

## Diseño y Análisis de Algoritmos

### Práctica 2 - Experimentación y complejidad computacional - Patrón Estrategia - Simulador de la máquina RAM

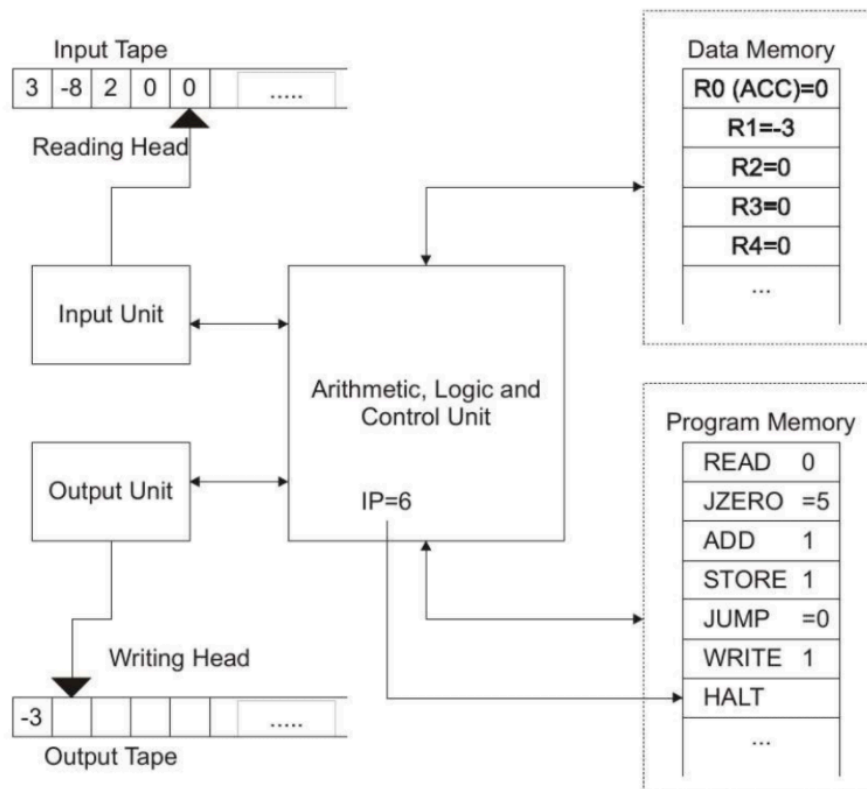
#### Factor de ponderación: 35%

#### Descripción de una máquina RAM

Una *Máquina de Acceso Aleatorio (RAM)* es aquella máquina teórica que se utiliza para el cálculo de la complejidad de los diferentes algoritmos en esta asignatura. Ello se debe a que posee 3 características básicas:

- No posee concurrencia.
- Sus instrucciones son de tiempo constante.
- Permite medir el número de instrucciones ejecutadas.

Una máquina RAM se estructura como se puede indicar en la siguiente imagen:



Así pues, podemos decir que la máquina RAM se divide en 5 grandes componentes:

1. **Memoria de programa:** Guarda el conjunto de instrucciones de un programa. Esta memoria **no es modificable**.
2. **Memoria de datos:** Posee una serie de registros indexados R0, R1, R2, ..., Rn para almacenar datos. El registro R0 es un registro especial llamado **acumulador**, el cual ayuda en la realización de operaciones aritméticas, lógicas y de control.
3. **Unidad de aritmética, control y lógica:** Permite realizar operaciones aritméticas (ADD, SUB, DIV, etc.), de control de programa (JUMP) o lógicas (JGTZ, JZERO).
4. **Unidad de entrada:** Gestiona la entrada de datos de la máquina. La entrada de datos viene representada por una cinta de elementos, y para su lectura se dispone de un cabezal que se mueve a la siguiente posición cada vez que se realiza una operación de lectura (READ).
5. **Unidad de salida:** Gestiona la salida de datos de la máquina. La salida de datos viene representada por una cinta de elementos, y para su escritura se dispone de un cabezal que se mueve a la siguiente posición cada vez que se realiza una operación de escritura (WRITE).

La máquina posee un conjunto de instrucciones finito, cada una de ellas formada por una combinación de OPERADOR + OPERANDO, pudiendo ser el operando una etiqueta. Las instrucciones disponibles en nuestra máquina RAM serán las siguientes:

Instrucción	Descripción
LOAD op	El operando se carga en R0.
STORE op	El contenido de R0 se guarda en la memoria según lo indicado en el operando.
ADD op	El operando se suma a R0 y el resultado se almacena en R0. ( $R0 = R0 + op$ )
SUB op	El operando se resta a R0 y el resultado se almacena en R0. ( $R0 = R0 - op$ )
MUL op	El operando se multiplica con R0 y el resultado se almacena en R0. ( $R0 = R0 * op$ )
DIV op	El operando divide a R0 y el resultado se almacena en R0. ( $R0 = R0 / op$ )
READ op	Se lee un valor de la cinta de entrada y se almacena en la memoria según lo indicado en el operando.
WRITE op	Se escribe lo indicado por el operando en la cinta de salida.
JUMP etiq	El control del programa salta a la instrucción indicada por la etiqueta.
JZERO etiq	El control del programa salta a la instrucción indicada por la etiqueta si el valor de R0 es igual a 0 ( $R0 == 0$ ).
JGTZ etiq	El control del programa salta a la instrucción indicada por la etiqueta si el valor de R0 es mayor a 0 ( $R0 > 0$ ).
HALT	Detiene la ejecución del programa.

Además de las etiquetas, la máquina RAM define 3 tipos de operandos:

- **Operando =i:** Constante con valor i. Por ejemplo,  $\text{LOAD } =10 \rightarrow R0 = 10$ . Este tipo de operando no tiene sentido para instrucciones que necesiten el valor de un registro como STORE.
- **Operando i:** Indica el registro Ri (direccionamiento directo). Por ejemplo,  $\text{LOAD } 10 \rightarrow R0 = R10$ .
- **Operando \*i:** Indica el registro contenido en el registro Ri (direccionamiento indirecto). Por ejemplo,  $\text{LOAD } *10 \rightarrow R0 = Rj$ , siendo j el contenido del registro R10.

Un programa RAM consiste en una secuencia de instrucciones que se ejecutan de manera secuencial, salvo que se encuentren sentencias de control como JUMP, JZERO o JGTZ. La ejecución del programa se hace de acuerdo a las siguientes reglas:

- La ejecución comienza en la primera instrucción del programa, con todos los registros de la memoria vacíos y con los datos de entrada cargados en la cinta de entrada.
- Se ejecuta una instrucción, se modifican los registros de memoria necesarios o la cinta de salida y, acto seguido, se pasa a la ejecución de la siguiente instrucción.
- Las instrucciones READ y WRITE leen y escriben en las cintas correspondientes, y en ambos casos se avanza una posición en la cinta. No se puede utilizar como operando de las instrucciones READ y WRITE el registro R0. Así por ejemplo, las instrucciones READ 0 y WRITE 0 no son válidas.
- El programa termina cuando no hay más instrucciones a ejecutar o cuando se encuentra la instrucción HALT.

La complejidad de un programa en la máquina RAM es equivalente al número de instrucciones ejecutadas por el mismo.

## Requisitos evaluables del simulador RAM

Los siguientes requisitos se evaluarán de cara a la entrega de la práctica:

1. Todo el código deberá estar adecuadamente comentado y desarrollado atendiendo al paradigma de Programación Orientada a Objetos.
2. Se deben seguir los principios SOLID así como el patrón estrategia, con el fin de poseer un buen diseño del software.
3. El simulador debe funcionar con los ejemplos de prueba proporcionados junto con este enunciado.
4. El programa RAM a cargar en la memoria, así como el contenido de la cinta de entrada y de la cinta de salida se modelará mediante tres ficheros independientes, los cuales serán parámetros de entrada del programa. Estos son:
  - Fichero con el programa RAM.
  - Fichero con el contenido de la cinta de entrada.
  - Fichero con el contenido de la cinta de salida.
5. Las instrucciones de la máquina RAM pueden estar escritas en mayúscula o en minúscula.

6. Se pueden especificar comentarios en un programa RAM, indicados por el carácter #. La línea que empiece por # no se cargará como parte del programa RAM.
7. Se deben contemplar los 3 tipos de operandos: constante, direccionamiento directo y direccionamiento indirecto.
8. Se debe comprobar que las instrucciones sean válidas. Sólo se permiten las instrucciones ya especificadas en la tabla correspondiente, con el formato indicado en la misma. Por ejemplo, la instrucción STORE =3 no es válida, ya que STORE no permite un comando de tipo constante.
9. Si se detecta algún error se debe imprimir en la consola un mensaje que indique la instrucción errónea y el número de línea dentro del programa RAM que lo provocó.
10. Cuando se alcance una instrucción HALT o se detecte un error, el contenido de la cinta de salida debe volcarse en el fichero indicado.

## Extensión de una máquina RAM

Esta extensión de la máquina RAM implementada consiste en ampliar los registros R para que permitan almacenar vectores de tamaño dinámico.

1. Esto implica que ahora el operando de direccionamiento directo puede verse como **i[j]**, siendo equivalente a decir  $R_i[j]$ . Por ejemplo, LOAD R1[2] carga en R0 el contenido de la posición 2 del vector contenido por R1 ( $R_0 = R_1[2]$ ).

Aprovechando la ampliación realizada, se debe implementar en la máquina RAM y comparar el número de ejecuciones para instancias de diferente tamaño 2 algoritmos de ordenación:

### Inserción:

```
1 for j = 2 to n do
2   key = A[j]
3   i = j - 1
4   while i > 0 and A[i] > key
5     A[i + 1] = A[i]
6     i = i - 1
7   end
8   A[i + 1] = key
9 end
```

### BubbleSort:

```
1 for i = 1 to n-1 do
2   for j = n downto i + 1 do
3     if A[j-1] > A[j] then begin
4       temp = A[j-1]
5       A[j-1] = A[j]
6       A[j] = temp
7     end
8   end
9 end
```

Los resultados de esta comparación deben reflejarse en un informe de una página.

## Objetivos de la práctica

Los siguientes objetivos se consideran **condición necesaria pero no suficiente** para aprobar la práctica:

1. Programar un simulador de máquina RAM atendiendo a las especificaciones indicadas en la sección **Descripción de una máquina RAM y Requisitos evaluables de un simulador RAM**.
2. Utilizar el paradigma de **Programación Orientada a Objetos**, así como seguir los principios **SOLID** y el **patrón estrategia** en caso de ser necesario.
3. Utilizar los lenguajes de programación **C++**, **Java** o **C#**.

La implementación de los objetivos anteriores permitirá conseguir hasta una nota máxima de **APTO**. Si se quiere alcanzar una nota de **APTO+** se debe implementar lo indicado en la sección **Extensión de la máquina RAM**.

Durante la defensa de la práctica **se podrá solicitar algún tipo de modificación o prueba adicional**, la cual afectará en diferente grado a la nota final.

## Entrega de la práctica

La práctica debe entregarse en tiempo y forma acorde a lo indicado en la tarea del campus virtual. Para poder considerar la práctica como aprobada, deben cumplirse 2 requisitos:

- La práctica debe defenderse en su sesión correspondiente. Además, debe funcionar tal y como se especifica en el presente enunciado.
- Se debe entregar la tarea del campus virtual incluyendo un fichero con extensión **tar.gz** o **zip** con el código fuente del programa.

La no realización de uno de estos 2 puntos conlleva la calificación como suspenso de la práctica.