, 2 contadores, 1 diccionario y nuestro archivo que se va a leer.
2 inst = []
3 param = []
4 pc = 0
5 ac = 0
6 memDatos = {}
7 archivo = open("programa2.txt", "r")
8
9 # En la primera parte del codigo se encarga de separar las instrucciones y los parametros con split(espacios).
10 # Para despues colocar las instrucciones en la lista inst y los parametros en la lista param.
11 for renglon in archivo:
12     datos = renglon.split()
13     if (len(datos)>0):
14         inst.append(datos[0]) #['LDV', 'STA', 'LDV', 'STA', 'LDV', 'STA', 'LDV', 'STA', 'LDA', 'SUB', 'JZ', 'LDA', '']
15         param.append(datos[1][: -1]) #['1', 'a', '5', 'b', '2', 'd', '0', 'c', 'c', 'b', '17', 'a', 'd', 'd', 'a', 'c']
16 archivo.close()

```

```
#En la segunda parte de nuestro codigo, depende de la instrucion(LDA,LDV,STA,DIV,MUL,SUB,ADD,JZ,JMP),
#el acumulador hara ciertas operaciones, hasta que encuentre en la lista la instruccion END(FIN).
#por ejemplo si la instruccion es ADD el acumulador sera igual al valor del acumulador actual mas
#el valor que se encuentra con la llave(param[pc]) en el diccionario.
#La operacion que realiza el acumulador la indica la instruccion, una vez realizada, se le aumenta 1 al contador(pc)
```

```
while (inst[pc]!="END"):
    #Imprime nuestro pc(direccion), acumulador,instruccion, parametro
    print ("PC: ", pc, " AC: ",ac," inst: ", inst[pc], " parametro: ", param[pc])
    if (inst[pc]=="LDV"): #Carga en el acumulador un valor.
        ac = int(param[pc])
        pc = pc + 1
    elif (inst[pc]=="STA"): #Acumulador->Memoria
        memDatos[param[pc]] = ac
        pc = pc + 1
    elif (inst[pc]=="LDA"): #Memoria-> Acumulador
        ac = memDatos[param[pc]]
        pc = pc + 1
    elif (inst[pc]=="ADD"): #Suma
        ac = ac + memDatos[param[pc]]
        pc = pc + 1
    elif (inst[pc]=="SUB"): #Resta
        ac = ac - memDatos[param[pc]]
        pc = pc + 1
    elif (inst[pc]=="MUL"): #Multiplica
        ac = ac * memDatos[param[pc]]
        pc = pc + 1
    elif (inst[pc]=="DIV"): #Divide
        ac = ac / memDatos[param[pc]]
        pc = pc + 1
    elif (inst[pc]=="JZ"): # Salto condicional, Salta si es cero. Funciona como etiqueta
        if (ac==0):
            pc = int(param[pc])
        else:
            pc = pc + 1
    elif (inst[pc]=="JMP"): #Salto incondicional
        pc = int(param[pc])
    elif (inst[pc]=="INC"): #Incrementa en 1
        valor = memDatos[param[pc]]
        valor = valor + 1
        memDatos[param[pc]] = valor
        pc = pc + 1
```

Cuando el acumulador sea 0, saltará hasta la línea X, por lo que pc valdrá X y la instrucción será END.

Caso contrario seguirá con la siguiente instrucción hasta encontrar JMP y regresar a la línea X.

b) Escriba 2 programas propios y ejecútelos

- PROGRAMA UNO

```
a=10;  
b=3;  
c=0;  
for (c; c<b; c++){  
a= a+b*b/b  
}
```

```
sugeilyPUno.txt  
1 LDV 10;  
2 STA a;  
3 LDV 3;  
4 STA b;  
5 LDV 0;  
6 STA c;  
7 LDA c;  
8 SUB b;  
9 JZ 16;  
10 LDA a;  
11 ADD b;  
12 MUL b;  
13 DIV b;  
14 STA a;  
15 INC c;  
16 JMP 6;  
17 END 0;
```

```
PC: 0 AC: 0 inst: LDV parametro: 10  
PC: 1 AC: 10 inst: STA parametro: a  
PC: 2 AC: 10 inst: LDV parametro: 3  
PC: 3 AC: 3 inst: STA parametro: b  
PC: 4 AC: 3 inst: LDV parametro: 0  
PC: 5 AC: 0 inst: STA parametro: c  
PC: 6 AC: 0 inst: LDA parametro: c  
PC: 7 AC: 0 inst: SUB parametro: b  
PC: 8 AC: -3 inst: JZ parametro: 16  
PC: 9 AC: -3 inst: LDA parametro: a  
PC: 10 AC: 10 inst: ADD parametro: b  
PC: 11 AC: 13 inst: MUL parametro: b  
PC: 12 AC: 39 inst: DIV parametro: b  
PC: 13 AC: 13.0 inst: STA parametro: a  
PC: 14 AC: 13.0 inst: INC parametro: c  
PC: 15 AC: 13.0 inst: JMP parametro: 6  
PC: 6 AC: 13.0 inst: LDA parametro: c
```

PC: 7	AC: 1	inst: SUB	parametro: b
PC: 8	AC: -2	inst: JZ	parametro: 16
PC: 9	AC: -2	inst: LDA	parametro: a
PC: 10	AC: 13.0	inst: ADD	parametro: b
PC: 11	AC: 16.0	inst: MUL	parametro: b
PC: 12	AC: 48.0	inst: DIV	parametro: b
PC: 13	AC: 16.0	inst: STA	parametro: a
PC: 14	AC: 16.0	inst: INC	parametro: c
PC: 15	AC: 16.0	inst: JMP	parametro: 6
PC: 6	AC: 16.0	inst: LDA	parametro: c

PC: 7	AC: 2	inst: SUB	parametro: b
PC: 8	AC: -1	inst: JZ	parametro: 16
PC: 9	AC: -1	inst: LDA	parametro: a
PC: 10	AC: 16.0	inst: ADD	parametro: b
PC: 11	AC: 19.0	inst: MUL	parametro: b
PC: 12	AC: 57.0	inst: DIV	parametro: b
PC: 13	AC: 19.0	inst: STA	parametro: a
PC: 14	AC: 19.0	inst: INC	parametro: c
PC: 15	AC: 19.0	inst: JMP	parametro: 6
PC: 6	AC: 19.0	inst: LDA	parametro: c

PC: 7	AC: 3	inst: SUB	parametro: b
PC: 8	AC: 0	inst: JZ	parametro: 16

- PROGRAMA DOS

```

a=15;
b=10;
d=5;
c= 0;
for (c; c<d; c++){
a= a+d/b
}

```

```

sugeilyPDos.txt
1  LDV 15;
2  STA a;
3  LDV 10;
4  STA b;
5  LDV 5;
6  STA d;
7  LDV 0;
8  STA c;
9  LDA c;
10 SUB d;
11 JZ 17;
12 LDA a;
13 ADD d;
14 DIV b;
15 STA a;
16 INC c;
17 JMP 8;
18 END 0;

```

```

['15', 'a', '10', 'b', '5', 'd', '0', 'c', 'c', 'd', '17', 'a', 'd', 'b', 'a', 'c', '8', '0']
['LDV', 'STA', 'LDV', 'STA', 'LDV', 'STA', 'LDV', 'STA', 'LDA', 'SUB', 'JZ', 'LDA', 'ADD', 'DIV', 'STA', 'INC', 'JMP', 'END']

```

```

PC: 0  AC: 0  inst: LDV  parametro: 15
PC: 1  AC: 15 inst: STA  parametro: a
PC: 2  AC: 15 inst: LDV  parametro: 10
PC: 3  AC: 10 inst: STA  parametro: b
PC: 4  AC: 10 inst: LDV  parametro: 5
PC: 5  AC: 5  inst: STA  parametro: d
PC: 6  AC: 5  inst: LDV  parametro: 0
PC: 7  AC: 0  inst: STA  parametro: c
PC: 8  AC: 0  inst: LDA  parametro: c
PC: 9  AC: 0  inst: SUB  parametro: d
PC: 10 AC: -5 inst: JZ  parametro: 17
PC: 11 AC: -5 inst: LDA  parametro: a
PC: 12 AC: 15 inst: ADD  parametro: d
PC: 13 AC: 20 inst: DIV  parametro: b
PC: 14 AC: 2.0 inst: STA  parametro: a
PC: 15 AC: 2.0 inst: INC  parametro: c
PC: 16 AC: 2.0 inst: JMP  parametro: 8

```

```

PC: 8  AC: 2.0 inst: LDA  parametro: c
PC: 9  AC: 1  inst: SUB  parametro: d
PC: 10 AC: -4 inst: JZ  parametro: 17
PC: 11 AC: -4 inst: LDA  parametro: a
PC: 12 AC: 2.0 inst: ADD  parametro: d
PC: 13 AC: 7.0 inst: DIV  parametro: b
PC: 14 AC: 0.7 inst: STA  parametro: a
PC: 15 AC: 0.7 inst: INC  parametro: c
PC: 16 AC: 0.7 inst: JMP  parametro: 8

```

```

PC: 8  AC: 0.7 inst: LDA  parametro: c
PC: 9  AC: 2  inst: SUB  parametro: d
PC: 10 AC: -3 inst: JZ  parametro: 17
PC: 11 AC: -3 inst: LDA  parametro: a
PC: 12 AC: 0.7 inst: ADD  parametro: d
PC: 13 AC: 5.7 inst: DIV  parametro: b
PC: 14 AC: 0.5700000000000001 inst: STA  parametro: a
PC: 15 AC: 0.5700000000000001 inst: INC  parametro: c
PC: 16 AC: 0.5700000000000001 inst: JMP  parametro: 8

```

```

PC: 8  AC: 0.5700000000000001 inst: LDA  parametro: c
PC: 9  AC: 3  inst: SUB  parametro: d
PC: 10 AC: -2 inst: JZ  parametro: 17
PC: 11 AC: -2 inst: LDA  parametro: a
PC: 12 AC: 0.5700000000000001 inst: ADD  parametro: d
PC: 13 AC: 5.57 inst: DIV  parametro: b
PC: 14 AC: 0.557 inst: STA  parametro: a
PC: 15 AC: 0.557 inst: INC  parametro: c
PC: 16 AC: 0.557 inst: JMP  parametro: 8

```

```

PC: 8  AC: 0.557 inst: LDA parametro: c
PC: 9  AC: 4  inst: SUB parametro: d
PC: 10 AC: -1 inst: JZ  parametro: 17
PC: 11 AC: -1 inst: LDA parametro: a
PC: 12 AC: 0.557 inst: ADD parametro: d
PC: 13 AC: 5.557 inst: DIV parametro: b
PC: 14 AC: 0.5557000000000001 inst: STA parametro: a
PC: 15 AC: 0.5557000000000001 inst: INC parametro: c
PC: 16 AC: 0.5557000000000001 inst: JMP parametro: 8

```

```

PC: 8  AC: 0.5557000000000001 inst: LDA parametro: c
PC: 9  AC: 5  inst: SUB parametro: d
PC: 10 AC: 0  inst: JZ  parametro: 17
PS C:\Users\55gus\OneDrive - UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO\6 semestre\Compiladores>

```

## Segunda parte de la tarea

**Investigue y haga un resumen de los siguientes conceptos:**

### Máquina virtual

Es un entorno virtual que funciona como sistema informático virtual con su propia CPU, memoria, interfaz de red y almacenamiento, pero se crea en un sistema de hardware físico, ya sea en las instalaciones o no. El sistema de software se llama hipervisor, y se encarga de separar los recursos de la máquina del sistema de hardware e implementarlos adecuadamente para que la VM pueda utilizarlos.

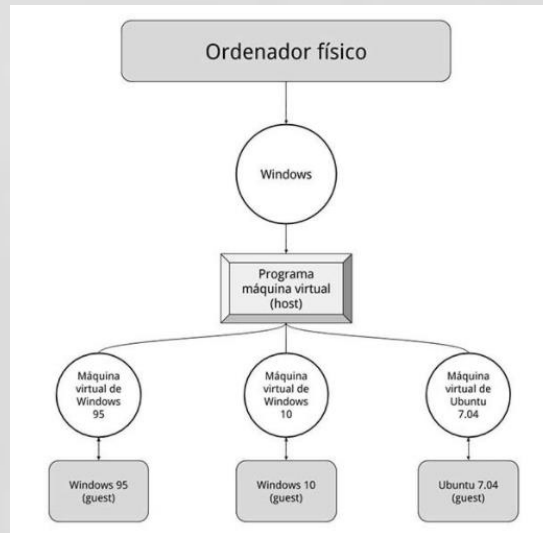
Todo funciona igual a si se estuviera ejecutando en un PC normal, sin que sepa que en verdad está metido dentro de una burbuja dentro de otro sistema operativo.

Hay dos tipos de máquinas virtuales diferenciadas por su funcionalidad;

- Máquinas virtuales de sistema

Es aquella que emula a un ordenador completo, tiene su propio disco duro, memoria, tarjeta gráfica y demás componentes de hardware, aunque todos ellos son virtuales.

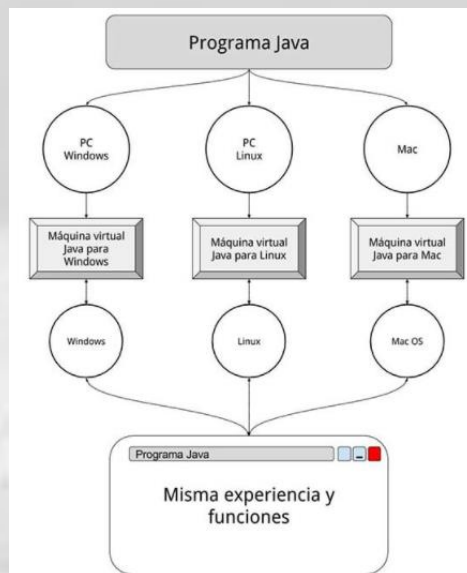
Para funcionar, una máquina virtual mapea los dispositivos virtuales que ofrece a su invitado con los dispositivos reales presentes en la máquina física.



- Máquinas virtuales de proceso:

Es aquella que ejecuta un proceso concreto, como una aplicación, en su entorno de ejecución.

Es de utilidad a la hora de desarrollar aplicaciones para varias plataformas, pues en vez de tener que programar específicamente para cada sistema, el entorno de ejecución (es decir, la máquina virtual) es el que se encarga de lidiar con el sistema operativo.



Las máquinas virtuales tienen una gran variedad de utilidades tanto en el entorno profesional como en el del consumidor final. Estos son los principales usos:

- Para poder probar otros sistemas operativos.
- Para ejecutar programas antiguos.



- Para usar aplicaciones disponibles para otros sistemas.
- Para probar una aplicación en distintos sistemas.
- Como seguridad adicional.
- Para aprovechar su gran dinamismo.

## Diferencia del manejo de memoria del modelo Harvard y el modelo de Von Neumann.

En la arquitectura Von Neumann hay un único espacio de memoria para datos y para instrucciones, en la arquitectura Harvard hay dos espacios de memoria separados: un espacio de memoria para los datos y un espacio de memoria para las instrucciones.

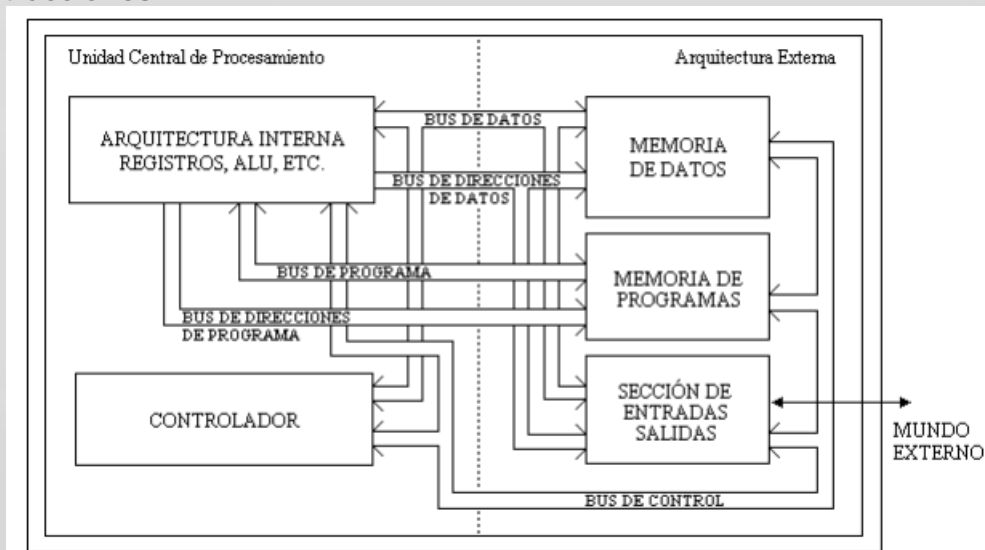


Figura 1.5. Computadora tipo Harvard.

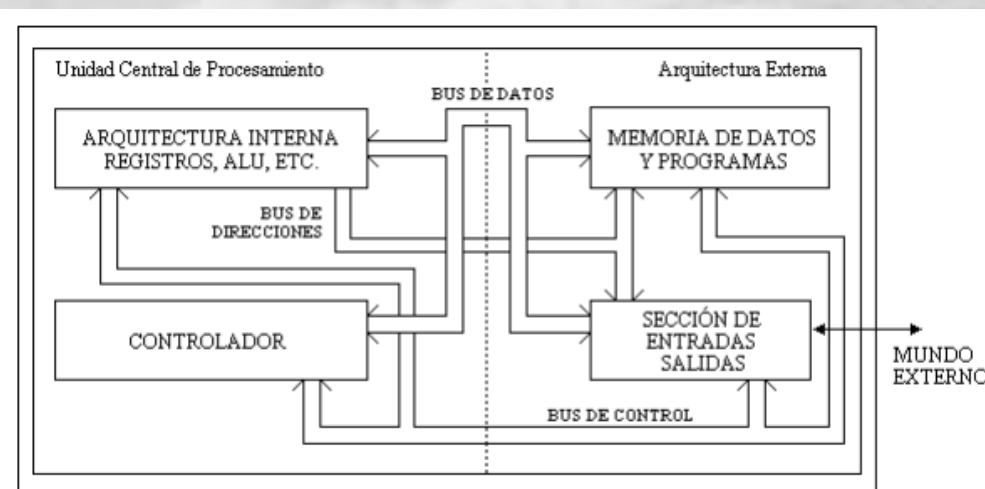


Figura 1.2. Computadora de von Neumann.



## Referencias

Iván Ramírez. (2016, July 25). Máquinas virtuales: qué son, cómo funcionan y cómo utilizarlas. Retrieved March 19, 2022, from Xataka.com website: <https://www.xataka.com/especiales/maquinas-virtuales-que-son-como-funcionan-y-como-utilizarlas>

Estructura de computadores. (2022). Retrieved March 19, 2022, from Uoc.edu website: [http://cv.uoc.edu/annotation/8255a8c320f60c2bfd6c9f2ce11b2e7f/619469/PID\\_00218274/PID\\_00218274.html](http://cv.uoc.edu/annotation/8255a8c320f60c2bfd6c9f2ce11b2e7f/619469/PID_00218274/PID_00218274.html)

Rubén Andrés. (2017, May 31). Qué es una máquina virtual, cómo funciona y para qué sirve. Retrieved March 19, 2022, from ComputerHoy website: <https://computerhoy.com/noticias/software/que-es-maquina-virtual-como-funciona-que-sirve-46606>

1.3: Von Neumann and Harvard Architectures. (2020, June 26). Retrieved March 19, 2022, from Engineering LibreTexts website: [https://eng.libretexts.org/Bookshelves/Electrical\\_Engineering/Electronics/Implementing\\_a\\_One\\_Address\\_CPU\\_in\\_Logisim\\_\(Kann\)/01%3A\\_Introduction/1.03%3A\\_Von\\_Neumann\\_and\\_Harvard\\_Architectures#:~:text=The%20major%20difference%20between%20the,data%20and%20one%20for%20instructions.](https://eng.libretexts.org/Bookshelves/Electrical_Engineering/Electronics/Implementing_a_One_Address_CPU_in_Logisim_(Kann)/01%3A_Introduction/1.03%3A_Von_Neumann_and_Harvard_Architectures#:~:text=The%20major%20difference%20between%20the,data%20and%20one%20for%20instructions.)

Von-Neumann vs Harvard Architecture | Differences & Uses. (2020, June 17). Retrieved March 19, 2022, from Teach Computer Science website: <https://teachcomputerscience.com/von-neumann-harvard-architecture/>