**Nombre: Andres Urbano Guillermo Gerardo**

**Total:\_\_\_\_\_\_\_\_/30**

**Conteste lo que se solicita. Cada pregunta tiene un valor indicado.**

1.- Responda lo siguiente:

a) (0.5 pts) ¿Qué es una interrupción?

Una interrupción es una operación que suspende la ejecución de un programa de manera que el sistema pueda realizar una acción especial.

b) (0.5 pt) ¿Qué es una interrupción interna y qué es una interrupción externa?

Interrupciones internas

ocurren como resultado de la ejecución de una instrucción de interrupción INT, o una operación de división que cause desbordamiento, ejecución en modo de paso a paso (debugger) o una petición para una operación externa como E/S de disco.

Interrupciones externas: son provocadas por dispositivos externos al microprocesador.

b) (1 pt) ¿Cómo se determina la posición en memoria de una rutina de interrupción?

Por la tabla de vectores que se encuentra en los primeros 1024 bytes de memoria, en las direcciones 00000000h –000003FFh.

2.- Responda lo siguiente:

a) (0.5 pt) ¿Qué es un cargador?

Es un programa que pone el código máquina en las principales localidades de memoria primaria en las cuales éste será ejecutado.

b) (0.5 pt) ¿Qué es un encadenador?

Es un programa que resuelve referencias entre unidades de programa (otros archivos, como .obj y .lib) y que también permite el acceso a bibliotecas del sistema.

c) (0.5 pt) ¿Cuáles son las 4 tareas principales de un cargador?

Enlace, reubicación, asignación de memoria y carga.

d) (0.5 pt) ¿Qué ventaja tiene el enlazado dinámico?

Ocupa menos memoria, pero requiere calcular las direcciones en tiempo de ejecución y es más propenso a fallas y errores.

3.- Conteste:

a) (0.5 pt) ¿Qué es el Acceso Directo a Memoria (DMA)?

Es una técnica que permite que un dispositivo de E/S tenga acceso a la memoria principal del sistema mientras el microprocesador está temporalmente deshabilitado

b) (1 pts) Describa el funcionamiento del Controlador DMA.

El controlador de DMA proporciona señales e información de direcciones a la memoria y a los dispositivos E/S durante la transferencia DMA.

El DMAC funciona como un microprocesador de propósito específico.

Se deshabilitan los buses desde el CPU y el control lo tiene el DMAC.

c) (0.5 pt) Con sus propias palabras, explique los problemas de memoria caché que se generan cuando se utiliza DMA.

En pocas palabras como dma tiene el control pude modificar la memoria cache y como el CPU esta siempre en contacto con la memoria Cache, cuando modifica no le avisa a la CPU y al no avisarle puede tener en la memoria Cache información obsoleta o no actualizada.

4.- (2 pts) Si le pidieran diseñar un programa que realiza asignación de memoria:

- ¿Lo haría con memoria contigua o no contigua?

- Si es contigua, ¿particionamiento estático o dinámico?

Parcionamiento dinámico porque cada bloque se asigna a una localidad de memoria disponible suficiente asi no habría huecos y el espacio de memoria de utilizaría correctamente.

- Si es no contigua, ¿por paginación o por segmentación?

Lo haría por paginación ya que se utiliza mejor la memoria, la administración es mucho mejor porque cada marco es del mismo tamaño que una página de un proceso.

Justifique sus respuestas.

5.- Marque la opción correcta a las siguientes afirmaciones:

a) (0.5 pts) Lee el archivo de código fuente en la primera pasada y una sola vez.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Ensamblador de una pasada | 2) Ensamblador de dos pasadas | 3) Ensamblador de una y media pasadas | 4) Todas las anteriores |

R.- 1)

b) (0.5 pts) Puede generar código objeto.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Ensamblador de una pasada | 2) Ensamblador de dos pasadas | 3) Ensamblador de una y media pasadas | 4) Todas las anteriores |

4.- Todas la anteriores

c) (0.5 pts) Genera un archivo intermedio antes de generar un archivo objeto.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Ensamblador de una pasada | 2) Ensamblador de dos pasadas | 3) Ensamblador de una y media pasadas | 4) Todas las anteriores |

2.- Ensamblador de dos pasadas

d) (0.5 pts) Contiene la información de palabras reservadas para el ensamblador, pero esas palabras no necesariamente se traducen en instrucciones del procesador.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Tabla de símbolos | 2) Tabla de instrucciones | 3) Tabla de directivas | 4) Tabla de definición de macros |

2.- Tabla de definición de marcros

e) (0.5 pts) Almacena símbolos en la tabla de símbolos.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Ensamblador de una pasada | 2) Ensamblador de dos pasadas | 3) Ensamblador de una y media pasadas | 4) Todas las anteriores |

4.- Todas la anteriores

f) (0.5 pts) Contiene el código de operación de cada instrucción.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Tabla de símbolos | 2) Tabla de instrucciones | 3) Tabla de directivas | 4) Tabla de definición de macros |

2.- Tabla de instrucciones

g) (0.5 pts) Es una tabla interna y dinámica que se genera y se usa por el ensamblador.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Tabla de símbolos | 2) Tabla de instrucciones | 3) Tabla de directivas | 4) Tabla de definición de macros |

1.- Tabla de simbolos

h) (0.5 pts) En la primera pasada genera un archivo objeto incompleto.

|  |  |  |  |
| --- | --- | --- | --- |
| 1) Ensamblador de una pasada | 2) Ensamblador de dos pasadas | 3) Ensamblador de una y media pasadas | 4) Todas las anteriores |

3.- Ensamblador de una y media pasadas.

6.- Observe las tablas que se muestran a continuación.

|  |  |  |
| --- | --- | --- |
| **Contador de localidades** | **Código fuente** | **Código objeto** |
|  | .data |  |
|  | x db 10 |  |
|  | y dw 0 |  |
|  | z db 50 |  |
|  | w dw 0 |  |
|  | .code |  |
| LC = 0 | inicio: |  |
| LC = 0 | mov ax,48B0h | B8 B0 48 |
| LC = 3 | mov ds,ax | 8E D8 |
| LC = 5 | mov al,[x] | A0 00 00 |
| LC = 8 | mov bl,[z] | 8A 1E 03 00 |
| LC = 12 | mul bl | F6 E3 |
| LC = 14 | mov [y],ax | A3 01 00 |
| LC = 17 | cmp ax,300d | 3D 2C 01 |
| LC = 20 | jl salir | 7C 14 90 90 |
| LC = 24 | cmp ax,1000d | 3D E8 03 |
| LC = 27 | jg salir | 7F 0D 90 90 |
| LC = 31 | mov al,[u] | A0 1F 00 |
| LC =33 | mov bl,[v] | 8A IE 21 00 |
| LC =37 | mul bl | F6 E3 |
| LC =39 | mov [w],ax | A3 04 00 |
| LC = 42 | salir: |  |
| LC = 42 | mov ax,4C00h | B8 00 4C |
| LC = 45 | int 21h | CD 21 |
| LC = 47 | end inicio |  |

|  |  |  |
| --- | --- | --- |
| **Tabla de instrucciones** | | |
| **Instrucción** | **Código de operación** | **Longitud [bytes]** |
| mov ax,imm16 | B8 ?? ?? | 3 |
| mov ds,ax | 8E D8 | 2 |
| mov al,m8 | A0 ?? ?? | 3 |
| mov bl,m8 | 8A 1E ?? ?? | 4 |
| mul bl | F6 E3 | 2 |
| mov m16,ax | A3 ?? ?? | 3 |
| cmp ax,imm16 | 3D ?? ?? | 3 |
| jl rel8 | 7C ?? | 2 |
| jl rel16 | 0F 8C ?? ?? | 4 |
| jg rel8 | 7F ?? | 2 |
| jg rel16 | 0F 8F ?? ?? | 4 |
| int imm8 | CD ?? | 2 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tabla de símbolos** | | | | |
| **Nombre** | **Segmento** | **Tamaño** | **Valor** | **Tipo** |
| X | Datos | Byte | 0000h | D |
| Y | Datos | Word | 0001h | D |
| z | Datos | Byte | 0003h | D |
| W | Word | Word | 0004h | D |
| Inicio | Código | Word | 0000h | D |
| Salir | Código | Word | 002Ah | D |
| U | Código | ? | 00 1Fh | U |
| V | Código | ? | 00 21h | U |

Realizando ensamblado de una pasada del código fuente:

1. (1 pts) Complete los valores del contador de localidades (LC) por cada línea del código fuente.
2. (2 pts) Complete la columna del código objeto con el código que se genera por cada una de las instrucciones ensambladas, resolviendo todas las referencias hacia delante.
3. (2 pts) Llene la tabla de símbolos que se genera durante el ensamblado del código.
4. (2 pts) ¿Cuál es el contenido del archivo objeto, o si hay errores, qué errores se obtienen a la salida?

ERRROR:

Error de símbolo indefinido U

Error de símbolo indefinido V

7.- (12 pts) Desarrolle un programa en lenguaje ensamblador para arquitectura x86 que, dadas tres variables, num1, num2 y resultado:

- Si num1 es mayor o igual que num2, resultado = num1\*num1+num1.

- Si num1 es menor que num2, resultado = num2! = factorial(num2).

- Si num1 es mayor que 9, o num2 es mayor que 9, el programa finaliza.

**Nota:** definir **num1** y **num2** de tipo byte, y **resultado** de tipo word. Los valores de num1 y de num2 se deben definir desde el código.

Considerar **num1** y **num2** como enteros sin signo.

Se tomará en cuenta:

* Que el programa no tenga errores.
* La lógica del programa (flujos).
* Que el programa se ejecute correctamente y realice lo solicitado.

8.- Se tiene el siguiente código:

title "EyPC - Segundo Examen Parcial" ;directiva title opcional

.model small ;directiva para especificar segmento de datos y codigo de 64 KB cada uno

.stack 64 ;define segmento de pila en bytes

.data ;segmento de datos

x db 11,0,13,60,48,26,7 ;arreglo ‘x’ de tipo byte que contiene datos de entrada

a dw 0001h ;variable ‘a’ de tipo word que almacena un dato de salida.

b dw 0 ;variable ‘b’ de tipo word que almacena un dato de salida.

dos db 2d ;variable ‘dos’ auxiliar para realizar operaciones con el valor 2.

.code ;segmento de código

inicio: ;etiqueta ‘inicio’

mov ax,@data ;carga @data en AX para especificar segmento de datos del programa

mov ds,ax ;DS = AX = @data

mov di,0 ; asigna al registro di el valor de 0

mov cx,7 ; asigna al registro dx el valor de 7

loop1: ; Crea una etiqueta llamada loop1

mov al,[x+di] ; Asigna al, el valor de un x + di dirección, es decir, lo va recorriendo

xor ah, ah ; Limpia el registro ah

div [dos] ; Divide siendo [dos] el divisor

cmp ah,0 ; Hace una comparación con con 0 con ah

jz et2 ; Condicional si es igual a cero se va a la etiqueta et2

et1: ; crea una etiqueta et1

mov al,[x+di] ; Asigna al, el valor de un x + di dirección, es decir, lo va recorriendo con el valor di

xor ah, ah ; limpia el registro ah

mul [a] ; Hace una multuplicacion siendo, “al” sera el Multiplicando,[a]l mi multiplicador

mov [a],ax ;asigna [a] con el valor de ax

inc di ; incrementa di

loop loop1 ;Instruccion de loop compara que cx sea diferente de 0, sino Vuelve la etiqueta loop1

jmp salir ; Hace un salto a la etiqueta salir

et2: ; crea una etiqueta et2

mov al,[x+di] ; Asigna al, el valor de un x + di dirección, es decir, lo va recorriendo con el valor di

xor ah, ah ; impia el registro ah

add [b],ax ; hace una suma con los valores [b] y ax

inc di ; Hace un incremente en 1

loop loop1 ; Instruccion de loop compara que cx sea diferente de 0, sino Vuelve la etiqueta loop1

salir: ;etiqueta salir

mov ax,4C00h ;AH = 4Ch, opcion 4C para int 21h, AL = 00, exit Code

int 21h ;ejecuta interrupción 21h

end inicio ;fin de programa

a) (1 pt) Agregue comentarios al código anterior.

b) (1 pt) Represente el mapa de memoria de datos al inicio del programa (con datos hexadecimales).

|  |  |  |
| --- | --- | --- |
| DS:000Bh |  | dos |
| DS:000Ah |  |  |
| DS:0009h |  | b |
| DS:0008h |  |  |
| DS:0007h |  | a |
| DS:0006h |  |  |
| DS:0005h |  |  |
| DS:0004h |  |  |
| DS:0003h |  |  |
| DS:0002h |  |  |
| DS:0001h |  |  |
| DS:0000h |  | x |

c) (1 pts) Represente el mapa de memoria de datos al final del programa (con datos hexadecimales).

|  |  |  |
| --- | --- | --- |
| DS:000Bh |  | dos |
| DS:000Ah |  |  |
| DS:0009h |  | b |
| DS:0008h |  |  |
| DS:0007h |  | a |
| DS:0006h |  |  |
| DS:0005h |  |  |
| DS:0004h |  |  |
| DS:0003h |  |  |
| DS:0002h |  |  |
| DS:0001h |  |  |
| DS:0000h |  | x |

d) (2 pts) En pocas palabras, ¿qué hace el programa?