	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): José Luis Arroyo Chavarría

No. de Equipo de cómputo empleado: 1

No. de Lista o Brigada: 5

Semestre: 2

Fecha de entrega: 01/03/2020

Observaciones:

CALIFICACIÓN: _____

Objetivo:

Aprender en C la forma de almacenar o guardar cualquier información con una memoria dinámica en un tiempo de ejecución.

Introducción:

La memoria dinámica es una memoria que no se sabe del número de la variable a considerarse, permite solicitar memoria un tiempo de ejecución, dependiendo de cuanta memoria se necesite. Se maneja la memoria con el uso de punteros.

Los tipos de datos se crean y se destruyen mientras se ejecuta el programa y por lo tanto la estructura de datos se va modificando a los requerimientos del programa, así evitando perder datos o desperdiciar memoria al momento de compilar el programa.

El sistema operativo divide el programa en cuatro partes que son: texto, datos (estáticos), pila y una zona libre o heap.

Heap es donde queda la memoria libre para poder utilizarla de forma dinámica.

También la pila cambia su tamaño dinámicamente, pero esto depende del sistema operativo.

Lenguaje C permite el almacenamiento de memoria en tiempo de ejecución a través de tres funciones: malloc, calloc y realloc.

Malloc:

Sirve para reservar un espacio de memoria tan grande como se especifica dentro de la función. Es imprescindible comprobar que dicho puntero no es nulo (NULL).

Calloc:

Sirve para reservar un espacio de memoria tan grande como se especifica dentro de la función y al mismo tiempo inicializa todos a 0.

Realloc:

Redimensiona un tipo de dato que asignamos con Malloc pero conservando sus valores.

NULL:

Cualquier función de reserva dinámica de memoria, devuelve un puntero nulo (**NULL**) si la reserva de memoria no puede realizarse, generalmente por falta de memoria disponible.

Free:

Durante la ejecución del programa puede ser interesante, e incluso necesario, proceder a liberar parte de la memoria reservada con anterioridad y que ya ha dejado de ser necesario tener reservada.

Esto puede realizarse mediante la función free() donde el puntero cuya zona de memoria asignada de forma dinámica queremos liberar.

Desarrollo y resultados:

- **Github**

https://github.com/WillyLuisPT3011/EDA-Practica-4

- **Código**

1. **Código (malloc)**

```
#include <stdio.h>
#include <stdlib.h>
int main (){
    int *arreglo, num, cont;
    printf("¿Cuantos elementos tiene el conjunto?\n");
    scanf("%d",&num);
    arreglo = (int *)malloc (num * sizeof(int));
    if (arreglo!=NULL) {
            printf("Vector reservado:\n\t[");
            for (cont=0 ; cont<num ; cont++){
                    printf("\t%d",*(arreglo+cont));
            }
            printf("\t]\n");
            printf("Se libera el espacio reservado.\n");
            free(arreglo);
    }
    return 0;
}
```

2. **Código (calloc)**

```
#include <stdio.h>
#include <stdlib.h>
int main (){
    int *arreglo, num, cont;
    printf("¿Cuantos elementos tiene el conjunto?\n");
    scanf("%d",&num);
    arreglo = (int *)calloc (num, sizeof(int));
    if (arreglo!=NULL) {
            printf("Vector reservado:\n\t[");
            for (cont=0 ; cont<num ; cont++){
                    printf("\t%d",*(arreglo+cont));
            }
            printf("\t]\n");
            printf("Se libera el espacio reservado.\n");
            free(arreglo);
    }
    return 0;
}
```

3. **Código (realloc)**

```
#include <stdio.h>
#include <stdlib.h>
int main (){
    int *arreglo, *arreglo2, num, cont;
    printf("¿Cuántos elementos tiene el conjunto?\n");
    scanf("%d",&num);
    arreglo = (int *)malloc (num * sizeof(int));
    if (arreglo!=NULL) {
            for (cont=0 ; cont < num ; cont++){
                    printf("Inserte el elemento %d del conjunto.\n",cont+1);
                    scanf("%d",(arreglo+cont));
            }
```

```
            printf("Vector insertado:\n\t[");
            for (cont=0 ; cont < num ; cont++){
                printf("\t%d",*(arreglo+cont));
            }
            printf("\t]\n");
            printf("Aumentando el tamaño del conjunto al doble.\n");
            num *= 2;
            arreglo2 = (int *)realloc (arreglo,num*sizeof(int));
            if (arreglo2 != NULL) {
                arreglo = arreglo2;
                for (; cont < num ; cont++){
                    printf("Inserte el elemento %d del conjunto.\n",cont+1);
                    scanf("%d",(arreglo2+cont));
                }
                printf("Vector insertado:\n\t[");
                for (cont=0 ; cont < num ; cont++){
                    printf("\t%d",*(arreglo2+cont));
                }
                printf("\t]\n");
            }
            free (arreglo);
        }
        return 0;
    }
```

- **Captura de pantalla**

```c
#include <stdio.h>
#include <stdlib.h>

int main (){
        int *arreglo, num, cont;
        printf("¿Cuantos elementos tiene el conjunto?\n");
        scanf("%d",&num);
        arreglo = (int *)malloc (num * sizeof(int));
        if (arreglo!=NULL){
                printf("Vector reservado:\n\t[");
                for (cont=0 ; cont<num ; cont++){
                        printf("\t%d",*(arreglo+cont));
                }
                printf("\t]\n");
                printf("Se libera el espacio reservado.\n");
                free(arreglo);
        }
        return 0;
}
```

```
Albania02:P3 edaI17alu05$ ./p3_1
¿Cuantos elementos tiene el conjunto?
4
Vector reservado:
        [       0       0       0       0       ]
Se libera el espacio reservado.
Albania02:P3 edaI17alu05$ ./p3_1
¿Cuantos elementos tiene el conjunto?
6
Vector reservado:
        [       0       0       0       0       0       0       ]
Se libera el espacio reservado.
```

```c
#include<stdio.h>
#include<stdlib.h>

int main(){
        int *arreglo, num, cont;
        printf("¿Cuantos elementos tiene el conjunto?\n");
        scanf("%d", &num);
        arreglo = (int *)calloc (num, sizeof(int));
        if(arreglo!=NULL){
                printf("Vector reservado:\n\t[");
                for(cont=0; cont<num; cont++){
                        printf("\t%d",*(arreglo+cont));
                }
                printf("\t]\n");
                printf("Se libera el espacio reservado.\n");
                free(arreglo);
        }
        return 0;
}
```

```
Albania02:P3 edaI17alu05$ ./p3_2
¿Cuantos elementos tiene el conjunto?
6
Vector reservado:
        [       0       0       0       0       0       0       ]
Se libera el espacio reservado.
```

```c
#include <stdio.h>
#include <stdlib.h>

int main (){
        int *arreglo, *arreglo2, num, cont;
        printf("¿Cuántos elementos tiene el conjunto?\n");
        scanf("%d",&num);
        arreglo = (int *)malloc (num * sizeof(int));
        if (arreglo!=NULL) {
                for (cont=0 ; cont < num ; cont++){
                        printf("Inserte el elemento %d del conjunto.\n",cont+1);
                        scanf("%d",(arreglo+cont));
                }
                printf("Vector insertado:\n\t[");
                for (cont=0 ; cont < num ; cont++){
                        printf("\t%d",*(arreglo+cont));
                }
                printf("\t]\n");
                printf("Aumentando el tamaño del conjunto al doble.\n");
                num *= 2;
                arreglo2 = (int *)realloc (arreglo,num*sizeof(int));
                if (arreglo2 != NULL){
                        arreglo = arreglo2;
                        for(; cont < num; cont++){
                                printf("Inserte el elemento %d del conjunto.\n",cont+1);
                                scanf("%d",(arreglo2+cont));
                        }
                        printf("Vector insertado:\n\t[");
                        for(cont=0; cont < num; cont++){
                                printf("\t%d", *(arreglo2+cont));
                        }
                        printf("\t]\n");
                }
                free(arreglo);
        }
        return 0;
}
```

```
Albania02:P3 edaI17alu05$ ./p4_3
¿Cuántos elementos tiene el conjunto?
5
Inserte el elemento 1 del conjunto.
6
Inserte el elemento 2 del conjunto.
3
Inserte el elemento 3 del conjunto.
9
Inserte el elemento 4 del conjunto.
12
Inserte el elemento 5 del conjunto.
15
Vector insertado:
        [       6       3       9       12      15      ]
Aumentando el tamaño del conjunto al doble.
Inserte el elemento 6 del conjunto.
5
Inserte el elemento 7 del conjunto.
10
Inserte el elemento 8 del conjunto.
15
Inserte el elemento 9 del conjunto.
20
Inserte el elemento 10 del conjunto.
25
Vector insertado:
        [       6       3       9       12      15      5       10      15      20      25      ]
```

Explicación:

Dependiendo de cuál función se utilice a realizar diferentes resultados. En el caso malloc se reservaran espacios, en el caso calloc se reservara pero también se liberaran iniciándolos a 0 y en el caso de realloc se tendrán varias dimensiones u otros tipos de información y reservandolos

Conclusión:

En lo personal este tema me parece muy complejo e interesante dependiendo de los distintos usos que se quiera utilizar para guardar distintos tipos de información. Siento que necesito más práctica para utilizarlo de manera correcta

Bibliografías y Cibergrafías:

- El lenguaje de programación C. Brian W. Kernighan, Dennis M. Ritchie, segunda edición, USA, Pearson Educación 1991.
- Ariel Rodríguez (2010). How knowing C and C++ can help you write better iPhone apps, part 1. [Figura 1]. Consulta: Enero de 2016. Disponible en: http://akosma.com/2010/10/11/how-knowing-c-and-c-can-help-you-write-betteriphone-apps-part-1/
- https://es.wikipedia.org/wiki/Memoria_dinámica_(programación)
- https://compilandoconocimiento.com/2016/12/24/memoria-dinamica/