Universidad Nacional
de General Sarmiento

TRABAJO PRÁCTICO
-ARM-

GRUPO 9 – COMISIÓN 4

ORGANIZACIÓN DEL COMPUTADOR

PROFESORES:

VELCIC FERNANDO

VARGAS MARIANO

ALUMNOS:

Noelia Soledad Díaz
DNI 36.022.286
cotillonjujon@gmail.com

Maximiliano Massa
DNI 25.769.579
maxi_massa@hotmail.com

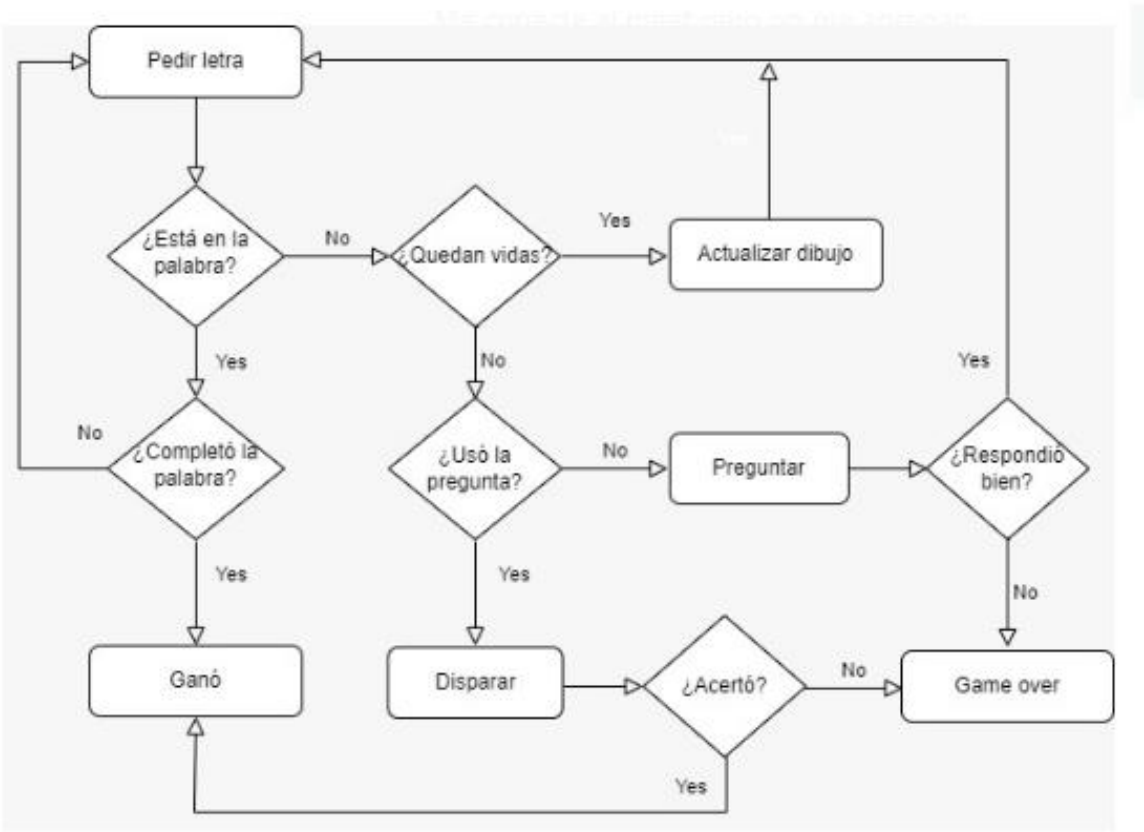
Julián Kidjekouchian
DNI 30.467.630
juliankidje@hotmail.com

TRABAJO PRACTICO FINAL – ARM

Organización del computador

Introducción:

Para comenzar a desarrollar el juego, cada integrante planteó por separado diferentes alternativas, en pseudocódigo, luego nos juntamos varias veces para procesar toda la información. Hicimos el siguiente diagrama de flujo alternativo para tener un primer panorama de como iniciar.



Una de las primeras decisiones que tuvimos, fue manejarnos con subrutinas independientes, y llamarlas desde un menú principal, para no tener problemas con los registros, y que el main quede lo mas resumido posible respetando las buenas prácticas de la programación

También tuvimos la precaución de mandar a pila los registros usados para no sobrescribirlos.
Hicimos un manejo responsable de los registros, tratando, en lo posible, de guardar en memoria la información.

Un inconveniente que tuvimos fue la implementación de la subrutina numeroRandom que investigando encontramos un código el cual nos entregaba un numero random muy largo.

Resolvimos este problema, usando instrucciones lógicas para rotar el numero y quedarnos con la información que necesitábamos.

También nos ocurrió que cuando queríamos acceder a la respuesta numérica del usuario (en la pregunta aproximativa), quedaban grabados los caracteres ingresados más el salto de línea (de cuando el usuario presiona enter), y luego para procesar esos caracteres y pasarlos de ascii a decimal, tuvimos que tener en cuenta ese salto de línea, antes de hacer la conversión.

Otra decisión que fuimos tomando a lo largo del proceso fue hacer subrutinas reutilizables, para no tener código repetitivo.

Comenzamos trabajando desde lo más simple, desarrollando y complejizando de a poco.
Primero hicimos una subrutina que muestre por pantalla al usuario un mensaje indicándole que deber ingresar una letra. Luego otra subrutina que lea por teclado dicha letra.

SeleccionarPalabra:
Recibe un numero random y extrae de una lista de palabras aquella con la que vamos a trabajar.

Verificador:
El siguiente paso fue desarrollar una subrutina más compleja (llamada “verificador”), que verifica la letra ingresada, la busca en la palabra elegida, y si la encuentra reemplaza a esta letra por el guion que está en el mapa, caso contrario resta una vida y dibuja la correspondiente parte del cuerpo.

DibujarCuerpo:
Si perdió una vida esta subrutina se encarga de actualizar el mapa dibujando una parte del cuerpo.

Reemplazar guion:
Dentro de la subrutina “verificador” , hicimos otra llamada “reemplazaGuión”, acá tuvimos el inconveniente ya que para hacer esto, era necesario obtener un algoritmo que calcule las posiciones de los guiones en el mapa. Lo solucionamos recorriendo la palabraelegida, y en caso de haber coincidencia, utilizamos esa posición para reemplazar el guion.

Restarvidas:
En el caso que no haya coincidencia, ejecuta la subrutina restarVidas que va restando de 1 vida por cada error, hasta que le quede 1, en ese caso, realiza una pregunta para darle una vida extra.

Ultima Oportunidad:
En la subrutina ultimaOportunidad se hace el llamado a otra subrutina que se llama preguntaAprox, la cual utiliza el mismo valor de random para elegir que pregunta realizar.

EvaluarRespuesta:
En el caso que la respuesta se encuentre dentro del rango aceptable se da una vida extra, de lo contrario, se ejecutara la siguiente subrutina “disparo”

Disparo:
Disparo es una subrutina que le da la posibilidad de salvar al ahorcado con su puntería. En esta, el usuario ingresa una coordenada x e y, que será evaluada, y en caso de coincidir con la correspondiente coordenada de la cuerda en el mapa, el jugador gana el juego.

ActualizarEstado:
Finalmente esta subrutina actualiza el mapa informando que el juego termino y su resultado.

Conclusión:

pudimos poner en práctica todos los conceptos aprendidos durante la cursada. Creemos que para programar en lenguaje ensamblador Arm, hay que tener en cuenta varios detalles, administrar los registros, trabajar con memoria, y demás recursos limitados a los cuales no estamos acostumbrados al programar en lenguaje de alto nivel.
Gracias a este trabajo, pudimos adquirir consciencia de cómo se programa en un lenguaje de bajo nivel. Por ejemplo, cuando trabajamos con java, tenemos recursos como variables, estructuras de datos, ciclos, y otras herramientas que facilitan la tarea del programador. En cambio, en ensamblador, al no tener todas esas herramientas, las tenemos que pensar y programar en un nivel mas bajo de abstracción.
Trabajar en ensamblador da como resultado un código mas extenso, y difícil de leer, pero nos ayuda a entender la organización del computador.

El trabajo se encuentra en la carpeta:

occ4g9@Alysa:~/tp_grupo9 $
donde se encuentra el archivo ejecutable **grupo9orga** , y el archivo grupo9orga.asm.