

Universidad Autónoma de Baja California
Facultad de Ciencias Químicas e Ingeniería



Alumno: Eduardo Marcelo Gutiérrez Soto
Profesora: Evangelina Lara Camacho
Practica 10 Interfaces entre lenguaje C con ensamblador

Teoría

Responda las siguientes preguntas.

- Que significa la palabra reservada Extern en Lenguaje C.

Nos sirve para poder declarar variables externas al bloque de código en el que se trabaja actualmente, nos sirve para acceder a variables o funciones fuera del mismo código utilizando la palabra extern, ya que las funciones son consideradas externas en C por defecto y normalmente no se necesitan ser declaradas, en pocas palabras la directiva extern nos sirve para acceder a funciones o variables locales que se encuentran en otro archivo ajeno al nuestro.

- Que significa la palabra reservada Public en Lenguaje Ensamblador.

Nos sirve para concatenar los segmentos a otros con el mismo nombre en la fase del linked y para acceder a funciones que se encuentran en otros módulos por separado.

Al realizar una interfaz entre código en lenguaje C y en lenguaje ensamblador.

- De que manera se pasan los parámetros a una función.

Para pasar los parámetros a una función en ensamblador son mediante la pila utilizando el registro BP ya que los parámetros son ingresados en la pila en un cierto orden y retirados también en un orden específico ya que en la pila también se almacenan otros datos de importancia, pero los parámetros se acceden de la siguiente manera, para el primer parámetro ser $[BP+4]$ ya que antes de ese parámetro no tendríamos más nada ya que se empiezan a enviar desde el primer parámetro más a la derecha hasta llegar con el más hacia la izquierda.

- Cuando se pasa más de un parámetro a una función, En que orden se envían los parámetros.

Siempre el primer parámetro más hacia la derecha sería el $[BP+4]$, si tenemos un segundo parámetro sería el $[BP+6]$, y si tenemos un tercer parámetro sería un $[BP+8]$, ya que los parámetros solo se ingresan a la pila de 2 bytes en 2 bytes y sigue ese orden.

- De que manera se retorna un valor menor o igual a 16 bits

Los valores se retornan en el registro de 16 bits el que es AX todos los tipos de valores que hay en C que pueden ser un unsigned char, char, unsigned short, short, unsigned int, int. Esos son los tipos de datos que retornan 16 bits o valores menores a el mismo.

- De que manera se retorna un valor mayor a 16 bits, pero menor o igual a 32 bits.

Los valores mayores a 16 bits pero menores o iguales a 32 bits se retornan mediante los registros DX-AX siendo en AX la parte menos significativa y en DX la parte más significativa, y los tipos de datos que se retornan mediante esos registros son los unsigned long, long, apuntadores cercanos y apuntadores lejanos. Esos son los tipos de dato que se pueden retornar en más de 16 bits y menor o igual a 32 bits.

1- Cree los programas myputc.asm y P10-1.c que contengan el código del Listado 1. Compile, ensamble y encadene mediante la línea de comando:

C:\OCLE>tcc -ms -f- P10-1.c myputc.asm Ejecute el archivo generado P10-1.exe, el cual desplegará el mensaje “

EJECUCIÓN:

```
C:\A>tcc -ms -f- P10-1.c myputs.asm
Turbo C Version 2.01 Copyright (c) 1987, 1988 Borland International
p10-1.c:
myputs.asm:
Turbo Assembler Version 2.01 Copyright (c) 1988, 1990 Borland International

Assembling file:    myputs.ASM
Error messages:     None
Warning messages:   None
Passes:             1
Remaining memory:   310k

Turbo Link Version 2.0 Copyright (c) 1987, 1988 Borland International

        Available memory 455288

C:\A>P10-1.EXE
Hola Mundo!!
```

2- Cree el programa ManipulacionCadenas.asm, el cual contiene las siguientes rutinas públicas implementadas en lenguaje Ensamblador, y que van a ser invocadas desde lenguaje C:

a) **my_erase**: borra una porción de una cadena. Recibe como parámetros un apuntador a una cadena, un apuntador a la posición inicial a borrar y un entero con la cantidad de caracteres a borrar a partir de esa posición. Si la cadena es más corta que los caracteres solicitados, el procedimiento borra todos los posibles.

Si la posición inicial está fuera del límite de la cadena, el procedimiento retorna un -1 en AX, en caso contrario retorna la cantidad de caracteres que pudo borrar. Su declaración externa en C es: `int my_erase(char * cadena, char * posicion, int cantidad)`.

EJECUCION:

```
Turbo Link Version 2.0 Copyright (c) 1987, 1988 Borland International

    Available memory 444476

C:\A>P10-2.EXE
Se borraron 4 caracteres y la cadena resultante es: Houndo!!!
C:\A>
```

Le insertamos el numero de veces que queremos que borrara caracteres y el resultado fue el siguiente el mismo que nos da al momento de la ejecución que el esperado en el ejemplo de la practica, al momento de insertar una posición mayor a que el tam de la cadena mandamos por ax el valor de -1 y se termina el programa.

```
res = my_erase(cad,cad+17,4);
```

Mandamos un valor mayor a los que hay en la cadena como el 17 ya que este supera el rango de la cadena y lo que hacemos al finan en el código ensamblador es mandar por ax un -1 que seria un 0ffh.

Caso propuesto por nosotros.

```
C:\A>P10-2.EXE
Se borraron 2 caracteres y la cadena resultante es: Hola Mdo!!!
Se copiaron 2 caracteres y la cadena resultante es: la
C:\A>_
```

Borramos dos letras mas adelante lo que son 6 espacios mas lo cual borramos un total de 2 letras las cuales fueron U y la N, y el recibir los parametros en ensamblador fue así.

```
mov bx,[bp+4]      ;indice de la cadena
mov si,[bp+6]      ;desde donde quiero borrar
mov cx,[bp+8]      ;num caracteres a borrar
```

b) **my_substr**: almacena en una cadena una copia de una porción de otra cadena. Recibe como parámetros un apuntador a una cadena fuente y una cadena destino, un apuntador a la posición inicial a copiar de la cadena fuente y la cantidad de caracteres a copiar a partir de esa posición. Si la cadena fuente es más corta que los caracteres solicitados, el procedimiento copia todos los posibles. Si la posición inicial está fuera del límite de la cadena, el procedimiento retorna un -1 en AX, en caso contrario retorna la cantidad de caracteres que pudo copiar

EJECUCIÓN:

ejemplo propuesto en la practica.

```
C:\A>P10-2.EXE
Se borrarón 4 caracteres y la cadena resultante es: Houndo!!!
Se copiaron 2 caracteres y la cadena resultante es: Ho
C:\A>_
```

Se miran los dos programas ya que deje en el mismo main las dos ejecuciones seguidas.

Ejemplo propuesto por nosotros.

```
C:\A>P10-2.EXE
Se borrarón 2 caracteres y la cadena resultante es: Hola Mdo!!!
Se copiaron 2 caracteres y la cadena resultante es: la
C:\A>_
```

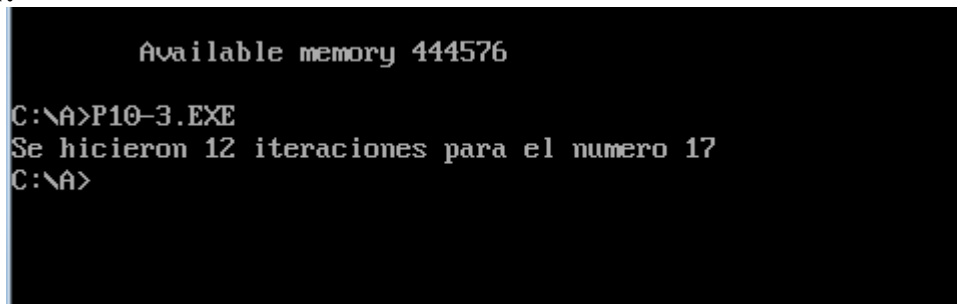
Decidimos copiar las letras que siguen despues del 2 elementos desplazados de la cadena original y para mas comodidad hice una cadena extra para que no se viera reflejada la modificación por el ejercicio anterior y copiamos las letras L y A.

el recibir los parametros en ensamblador fue de la siguiente manera

```
mov si,[bp+4]      ;fuente
mov bx,[bp+6]      ;destino
mov di,[bp+8]      ;posicion
mov cx,[bp+10]     ;n veces
```

3- Cree el programa ConjCollatz.asm, el cual contiene la rutina pública ConjeturaDeCollatz implementada en lenguaje Ensamblador, y que va a ser invocada desde lenguaje C. Esta rutina recibe como parámetro un número de 8 bits, denominado a, y retorna en AX el número de iteraciones necesarias para llegar al valor 1.

EJECUCIÓN:



```
Available memory 444576
C:\A>P10-3.EXE
Se hicieron 12 iteraciones para el numero 17
C:\A>
```

Conclusiones y dificultades.

Fue una practica muy entretenida y me gusto mucho como es el manejo de los datos con el lenguaje C y el lenguaje ensamblador, ya que podemos desarrollar tareas especificas que resultan mas fáciles en ensamblado y así ejecutarlas en lenguaje C haciendo el envío de parámetros de la manera que se debe hacer, mi única dificultad fue el ultimo ejercicio ya que no estaba contemplando un caso y por eso no me salia pero al final salio todo bien y lo logre realizar sin ningún problema.

Bibliografías.

Indefinido. (2017). La Conjetura de Collatz. 06/05/2020, de Youtube Sitio web: <https://www.youtube.com/watch?v=5gz-SKqg-do>