ESTRUCTURA DE COMPUTADORES

GRADO INGENIERIA INFORMATICA (2017 – 2018)

UNIVERSIDAD DE GRANADA

Ejercicios saludo.s:

Ejercicio 1: *¿Qué contiene edx tras ejecutar mov logsaludo, %edx? ¿Para que necesitamos esa instrucción, o ese valor?*

Contiene el tamaño en bytes de la cabena saludo. Cada carácter es un byte por lo que longsaludo, es un int con un valor de 28. Se usa para acabar la ejecución del programa si el valor es erroneo.

Ejercicio 2: *¿Qué contiene ecx tras ejecutar mov $saludo, %ecx?*

Contiene el string, mov $saludo, %ecx es equivalente a mov $0x8049097, %ecx. He mirado el contenido haciendo x/32cb &ecx y aparece en cada posición el carácter que guarda.

Ejercicio 3: *¿Qué sucede si se elimina el simbolo de dato inmeditato de la instrucción anterior?*

Ahora lo que se produce es un error ya que no sabe que es saludo y creo que rellena el registro de basura, por eso el programa falla.

Nota: Al principio me dio un error luego repasando los ejercicios no me dio y me aparecía el mensaje exactamente igual. No me ha repercutido poner o quitar el simbolo $.

Ejercicio 4: *¿Cuantas posiciones de memoria ocupa la variable longsaludo? ¿Y saludo? ¿Cuantos bytes ocupa la sección de datos?*

longsaludo ocupa exactamente lo que ocuparía guardaría un int es decir una palabra, dos bytes, sin embargo saludo guarda 28 caracteres donde cada carácter son 4 bytes por lo que necesitamos 28 x 4 = 112 bytes que ocuparía 56 palabras.

Ejercicio 5: *Añadir dos volcados Data→Memory de la variable longsaludo, uno como entero hexadecimal, y otro como 4 bytes hex. ¿Que direcciones de memoria ocupa logsaludo? ¿Cuál byte está en la primera posición el más o menos signicativo? ¿Los procesadores de la linea x86 usa el criterio del extremo mayor o menor? Razonalo.*

0x82490b3 0x0000001c	0x82490b3 0x1c 0x00 0x00 0x00

Antonio Miguel Morillo Chica
77379021-Y

Solo ocupa una dirección de memoria, una palabra. El menos significtivo está a la derecha y al izquierda el que menos.

Ejercicio 6: *¿Cuantas posiciones de memoria ocupa la instrucción mov $1, %ebx? ¿Cómo se ha obtenido esa información?*

Ocupa 5 posiciones de memoria. Lo he optenido mirando la orden dump desde la instrucción mov $1, %ebx a la siguiente y mirando la parte izquierda que indica la posición:

```
0x08048079  <+5>:    mov   $0x1          , %ebx
0x0804807e  <+10>:   mov   $0x8049097, %ecx
```

Ejercicio 7: *¿Que sucede si se elimina del programa la primera instrucción int 0x80? ¿Y si se elimina la segunda? Razona.*

Si se elimina la primera no existe filtro de error con el valor de logsaludo por lo que aceptaría valores correctos. Si sequita el segundo se anula la opción de que el programa dinalice una vez terminado con éxito.

Ejercicio 8: *¿Cuál es el número de la llamada al sistema READ (en kernel Linux 32bits? ¿De dónde se ha obtenido esa información?*

Ejercicios media.s

Ejercicio 1: *Rellenando la lista al valor -1, la media es -1. Cambiando un elemento a 0, la media pasa a valer 0. ¿ Por qué? Consultar el manual de Intel sobre la instrucción de división. ¿Cuánto vale el resto de la división en ambos casos? Probarlo con ddd.*

Esto es debido a que div e idiv hacen truncamiento, en caso de ser -1, -1, 0, … etc, en cuanto uno de los valores sea mayor o igual a 0 el programa mostrará un 0 ya que se dividirá un número menor entre otro mayor. Ejem.: -3 / 4 ; Cociente: 0 Resto: -3

Ejercicio 2: *También se obtiene cociente 0 si se cambia lista[0]=1, o si lista[0]=2, ó si lista[0]=3... Comprobarlo con ddd. La siguiente pregunta lógica es hasta cuántos se puede incrementar lista[0] sin que cambie cociente=0.*

Se podrá hacer hasta que la media de los ultimos n-1 terminos sea mayor al termino lista[0] que introducimos. Es basicamente por la respuesta anterior, buscamos trucar la division para que el cociente sea 0.

Ejercicio 3: *¿Para qué rango de valores de lista[0] se obtiene media 1? ¿Cuánto vale el resto en ese rango? Comprobarlo Con ddd, y notar que tanto los dividendos como los restos son positivos (el cociente se redondea hacia cero).*

Antonio Miguel Morillo Chica
77379021-Y

Aparentemente en mi batería de pruebas ninguno pero lo he estado pensando y serían para todos aquellos cuya suma sea igual al tamaño de la lista o, en su defecto, mayor sin que supere al doble del tam de la lista. El resto serán valores comprendidos entre [0,tam_lista-1].

Ejercicio 4: *¿Para qué rango de valores de lista[0] se obtiene media -1? ¿Cuánto vale el resto en ese rango? Comprobarlo con ddd, y notar que tanto los dividendos como los restos son negativos (el cociente se redondea hacia cero).*

Para los valores: [1,2,-3,-4], [-1,-1,-1,-1], [0,-2,-1,-1], [-2,-2,-1,-1] y los restos suelen ser entre [-31,0]. Efectivamente se cumple lo que se ha dicho.

Batería de pruebas suma <u>sin</u> signo:

[1,1...]	suma = 32 = 20 hex
[2,2...]	suma = 64 = 40 hex
[1,2,3,4,1,...]	suma = 80 = 50 hex
[0xffffffff,0xffffffff,...] o [-1,-1,...]	suma = 137438953440 = 1fffffffe0 hex
[0x08000000,0x08000000,0x08000000,...]	suma = 4294967296 = 100000000 hex
[0x100000000,0x200..,0x400..,0x800..,0x100..]	suma = 32212254720 = 780000000 hex

Batería de pruebas suma <u>con</u> signo:

[-1,...] o [0xFFFFFFFF,...]	suma = -32 = ffffffffffffffe0 hex
[1,-2,1,-2,...]	suma = -16 = fffffffffffffff0 hex
[1,2,-3,-4,1,...]	suma = -32 = ffffffffffffffe0 hex
[0x7FFFFFFF,0x7FFFFFFF,0x7FFFFFFF,...]	suma = 68719476704 = fffffffe0 hex
[0x80000000,0x80000000,0x80000000,...]	suma = -68719476736 = fffffff000000000 hex
[0X04000000,0X04000000,0X04000000,...]	suma = 2147483648 = 80000000 hex
[0x08000000,0x08000000,0x08000000,...]	suma = 4294967296 = 100000000 hex
[0xFC000000,0xFC000000,0xFC000000,...]	suma = -2147483648 = ffffffff80000000 hex
[0xF8000000,0xF8000000,0xF8000000,...]	suma = -4294967296 = ffffffff00000000 hex
[0xF0000000,0xE0...,0xE0...,0xD0...,0xF0...]	suma = -17179869184 = fffffffc00000000 hex

Batería de pruebas media:

[1,-2,1,-2,...]	media = 0 = 0 hex resto = -16 = fffffff0 hex
[1,2,-3,-4,1,...]	media = -1 = ffffffff hex resto = 0 = 0 hex
[0x7FFFFFFF,...]	media = 2147483647 = 7fffffff hex resto = 0 = 0 hex
[0x80000000,...]	media = -2147483648 = 80000000 hex resto = 0 = 0 hex
[0xF0...,0xE0...,0XE0...,0XD0...,...]	media = -536870912 = e0000000 hex

Antonio Miguel Morillo Chica
77379021-Y

	resto = 0 = 0 hex
[-1,...] o [0xFFFFFFFF,...]	media = -1 = ffffffff hex resto = 0 = 0 hex
[0,-1,-1,-1,...]	media = 0 = 0 hex resto = -24 = ffffffe8 hex
[0,-2,-1,-1,...]	media = -1 = ffffffff hex resto = 0 = 0 hex
[1,-2,-1,-1,...]	media = 0 = 0 hex resto = -24 = ffffffe8 hex
[-2,-2,-1,-1,...]	media = -1 = ffffffff hex resto = -8 = fffffff8 hex
[2,-2,-1,-1,...]	media = 0 = 0 hex resto = -16 = fffffff0 hex
[3,-2,-1,-1,...]	media = 0 = 0 hex resto = -8 = fffffff8 hex
[32,-2,-1,-1,...]	media = 7 = 7 hex resto = 0 = 0 hex
[33,-2,-1,-1]	media = 7 = 7 hex resto = 8 = 8 hex
[34,-2,-1,-1,...]	media = 7 = 7 hex resto = 16 = 10 hex
[35,-2,-1,-1,....]	media = 7 = 7 hex resto = 24 = 18 hex
[63,-2,-1,-1,...]	media = 14 = e hex resto = 24 = 18 hex
[64,-2,-1,-1,...]	media = 15 = f hex resto = 0 = 0 hex
[95,-2,-1,-1,..]	media = 22 = 16 hex resto = 24 = 18 hex
[-31,-2,-1,-1,..]	media = -8 = fffffff8 hex resto = -24 = ffffffe8 hex

Antonio Miguel Morillo Chica
77379021-Y