Estructura de computadores:

Practica 2

Grupo 50

Javier Bautista Rosell
100315139
100315139@alumnos.uc3m.es

Alejandro Blanco Samaniego
100315058
100315058@alumnos.uc3m.es

Indice

Importante: las pruebas están en links de Gyazo, ya que se ven mal en pdf, en caso de que no funcione ninguno de los links, copiar la dirección que aparece justo debajo de la explicación de lo que hace la prueba.

Ejercicio 1:

Respondemos a las cuestiones planteadas en este ejercicio.

a) 0x00400024
b) `400018  volver a jal`
c) el 61 que corresponde a la "a"
d) 10010037
e) 10e
f)

G)
la $a0, cadena
lw $t0, 3($a0)
move $a0 $t0
li $v0 1
syscall

Ejercicio 2:

En este ejercicio hemos creado un programa en lenguaje ensamblador que devuelve la posición en la que se encuentra un carácter seleccionado dentro de un string

Desde el main invocamos a la función, donde iniciamos un contador. Hacemos un bucle que localice el carácter dentro del string, y en caso de que esté salta a la zona donde imprime su posición. En caso contrario salta a imprimir -1, donde devolvemos este valor si no se encuentra el carácter en la cadena. Retornamos luego al main, donde concluye el programa.

Batería de pruebas:

Prueba1: Mostramos la solución como en el ejemplo
http://gyazo.com/544b6a6470ca94578379511d2e6e1819

Prueba2 : Mostramos que también es válido para buscar espacio y otros caracteres
http://gyazo.com/cbd7b3bc86dbefdc295d8c36cb90d190

Prueba3 : Mostramos que si el carácter no está en la cadena, devuelve -1
http://gyazo.com/740ed06a7d29717e6633929f343e7778

Ejercicio 3

En este ejercicio hemos creado en lenguaje ensamblador un programa que lee cuantas posiciones tiene un string.

Primero generamos en el data un byte de espacio, para contar los espacios entre las letras del vector, y que estos no cuenten en el resultado del tamaño final. Desde el main, invocamos a la función, donde iniciamos contadores. Iniciamos un bucle que cuenta cuantos caracteres en total y cuantos espacios llevamos. Al final, restamos los espacios a los caracteres totales, para ver el tamaño del string, lo imprimimos por pantalla y retornamos al main, donde se termina el programa.

Batería de pruebas:

Prueba 1 : Mostramos la solución como en el ejemplo.
http://gyazo.com/16346ea73cdc9fd57b350ac6fc0c5c5e

Prueba 2 : Mostramos como la cantidad de espacios no varían el tamaño del string.
http://gyazo.com/532bdbb8174a120ac728842403f4a2c6

Prueba 3: Mostramos como esto funciona también con cadenas de otros tamaños.
http://gyazo.com/8a650de268076ddf2d9ea7397e8fa135

Ejercicio 4:

En este ejercicio hemos creado en lenguaje ensamblador un programa que realiza la media entre los enteros de un vector.

Desde el main, invocamos a la función, donde iniciamos contadores. Luego en el bucle sumamos cada entero del vector. Tras ello, pasamos el entero resultado de la suma y el entero del tamaño del vector a float, para poder realizar la división. Dividimos la suma entre el tamaño del vector e imprimimos el resultado como float. Tras ello, retornamos al main donde se termina el programa.

Batería de pruebas:

Prueba1: Mostramos la solución como en el ejemplo
http://gyazo.com/b18867403a88ac706ed056a451f08f3a

Prueba2: Mostramos que también es válido para otros números en el vector
http://gyazo.com/317d54699f0f996f2f80f84641faf476

Prueba3: Mostramos que también es válido para otros tamaños de vectores.
http://gyazo.com/8f4ab2905ec20b2be481255af8a11b38

Ejercicio 5

En este ejercicio hemos creado en lenguaje ensamblador un programa que lee un vector de enteros e imprime por pantalla primero el menor y luego el mayor.

Primero generamos en el data un salto de línea, para que en la consola aparezca el resultado del mismo esquema que el ejemplo, el menor arriba y el mayor abajo.

Desde el main, invocamos a la función, donde iniciamos contadores. Cargamos en un temporal el primer número del vector, el cual llamamos mínimo y máximo a la vez. En el bucle, recorremos el vector en busca de un número menor o mayor, y lo actualizamos en sus respectivas funciones. Luego, comparamos con los nuevos valores de mínimo y máximo. Al terminar de recorrer el vector, imprimimos primero el mínimo, luego el salto de línea y por último, el máximo. Retornamos luego al main, donde termina el programa.

Batería de pruebas:

Prueba 1: Mostramos la solución como en el ejemplo.
http://gyazo.com/89f5be1e76654dc878aa07282828b1dd

Prueba 2: Mostramos como el programa funciona también para otros tamaños de vector y diferentes enteros.
http://gyazo.com/3dbdd72cb846881b932169262c56f6ee

Prueba 3: Mosteamos como el programa funciona también con valores repetidos.
http://gyazo.com/64251fa2130999fa28bb330941da0283

Ejercicio 6:

En este ejercicio hemos creado en lenguaje ensamblador un programa que encadena 2 strings, es decir, tras el primero, en un mismo vector, continúa con el segundo.

Desde el main, invocamos a la función, donde iniciamos contadores. Luego contamos el numero de posiciones que tiene el primer string, una vez terminado, contamos las posiciones que tiene el segundo. Tras ello, sumamos estas cantidades y usando el sbrk, reservamos memoria donde crear un nuevo vector, del tamaño de la suma de los tamaños de los otros dos, recién calculada. Tras ello, colocamos en orden primero los caracteres del string 1 y luego los del 2 en el nuevo vector. Por último, retornamos al main, donde imprimimos por pantalla el nuevo vector, antes de terminar el programa.

Prueba1: Mostramos la solución como en el ejemplo
http://gyazo.com/6d32a6cd0d07a9b60f8bb7d04167d2b3

Prueba2: Mostramos como el programa funciona correctamente para otros dos strings cualesquiera.
http://gyazo.com/3f261f67fa4ffcf2fed05b40de6c6bc6

Ejercicio 7

En este ejercicio hemos intentado crear un programa que ordene de menor a mayor los números de un vector de enteros.

Desde el main invocamos a la función, donde iniciamos contadores. Comenzamos el bucle que recorre el vector, rellenando en orden las posiciones del vector. De esta manera, tras recorrer el vector por primera vez, el menor número estará en la primera posición, en la segunda vuelta, lo recorreremos a partir de la segunda posición, y en esta metemos el segundo menor…

En el bucle min, buscamos el menor entero del vector y lo actualizamos. Luego, en buscar_posición, buscamos en qué posición del vector se encuentra este mínimo. En mover_menor intercambiamos las posiciones del mínimo y el entero que estuviese en la posición a ordenar, para no perder su valor. Luego, actualizamos contadores. Por último, llamamos a la función principal y desde ahí imprimimos el vector resultante.

Desafortunadamente, el programa no consigue pasar correctamente estos valores. La impresión por lo tanto sale irregular.

Batería de pruebas:

Prueba 1: Aquí creemos que está el fallo.
http://gyazo.com/06fae6bd200906601e4b091f9cf95f9c

Prueba 2: Aquí mostramos el resultado que imprime.
http://gyazo.com/ed610760525418d8e4062ba75941c184