Sistemas Operativos
Carlos Hernán Gómez Gómez
CHMAQUINA V2022

Implemente un Programa **que corra sobre un computador y la Web o un dispositivo inteligente (ej. Telefono movil),** que realice una simulación gráfica de un **chcomputador** ficticio de funcionamiento básico.
La herramienta para el desarrollo del proyecto queda a discreción del estudiante, se recomienda utilizar lenguajes/herramientas de programación modernos que tengan implementadas facilidades graficas de desarrollo.

El programa debe simular un procesador muy elemental y una memoria principal a través de un vector o arreglo unidimensional de hasta 1000*Z+ 100 posiciones (donde Z será el último digito de la cedula del estudiante), las cuales pueden ser variadas al momento de iniciar el programa, se asume por defecto que el chcomputador empieza con Z*10 + *50* posiciones de memoria para facilitar el proceso de pruebas. El programa debe estar en capacidad de leer un conjunto de programas en un seudo lenguaje de máquina que llamaremos **CHMAQUINA** y los cargará en las posiciones disponibles de la citada memoria, leerá una instrucción por cada línea de entrada.

Se asumirá que las primeras posiciones de la memoria estarán reservadas para el núcleo del sistema operativo (parte residente del sistema operativo o kernel), su contenido para este proyecto no es importante y solo se verá como área reservada, el tamaño de este deberá poderse ingresar al iniciar la corrida del simulador, su valor por defecto es 10*Z + 9 posiciones y su tamaño se podrá variar solo al iniciar el ambiente de trabajo.

El proyecto estará dividido por fases como se detalla a continuación.

CHMAQUINA Fase A. Interfaz gráfica, carga, chequeo de sintaxis y ejecución.
En esta fase el estudiante diseñara la interfaz gráfica que realizara para el proyecto, es decir la disposición de componentes gráficos que se le mostraran al usuario, permitirá la lectura de los archivos de programas con extensión ".ch" y los colocará en el panel de carga de programas, presentara las estructuras de código que permitan hacer el chequeo de sintaxis.

El programa deberá realizar un chequeo de sintaxis, produciendo un listado de errores si los hay, de lo contrario procederá a la carga definitiva del programa en memoria (una instrucción por celda) y quedará listo para ejecución del mismo bajo las reglas de corrida multitarea (es decir **múltiples programas)** como se indica más adelante.

Reglas del CHMAQUINA

El programa utilizará un acumulador para registrar los valores de los cálculos y recibirá como nombre reservado "**acumulador**".

Las posiciones de memoria que almacenen datos (variables) tendrán un nombre asociado que iniciara con una letra y máximo tendrá 255 caracteres sin espacios intermedios. Estas variables deberán ser creadas antes de ser usada.

El código puede tener comentarios por líneas, los cuales se denotarán por dos *backslash* (//) en las dos primeras posiciones de la instrucción, de igual manera se podrán insertar líneas en blanco entre instrucciones del programa, cuyo propósito es de legibilidad del programa. Los chprogramas serán almacenados previamente en archivos con extensión "ch" en cualquier carpeta de algún medio de almacenamiento, de allí podrán ser cargados.

El sistema es sensible a *case*, es decir diferenciara letras mayúsculas de minúsculas.

Sistemas Operativos
Carlos Hernán Gómez Gómez
CHMAQUINA V2022

Se podrán realizar operaciones entre valores enteros y reales, los resultados intermedios se manejarán como reales y el resultado final obedecerá al tipo de variable que almacena el resultado (realizando truncamiento si es necesario).

Las instrucciones constarán de 2 partes; el código de la operación y el(los) operando(s) dependiendo el tipo de instrucción (operacion opernado1 operando2…..operandon).

El código de operación corresponde al nemónico del código de operación que se explican en la siguiente tabla:

Operación	Descripción
cargue	Cárguese/copie en el acumulador el valor almacenado en la variable indicada por el operando.
almacene	Guarde/copie el valor que hay en el Acumulador en la posición de memoria que corresponda a la variable indicada por el operando.
nueva	Crea una nueva variable cuyo nombre es el especificado en el primer operando, en el segundo operando definirá el tipo de variable (C Cadena/alfanumérico, I Entero, R Real/decimal, L lógico o booleano (**1** Verdadero o **0** Falso), un tercer operando establecerá un valor de inicialización. A cada variable se le asignará automáticamente una posición en la memoria. **Las variables deberán estar definidas antes de ser utilizadas**. Las variables no inicializadas tendrán por defecto el valor cero para reales y enteros, espacio para cadenas, 0 para lógicos. El separador de decimales es el punto.
lea	Lee por teclado/pantalla el valor a ser asignado a la variable indicado por la variable referida en el operando.
sume	Incremente/sume al valor del acumulador el valor indicado por la variable señalada por el operando.
reste	Decremente/reste del valor del acumulador el valor indicado por la variable que señala el operando.
multiplique	Multiplique el valor del acumulador por el valor indicado por la variable señalada por el operando.
divida	Divida el valor del acumulador por el valor indicado por la variable señalada por el operando. El divisor deberá ser una cantidad diferente de cero.
potencia	Eleve el acumulador a la potencia señalada por el operando (los exponentes pueden ser valores enteros, positivos o negativos)
modulo	Obtenga el modulo al dividir el valor del acumulador por el valor indicado por la variable señalada por el operando.
concatene	Genere una cadena que una la cadena dada por el operando a la cadena que hay en el acumulador (Operando alfanumérico). El contenido del acumulador deberá tratarse como cadena en caso de ser numérico.
elimine	Genere una subcadena que elimine cualquier aparición del conjunto de caracteres dados por el operando de la cadena que se encuentra en el acumulador (operando alfanumérico).
extraiga	Genere una subcadena que extraiga los primeros caracteres (dados por el operando con valor numérico) de la cadena que se encuentra en el acumulador.
Y	Produce una operación lógica Y (AND) entre el primer operando y el segundo operando que son variables lógicas y la almacena en el tercer operando.

Sistemas Operativos
Carlos Hernán Gómez Gómez
CHMAQUINA V2022

O	Produce una operación lógica O (OR) entre el primer operando y el segundo operando que son variables lógicas y la almacena en la variable del tercer operando.
NO	Produce una operación de negación lógica para el primer operando que es una variable lógica y el resultado se almacena en la variable del segundo operando.
muestre	Presente por pantalla el valor que hay en la variable indicada por el operando, si el operando es acumulador muestre el valor del acumulador.
imprima	Presente por la impresora el valor que hay en la variable indicada por el operando, si el operando es acumulador muestre el valor del acumulador.
retorne	El programa termina; debe ser la última instrucción del programa y tiene opcionalmente un operando numérico entero.
vaya*	Salte a la instrucción que corresponde a la etiqueta indicada por el operando y siga la ejecución a partir de allí.
vayasi*	Si el valor del acumulador es mayor a cero salte a la instrucción que corresponde a la etiqueta indicada por el primer operando y continue la ejecución a partir de allí. Si el valor del acumulador es menor a cero salte a la instrucción que corresponde a la etiqueta indicada por el segundo operando y continue la ejecución a partir de allí. o Si el acumulador es cero salte a la siguiente instrucción adyacente a la instrucción vayasi y siga la ejecución a partir de allí.
etiqueta	La etiqueta es un nombre que opcionalmente se le puede asignar a una instrucción en el programa para evitar trabajar con las posiciones en memoria de las instrucciones y poder utilizar un nombre simbólico independiente de su ubicación. Crea una nueva etiqueta cuyo nombre es el especificado en el primer operando y a la cual le asignará automáticamente la posición indicada en el segundo operando (esta será la posición relativa de la instrucción a la que se le asigna este nombre con respecto a la primera instrucción del programa). Las instrucciones que definen etiquetas podrán definirse en cualquier posición del programa, pero en todo caso antes de la instrucción retorne.
XXX	**Esta será una función complementaria a las anteriores creada por el estudiante, se deja a voluntad su nombre y forma de funcionamiento.**

La ejecución de los programas normalmente se hace de forma secuencial de instrucciones, la primera después la segunda, la tercera....etc, las instrucciones de transferencia de control (vaya y vayasi) son la forma de cambiar este orden de ejecución, obligando que el programa no siga en el orden secuencial predeterminado, sino que continué en la instrucción señalada por una etiqueta (es decir una instrucción que tiene asignado un nombre como referencia). vaya y vayasi cumple esta función, la primera de forma incondicional y la segunda condicionada al valor del acumulador como se especifica en su definición.

El programa no debe permitir la sobrecarga del acumulador (Overflow/desborde) por lo cual sacará un mensaje de error que le permita al usuario tomar la decisión que corresponda. Hasta la fase C(inclusive) la protección de memoria se hará por registro base y registro límite, esto es, cada programa empieza en una posición de memoria (registro base) y termina en otra posición de memoria denominada (registro limite) con base en las cuales se evitará la violación de las normas básicas de ejecución, también debe tenerse claro que los programas tendrán área de código y área de datos.

Para prueba del programa pueden plantearse diversos **CHMAQUINA** que irán aumentando en complejidad; también podrá aumentarse el grupo de instrucciones y la capacidad de memoria de nuestro Computador ficticio.

Sistemas Operativos
Carlos Hernán Gómez Gómez
CHMAQUINA V2022

Para cada una de las fases debe presentarse el manual técnico (donde se explica de forma técnica como esta construido el programa con los diagramas y código utilizados en todo el proceso de realización del proyecto) y de usuario (paso a paso como instalar y usar el programa), código fuente, código ejecutable, sustentación completa.
Semanalmente debe registrar la bitácora de avance en el blog de seguimiento del estudiante, donde se informará en forma concisa de las tareas realizadas y pendientes de realizar en el proyecto. En el asunto del mensaje se colocara: "Bitácora de avance chmaquina AA/MM/DD inicial a AA/MM/DD final". La bitácora deberá incluir el código fuente elaborado del proyecto hasta ese momento. Toda la documentación y avances del proyecto se irán registrando en el blog creado por el estudiante. La no presentación de bitácora generará un castigo en la nota del 10% por cada semana de no presentación**. Estas condiciones rigen para todas las fases**
En la primera bitácora se presentará un cronograma tentativo de desarrollo del proyecto en el cual se incluirán actividades (como estudio preliminar, planeación, diseño, desarrollo, control de calidad, documentación). Deberá entregarse un documento con la tabla de contenido tentativa de la documentación del proyecto. Cada semana debe mostrarse el avance en el cronograma y en la documentación.

Se podrán cargar y correr varios chprogramas hasta agotar la memoria disponible, para la corrida de los chprogamas se recomienda utilizar una lista de tipo cola circular, dado que en la fase B se requerirá.

El programa podrá ejecutarse en modalidad normal (corrida continua) o paso a paso (instrucción por instrucción), en todo caso se podrá visualizar la instrucción que se esté ejecutando en cada momento y el respectivo valor del acumulador.

En cualquier momento en la ejecución de los programas podrá pedírsele al sistema mostrar el mapa de memoria (es decir el Vector de memoria y sus posiciones, las variables con sus valores, lo mismo que el valor del acumulador).

Deberá proveerse al ambiente de desarrollo con un editor de programas chmaquina (en el panel de carga de programas), en el cual se puedan crear chprogramas y/o modificarlos, este editor hará el chequeo de sintaxis básico indicándole al usuario las instrucciones con errores y podrá sobre el mismo permitir la corrida del programa si no presenta errores.

El sistema debe indicar por medio de alguna convención si está trabajando en modo usuario (ejecución del programa) o modo kernel (el sistema tiene el control y administración del ambiente), mostrando la acción de cambio de contexto (el paso de un modo al otro). Se podrán ver los distintos estados en los cuales estén los procesos, a nivel de proceso y a nivel de cola.

Todas las especificaciones planteadas hasta este punto conforman los objetivos a cumplir en la fase A del programa, es decir el proyecto debe estar en capacidad de cargar los programas con extensión ch, realizar la verificación de sintaxis y poderlo correr si no tiene errores el programa, el proyecto deberá permitir cargar y correr tantos programas como los que admita la capacidad de memoria de la memoria del simulador.

Sistemas Operativos
Carlos Hernán Gómez Gómez
CHMAQUINA V2022

CHMAQUINA Fase B. Planificación de procesos y consolidación del ambiente.

Para abordar esta fase se debe garantizar que las fases anteriores estén funcionando adecuadamente, además deberán realizarse o preverse los ajustes necesarios para culminar con esta fase. **En esta fase se podrán variar los algoritmos de planificación de procesos.**

Para la planificación de procesos y administración de memoria asuma :
Que el conteo de las instrucciones de entrada y salida da las ráfagas de I/O (Lea, Muestre, Imprima), el conteo de las demás instrucciones excluyendo las declarativas (Nueva, Etiqueta, Retorne, comentarios) da las ráfagas de CPU.

El tiempo de llegada de un programa es igual al tiempo de llegada del programa anterior más el número de instrucciones de tal programa sobre 4, se asume que el primer programa llego en t=0.

Se debe permitir la carga de programas en cualquier momento, en el proceso de carga de programas se supone que la cpu está suspendida en modo kernel(no hay actividad de los contadores del sistema).

Las prioridades, que serán establecidas al cargar cada programa, se definirán como un numero entero entre 0 y 99 siendo la más alta la de mayor valor.
Se debe permitir elegir el método de planificación de procesos al iniciar el chmaquina, de acuerdo con él se pedirán los parámetros adicionales necesarios a cada proceso, por defecto se asumirá el método round-robin, a cada proceso se le asignarán **Q** unidades de tiempo (slice), este parámetro se podrá modificar al cargar el ambiente de trabajo, tendrá un valor por defecto de 5 (**para efectos de este programa se asumirá que cada instrucción de entrada y salida consume de forma aleatoria de 1 a 9 unidades de tiempo, las demás instrucciones consumen una unidad de tiempo**),

Los métodos a tener en cuenta son: FCFS, Round Robin, SJF expropiativo y no expropiativo, y por prioridad (expropiativo y no expropiativo). Para evitar la inanición en los algoritmos que corresponda se debe usar compensación por envejecimiento (esta implementación se deja a criterio del desarrollador).

CHMAQUINA Fase C. Administración de memoria.

El sistema permitirá seleccionar el método de administración de memoria: múltiples particiones fijas (permitiendo elegir el tamaño de las particiones al seleccionar esta opción), particiones variables (cuando corresponda debe permitirse condensación y/o compresiòn), almacenamiento virtual con paginación de un solo nivel (tome por defecto el área de intercambio igual al doble de la memoria principal, pero permita que el usuario pueda cambiar su tamaño al momento de seleccionar esta opción), asuma un tamaño de página de 5 unidades de memoria que podrán cambiarse al momento de elegir este método, asuma un algoritmo de reemplazo de páginas de reloj/segunda oportunidad, asuma que el conjunto de trabajo es la página donde está el contador de programa (PC) y la siguiente página.

El sistema debe permitir ver la tabla de procesos (con la información relevante), las distintas colas, el mapa de memoria principal y de intercambio (si aplica)

Sistemas Operativos
Carlos Hernán Gómez Gómez
CHMAQUINA V2022

A continuación, relaciono algunos chprogramas de ejemplo para su análisis y prueba de escritorio, se pondrá a disposición un conjunto de programas de prueba.

Programa factorial.ch

```
// Programa para calcular el factorial de 5
nueva unidad I 1
nueva m I 5
nueva respuesta I 1
nueva intermedia I 0
cargue m
almacene respuesta
reste unidad
// Se inicia el ciclo de cálculo del factorial
almacene intermedia
cargue respuesta
multiplique intermedia
almacene respuesta
cargue intermedia
reste unidad
vayasi itere fin
etiqueta itere 9
etiqueta fin 21
muestre respuesta
imprima respuesta
retorne 0
```

Programa Multipliquevar.ch

```
// Calcula el producto de dos valores entrados por teclado.
nueva unidad  I 1
nueva m I 0
nueva n  I 0
lea m
lea n
nueva respuesta I 0
nueva resultado C Resultado=
nueva intermedia I 0
cargue n
// inicia ciclo
reste unidad
almacene intermedia
cargue respuesta
sume m
almacene respuesta
cargue intermedia
vayasi ciclo fin
```

```
etiqueta ciclo 11
etiqueta fin 25
muestre resultado
muestre respuesta
imprima resultado
imprima respuesta
retorne 0
```

Programa operacioneslogicas.ch: Realiza varias operaciones lógicas

```
nueva unidad I 1
nueva  n I 9
// creo variable P de tipo lógico valor inicial 1 (verdadero)
Nueva P L 1
// creo variable Q de tipo lógico valor inicial 1 (verdadero)
Nueva Q L 1
// creo variable R de tipo lógico sin valor inicial, el sistema lo pone en 0
Nueva R L
// Se realiza operación lógica AND entre P y Q, resultado se coloca en R
Y P Q R
Muestre R
// Se realiza operación lógica OR entre P y Q, el resultado se coloca en R
O P Q R
Muestre R
// Se realiza operación lógica Negación entre P y Q, el resultado se coloca en R
NO PQR
Muestre R
Retorne 0
```