	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: TISTA GARCÍA EDGAR

Asignatura: ESTRUCTURA DE DATOS Y ALGORITMOS I

Grupo: 1

No de Práctica(s): 04

Integrante(s): GÓMEZ LUNA ALEJANDRO

No. de Equipo de cómputo empleado 28

Semestre: 2019-2

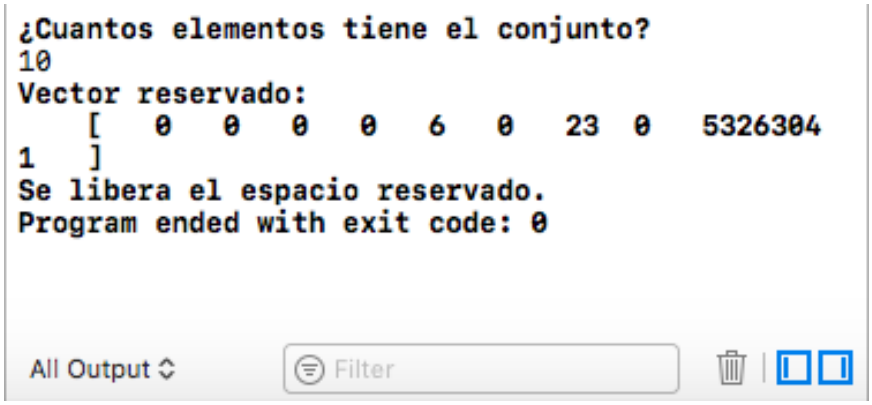
Fecha de entrega: 4/Marzo/2019

Obervaciones:

CALIFICACIÓN: _____

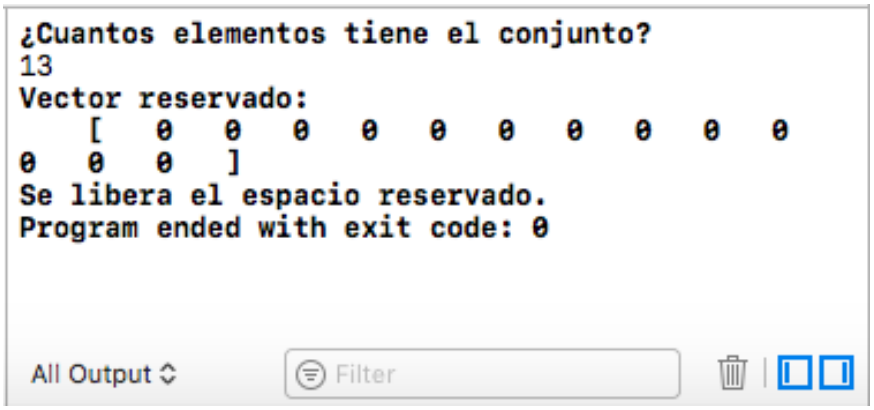
- Objetivo de la práctica
  Utilizarás funciones en lenguaje C que permite reservar y almacenar de manera dinámica (en tiempo de ejecucuón).

- Desarrollo
  Ejemplos de la guía
  1) Ejemplo free: El programa inicia preguntándole al usuario cuantos elementos tiene el conjunto. Dicho dato, se multiplica por el tamaño que ocupa un entero y se le pasa como parámetro a la función malloc que retorna un puntero void que indica la primera posición desde la cual se reservó el espacio solicitado y, este se convierte en uno de tipo entero y se guarda en una variable, la cual se puede visualizar como si fuera un arreglo unidimensional. Posteriormente se muestra el contenido de cada espacio de dicha variable, en donde podemos observar que algunos espacios están inicializados con ceros, pero otros guardan "basura". Por último, se libera el espacio reservado por la función malloc mediante un llamado a la función free.



  2) Ejemplo calloc: El programa inicia preguntándole al usuario cuantos elementos tiene el conjunto. Dicho dato, junto con el tamaño que ocupa un entero se le pasa como parámetro a la función calloc que retorna un puntero void que indica la primera posición desde la cual se reservó el espacio solicitado y, este se convierte en uno de tipo entero y se guarda en una variable, la cual se puede visualizar como si fuera un arreglo

unidimensional.
Posteriormente se muestra el contenido de cada espacio de dicha variable, en donde podemos observar que todos los espacios están inicializados con ceros. Por último, se libera el espacio reservado por la función calloc mediante un llamado a la función free.



3) Ejemplo realloc: El programa inicia preguntándole al usuario cuantos elementos tiene el conjunto. Dicho dato, se multiplica por el tamaño que ocupa un entero y se le pasa como parámetro a la función malloc que retorna un puntero void que indica la primera posición desde la cual se reservó el espacio solicitado y, este se convierte en uno de tipo entero y se guarda en una variable, la cual se puede visualizar como si fuera un arreglo unidimensional. Una vez creado, se le pedirá al usuario que inserte valores enteros en cada espacio hasta alcanzar el límite de elementos que se pueden almacenar. Posteriormente se muestra el contenido de cada espacio de dicha variable, en donde podemos observar que todos los espacios contienen los datos que introducimos anteriormente. Finalizado lo anterior, se duplica el número de elementos que tenía el conjunto inicialmente y se multiplica por el tamaño que ocupa un entero para que, junto con el espacio asignado previamente por la función malloc, se le pasen como parámetros a la función realloc, la cual redimensiona un espacio previamente ocupado. Se le vuelve a pedir al usuario que ingrese valores enteros en cada espacio.

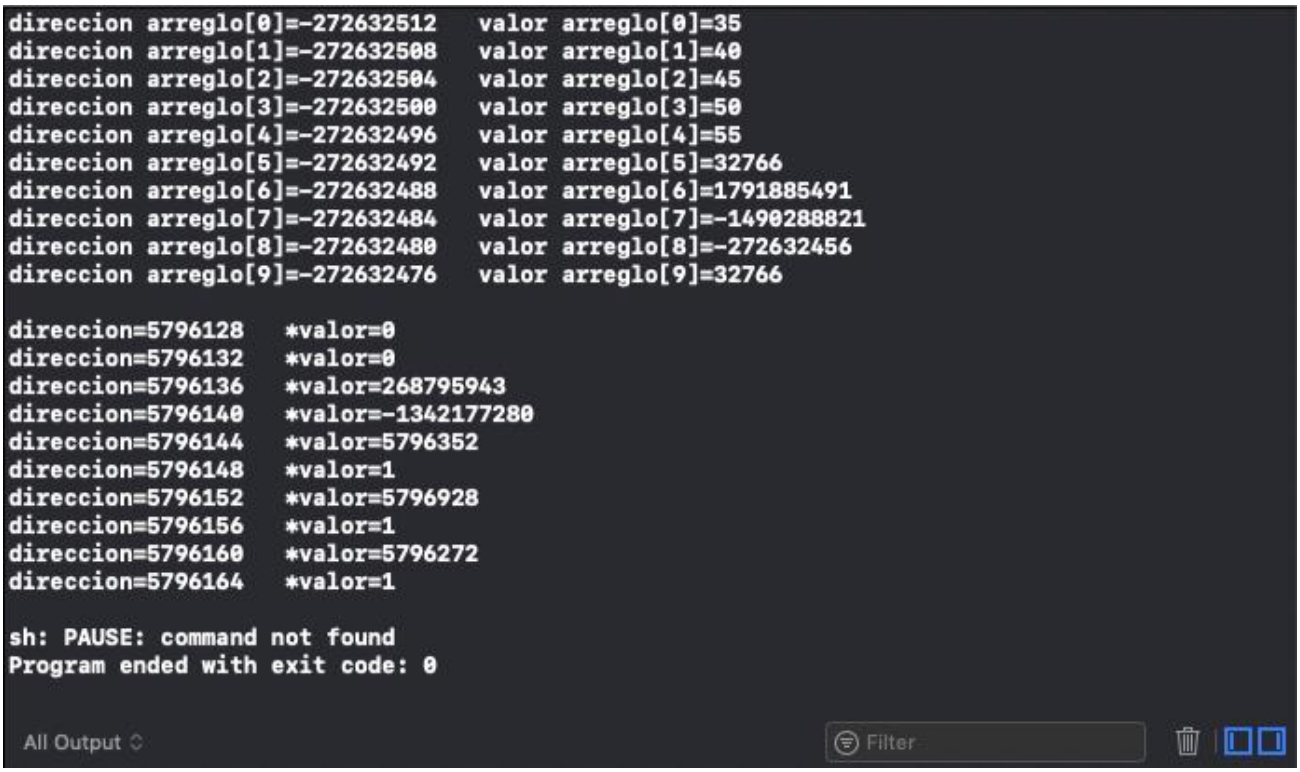
hasta alcanzar el límite de elementos almacenables.
La diferencia es que empieza a guardar valores después de los espacios ocupados por la primera variable además, esta nueva variable tiene el doble de elementos que la primera variable declarada. Nuevamente, se muestran los valores ingresados previamente.
Por último, se libera el espacio reservado por la función calloc mediante un llamado a la función free sin embargo, también se debería de liberar el espacio ocupado por la variable arreglo2, y no se realiza, por lo que sería el único error dentro del programa.

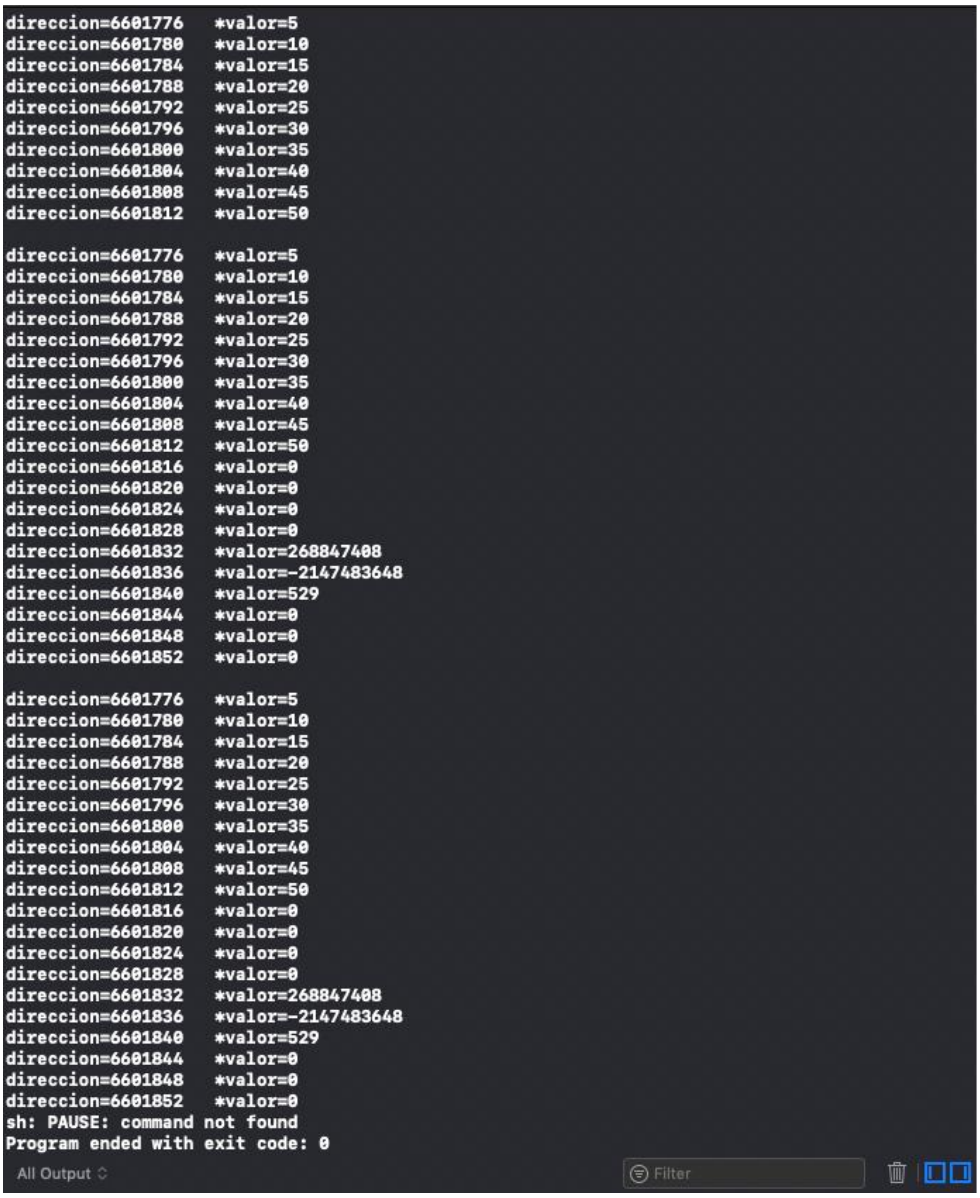
```
¿Cuántos elementos tiene el conjunto?
5
Inserte el elemento 1 del conjunto.
6
Inserte el elemento 2 del conjunto.
8
Inserte el elemento 3 del conjunto.
2
Inserte el elemento 4 del conjunto.
8
Inserte el elemento 5 del conjunto.
34
Vector insertado:
    [   6   8   2   8   34  ]
Aumentando el tamaño del conjunto al doble.
Inserte el elemento 6 del conjunto.
39
Inserte el elemento 7 del conjunto.
65
Inserte el elemento 8 del conjunto.
432
Inserte el elemento 9 del conjunto.
98
Inserte el elemento 10 del conjunto.
54
Vector insertado:
    [   6   8   2   8   34  39  65  432 98
54  ]
Program ended with exit code: 0
```

Ejercicios propuestos

1)Ejercicio 1: En este programa se puede observar la diferencia que existe entre el espacio en donde se almacena un arreglo y el espacio en donde se almacena la memoria dinámica. El arreglo se almacena en localidades de memoria mucho más altas que las localidades de la memoria dinámica.

Cuando se utiliza calloc, se puede apreciar que todos los valores que no se habían asignado previamente, se inicializan con ceros.
Al momento de redimensionar el tamaño de la memoria dinámica utilizando la función calloc, se conservan los valores que ya se habían asignado previamente, en donde se guardaron los múltiplos de cinco. Además de esto resulta lo mismo asignar el nuevo tamaño al mismo apuntador ptr que asignar el nuevo tamaño a otro apuntador.

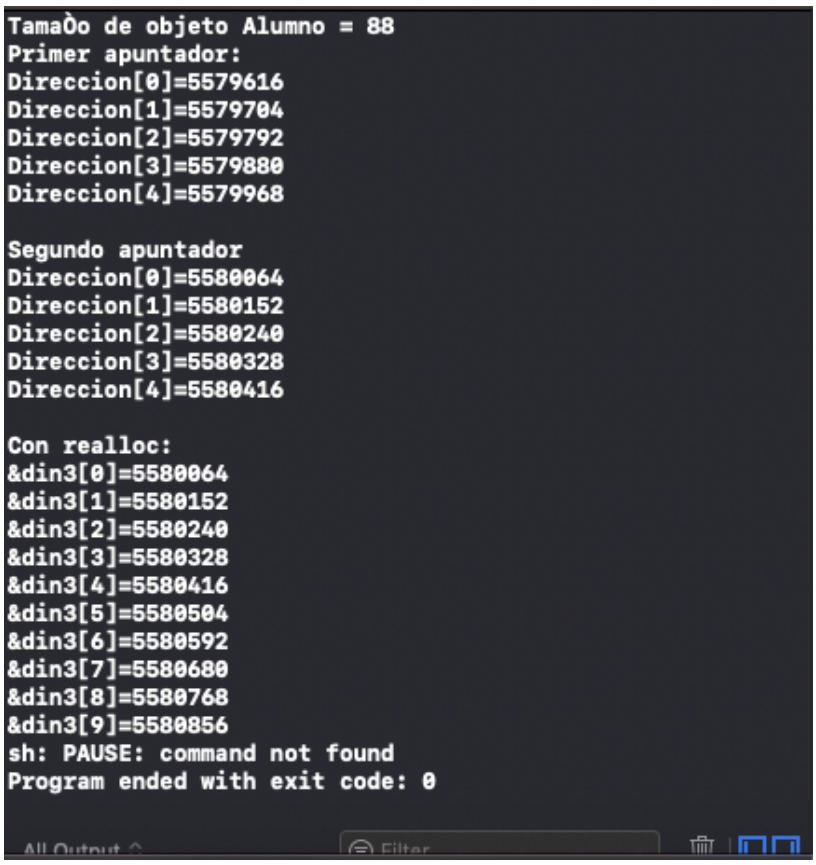




2)Ejercicio 2: Al iniciar el programa podemos observar que se muestra en pantalla el tamaño que ocupa la estructura previamente declarada como Alumno, la cual a su vez contiene una estructura de tipo Direccion. Después se observa que, haciendo uso de un primer puntero de tipo Alumno y utilizándolo para hacer uso de memoria dinámica mediante la función malloc, se muestra la dirección de memoria de cada espacio reservado previamente, en donde se puede almacenar cierto valor de tipo estructura Alumno. Se aprecia que entre cada localidad existe exactamente 88 bytes de diferencia, lo que confirma que el tamaño que ocupa la estructura Alumno es de 88 bytes.

Se realiza lo mismo para un segundo puntero de tipo Alumno, el cual también se utiliza para hacer uso de memoria dinámica mediante la función calloc. Finalmente, se modifican las dimensiones del puntero dos y el resultado se guarda en un puntero tres e imprime dicho resultado en pantalla, en donde se observa que las direcciones de memoria del segundo puntero coinciden con las del tercer puntero hasta que se alncanza el último espacio que se había reservado antes de ser redimensionado.

Una forma para poder liberar memoria previamente reservada, sería haciendo uso de realloc y redimensionando la memoria reservada por malloc o calloc para que sea 0, y de esta forma se reduciría el espacio asignado hasta volverse cero, lo cual podría verse como si dicho espacio fuera liberado.



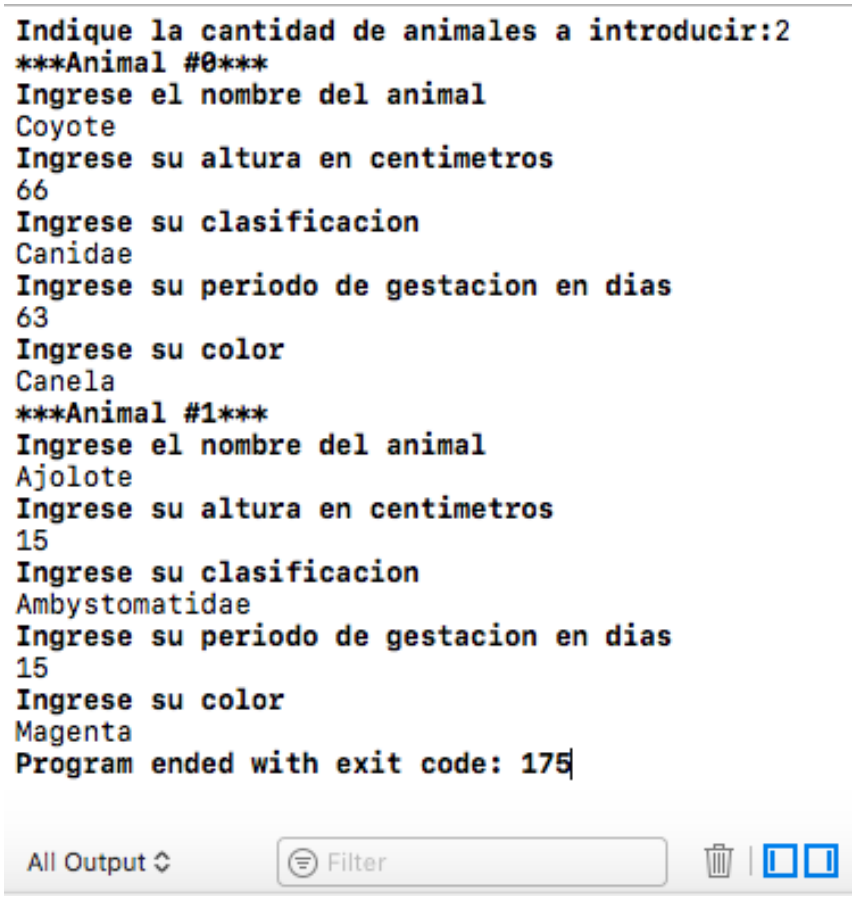
3)Ejercicio 3: Se crea una estructura de tipo Animal, la cual cuenta con tres arreglos de tipo entero, una variable de tipo entero y otra variable de tipo corto.

Durante la ejecución del programa se le pregunta al usuario la cantidad de animales que quiere crear, dicho valor se guarda en una variable de tipo corta y se pasa como parámetro a la función crearAnimal.

Dentro de la función crearAnimal se crea un apuntador de tipo Animal, el cual se utiliza para crear un arreglo dinámico de tipo Animal mediante la función calloc, el cual toma como parámetros la cantidad de animales que el usuario indicó que creará y el tamaño que ocupa la estructura Animal.

Por último, se le pide al usuario que ingrese los datos correspondientes al nombre, altura, clasificacion, periodo de gestación y color del animal, en donde cada uno se guarda en su correspondiente miembro de la estructura de tipo Animal.

La única dificultad que se encontró al momento de la realización del programa es el como hacer uso correcto de la memoria dinámica, ya que se deben de cuidar los parámetros que se le pasen a la función, ya sea malloc o realloc. De igual forma, como se vió en clase, es bastante importante que después de utilizar la memoria dinámica, esta se libere haciendo uso de la función free. Por último, se observó la manera en como un puntero de tipo estructura Animal sirve para poder indicar en que espacio se reservó la memoria solicitada , y haciendo uso de ese mismo puntero poder utilizarlo como un arreglo dinámico de tipo estructura Animal.



- Conclusiones de la práctica

  A manera de conclusión, en esta práctica se observaron las diferencias entre el uso de malloc y calloc, en donde calloc inicializa los valores destinados al espacio solicitado con ceros, mientras que malloc no lo hace, lo que algunas veces provoca que se muestran datos aleatorios cuando se muestra lo que tiene cada espacio reservado con malloc sin inicialización previa.

  Lo objetivos de la práctica se cumplieron, ya que se lograron identificar las diferentes formas de hacer uso de la memoria dinámica, como hacer uso de realloc para redimensionar un espacio de memoria asignado previamente, la importancia de liberar el espacio asignado y el uso de punteros para la creación de arreglos dinámicos.

  La práctica ayudó a ejemplificar lo visto en clase, aunque se pudo haber incluido un ejercicio adicional para lograr un mayor refuerzo del tema, sin embargo la práctica permite que se logre un mejor entendimiento de las diferencias entre malloc y calloc, así como la posición de memoria en la que se almacena la memoria dinámica.