|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Marco Antonio Martinez Quintana |
| *Asignatura:* | Estructura de Datos y Algoritmos I |
| *Grupo:* | 17 |
| *No de Práctica(s):* | 4 |
| *Integrante(s):* | Enzo Valdés Zavala |
| *No. de Equipo de cómputo empleado:* |  |
| *No. de Lista o Brigada:* |  |
| *Semestre:* | 2020-1 |
| *Fecha de entrega:* | 1 de marzo del 2020 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

**Objetivos:**

Utilizar las funciones malloc, calloc y realloc que nos permiten reservar y almacenar información de manera dinámica (en tiempo de ejecución).

**Introducción:**

La memoria dinámica se refiere al espacio de almacenamiento que se reserva en tiempo de ejecución, debido a que su tamaño puede variar durante la ejecución del programa.  
La memoria reservada de forma dinámica se aloja en el “heap” o almacenamiento libre y la memoria estática en el “stack” o pila (que es una zona muy limitada).

El heap va cambiando de tamaño durante la ejecución de un programa.  
Tambien llamado segmento de datos, es un medio de almacenamiento de mayor capacidad que la pila, y almacena datos durante toda la ejecución de las funciones.  
  
Para acceder a cualquier dato almacenado dentro, se utiliza un apuntador.  
  
Para redimensionar el heap, existen 4 funciones que nos ayudan a modificarlo:

1. Función malloc:

Permite reservar un bloque de memoria dinámica mediante una función void:

Void \*malloc(size\_t size);

1. Función calloc:

Funciona de manera similar a malloc, pero además de reservar memoria en tiempo real, inicializa esta en 0:

Void \*calloc(size\_t nelem, size\_t size);

(Donde nelem indica el numero de datos que se van a reservar).

1. Función realloc:

Permite redimensionar el espacio asignado previamente de forma dinámica, aumentando el tamaño de la memoria reservada:

Void \*realloc(void \*ptr, size\_t size);

(Donde ptr hace referencia al apuntador que va a redimensionar, y si este es 0 funciona como malloc).

1. Función free:

Libera la memoria almacenada de forma dinámica:

Void free(void\*ptr);

**Desarrollo:**

**Explicación:**

En esta practica, utilizamos las 4 funciones para almacenar y modificar la memoria de manera dinámica, una para cada función.

Función malloc:

Donde solamente reservamos memoria.

A screenshot of a social media post

Description automatically generatedA screenshot of a computer

Description automatically generated

Función calloc:

Donde inicializamos esta memoria en 0:

A screenshot of a social media post

Description automatically generatedA screenshot of a computer

Description automatically generated

Y finalmente, función realloc:

Donde definimos 2 apuntadores llamados arreglos que nos ayudan a redimensionar la memoria al llamar a realloc:

A screenshot of a computer

Description automatically generatedA screenshot of a social media post

Description automatically generatedA screenshot of a computer screen

Description automatically generated

**Conclusión:**

Se cumplio el objetivo de la practica al poder utilizar correctamente las diferentes funciones que reservan y manejan la memoria dinámica.

**Bibliografía:**

-El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.

-Teoria vista en el salón de clases.