|  |  |  |
| --- | --- | --- |
| **Apellido y Nombres** | **Legajo** | **# de Hojas** |
|  |  |  |

# Parte Teórica

Dibuje conceptualmente el contenido del stack del programa en ejecución, si al correrlo con un debugger colocamos un *breakpoint* en la línea indicada. Asuma un sistema operativo de **32 bits**, y que al arrancar el programa se ha asignado la dirección de memoria **0xbfff0000** como comienzo del stack.

**#include <stdio.h>**

**int local (int a, int b);**

**int main (void) {**

**printf("%d\n",local(5,10));**

**return 0;**

**}**

**int local(int a, int b) {**

**int r;**

**r = a+b;**

**>>>> return r;**

**}**

# Parte Práctica

Se recibe en los argumentos de línea de comandos una lista de strings a procesar. La cantidad de argumentos recibidos es variable y no se conoce previamente. El nombre del programa debe ignorarse y en ningún momento forma parte de la lista de strings a procesar.

Se pide implementar lo siguiente:

**int contar\_strings(char \*\*s);**

Recibe un vector de punteros a char, terminado en NULL, y cuenta la cantidad de strings en dicha lista.

**int es\_numerico(char \* s);**

Devuelve (int)1 si el string especificado es puramente numérico (sólo contiene dígitos del 0 al 9 y/o el caracter punto), devuelve (int)0 en caso contrario.

**char \*\* copiar\_strings(char \*\* s);**

Copia el vector de punteros a strings a un nuevo espacio de memoria, cuyo comienzo es devuelto por la función al finalizar. “s” es el vector de punteros a strings a copiar, terminado en NULL. **Si el string en cuestión es puramente numérico, no hay que copiarlo al nuevo espacio de memoria.**

**void ordenar\_strings(char \*\* s);**

Ordena alfabéticamente el vector de punteros a strings, terminado en NULL.

**void imprimir\_strings(char \*\* s, char \*\* d);**

Al finalizar, el programa debe imprimir la lista de strings originales y los ordenados alfabéticamente (sin numéricos), incluyendo su posición en el vector. Tenga en cuenta que al ignorar los strings numéricos, la lista ordenada puede ser más corta que la original. Por ejemplo:

**./programa bbb ccc aaa 78 99 ddd**

Debe imprimir:

**[0] bbb aaa**

**[1] ccc bbb**

**[2] aaa ccc**

**[3] 78 ddd**

**[4] 99 -**

**[5] ddd -**

Notas:

* Recuerde que se debe verificar que por lo menos se le hayan pasado una cantidad mínima de strings en la línea de comandos.
* Toda la memoria asignada dinámicamente debe ser correctamente liberada antes de finalizar el programa.
* Pueden implementar archivos de prueba para usar redirección de la línea de comandos $(<prueba.txt)

# Condición mínima de aprobación

* Un total de 4 (cuatro) puntos bien implementados/resueltos, ya sea el punto teórico y 3 funciones; o bien 4 de las 5 funciones pedidas.
* Cualquier error de índole conceptual, segmentation fault o memory leak invalida la solución en cuestión.

# No forma parte del exámen

* No se pide reimplementar la rueda. **Utilice funciones de la librería estándar del C siempre que sea posible.**