Ministerio del Poder Popular para la Educación Universitaria
Universidad de Carabobo
Facultad de Ciencia y Tecnología

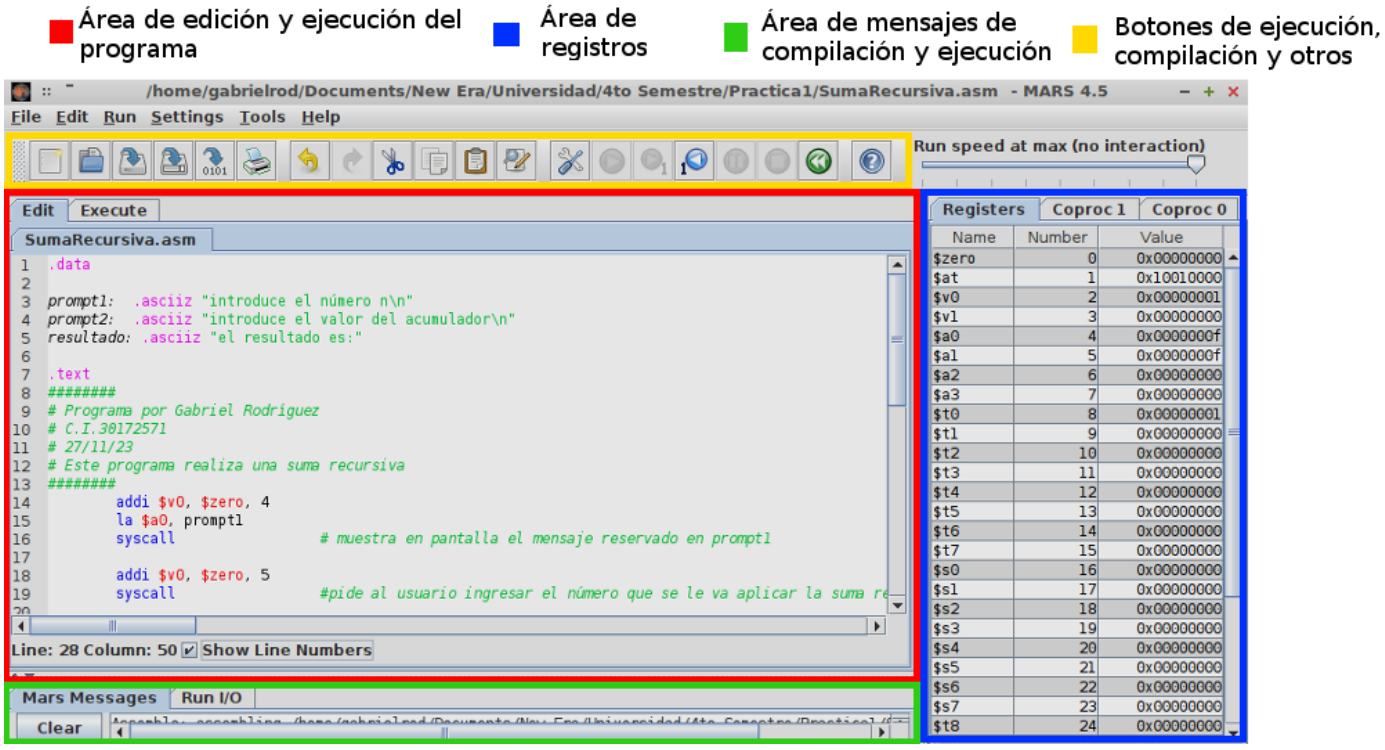
Informe de Mars

Por: Gabriel Isaac Rodríguez Carvallo
C.I. 30172571
Asignación: Arquitectura del Computador
Carrera: Computación

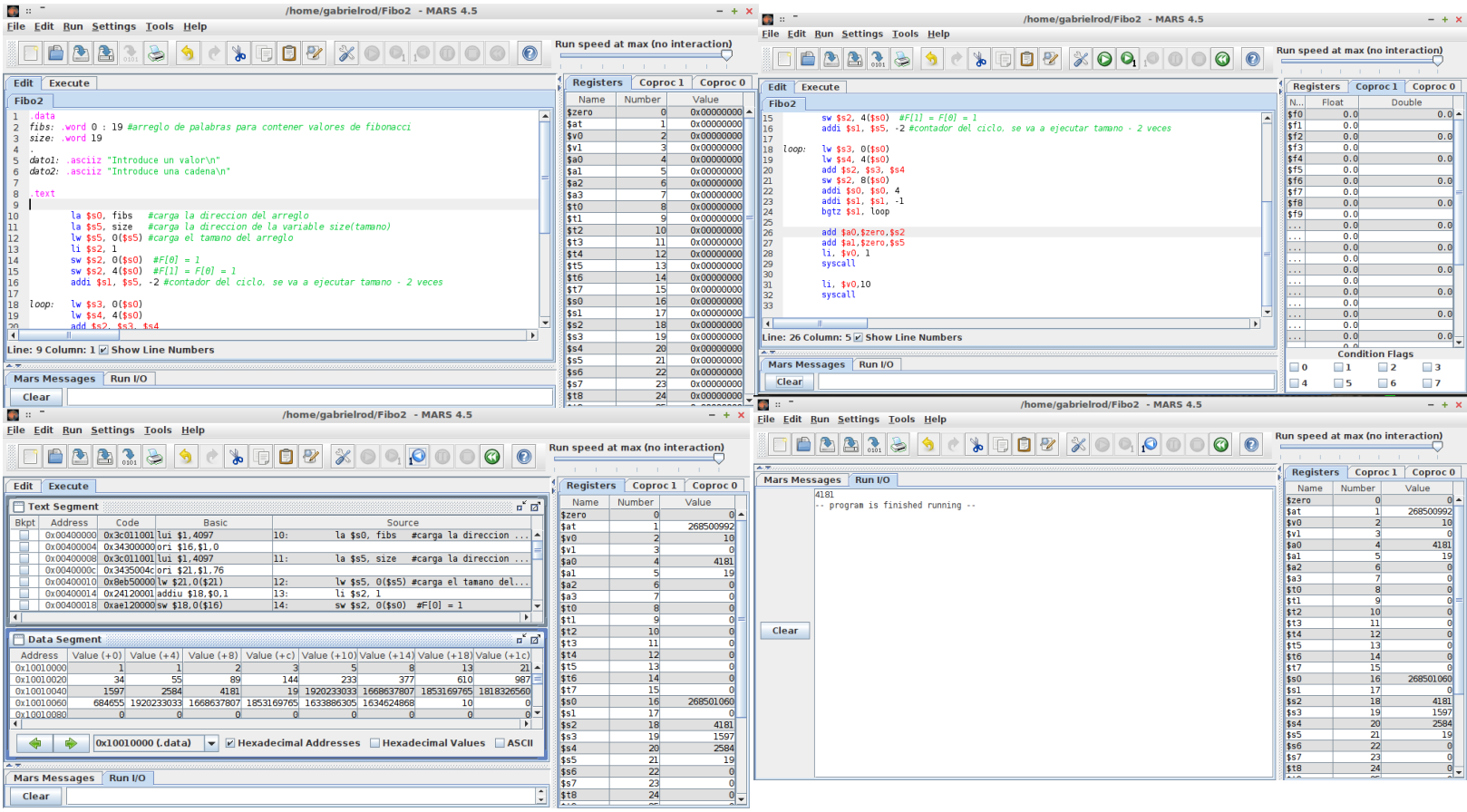
Acción #1, configuración del setup de Mars:

Ejecutar Mars resultó ser simple en mi sistema operativo(LXLE Eclectica 16.04.3 64-bit Edition(Ubuntu)), dado que tengo una versión(OpenJDK Java 8 Runtime) del JRE(Java Runtime Enviroment) que funciona con la versión de Mars(v4.5) que descargue. Al ejecutarla la pantalla se divide en varias áreas, la principal(en rojo en la imagen) tiene dos pestañas, la de edición(Edit), qué es donde aparece el código despues de crear o abrir un nuevo archivo .asm y la pestaña de ejecución(Execute), que contiene dos segmentos, el segmento de texto(Text Segment) muestra información respecto a las instrucciones que fueron, estan siendo o van a ser ejecutadas, como las direcciónes de dichas instrucciones o la fuente en el código de dicha instrucción. Por otro lado el segmento de datos(Data Segment) contiene la información en direcciones de memoria que podrian ser utilizadas al momento, como por ejemplo, el contenido de un vector. En el área de la derecha(azul en la imagen) en la pestaña de registros(Registers) salen todos los registros principales con su contenido actual, otras pestañas en esta área muestran el contenido de registros de números de punto flotante, banderas de condición y otros registros particulares. El área inferior izquierda(en verde en la imagen) también contiene dos pestañas, una de ellas contiene la salida en pantalla y la entrada por teclado(durante la ejecución de un programa) cuando esta es necesaria, esta pestaña se llama Run I/O(Input/Output), y la otra, llamada mensajes de Mars(Mars messages) contiene el estado de la compilación y ejecución del programa. Los botones en el área superior(en amarillo en la imagen) permiten compilar y ejecutar el programa que se esté creando, adicionalmente se puede ejecutar paso por paso, resetear la memoria los registros y se encuentra un boton de ayuda que contiene información util acerca de instrucciones y registros. Otros botones aquí sirven para guardar el programa, deshacer, copiar, pegar, entre otros.

En cuanto a la configuración del ambiente de trabajo el único cambio que realicé fue quitar el popup de guía de instrucciones porque resultaba molesto al escribir código y en algunos casos ocupaba mucho espacio. Para remover este popup, se selecciona Settings en el menu principal(ubicado arriba de el área amarilla) seguidamente se le da click en Editor…, y en la ventana nueva aparece en la parte inferior derecha para desactivar el popup, posteriormente se le da click a Apply and Close. Cabe destacar que Settings permite hacer otras configuraciones.



Las imagenes abajo contienen un código que imprime en pantalla el valor de un indice en la secuencia fibonacci(en este caso el decimo noveno), además de la ejecución de dicho código.



Acción #2, comparación de los tres ejemplos de Mars con el libro de David A. Patterson:

Los ejemplos de Mars de la página tienden a usar instrucciones como la(load address) y li(load inmediate), las cuales no se usan en el libro de David A. Patterson, estas permiten usar punteros y cargar valores en una variable sin usar el registro $zero, y otras instrucciones como mflo. Además los ejemplos de Mars también contienen secciones importantes, como la sección de .data que contiene información respecto a los vectores que seran utilizados, el tamaño de dichos vectores, cadenas de texto a ser utilizadas, entre otros datos importantes. El código de Mars tampoco guarda los registros previos en la pila($sp), esto debido a que los ejemplos son programas como tal y no funciones dentro de un programa. En otras palabras, algunos códigos que estan en el libro de David A. Patterson estan hechos para ser usados como funciones dentro de un programa existente. Estos tres ejemplos contienen varias llamadas al sistema(syscall) que son utilizadas para leer datos ingresados por el usuario, para imprimir resultados en pantalla, entre otros usos. Dichas llamadas al sistema no estan presentes en los ejemplos que aparecen en el libro de David A. Patterson.

Acción #3, suma recursiva en Mars:

El código de suma recursiva ameritó ligeras modificaciones en el código y los registros para que permitiera entrada y salida por pantalla usando llamadas al sistema. Nota: una parte del programa pide al usuario ingresar el número acumulador(acum), esto es debido a que la función tiene dos parámetros, el número “n” y el acumulador(acum). Facilmente se podría hacer una modificación al programa para que solo pida el número “n” e inicialice el acumulador con 0. Se llama SumaRecursiva.asm en el repositorio.

Acción #4, algoritmo de ordenamiento:

El algoritmo de ordenamiento que decidi programar fue el ordenamiento burbuja bidireccional(también conocido como CocktailSort). Mediante varias instrucciones que se encargan de ordenar un vector(de un tamaño entre 1 y 100) de números enteros(aunque facilmente podría ser modificado para recibir caracteres o números punto flotante). También usa entrada y salida por pantalla. Se llama CocktailSort.asm en el repositorio.