



## 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):* 02

*Integrante(s):* GÓMEZ LUNA ALEJANDRO

*No. de Equipo de  
cómputo empleado* 34

*Semestre:* 2019-2

*Fecha de entrega:* 18/FEBRERO/2019

*Obervaciones:*

**CALIFICACIÓN:** \_\_\_\_\_

- Objetivo de la práctica

- Desarrollo

### Ejercicios Propuestos

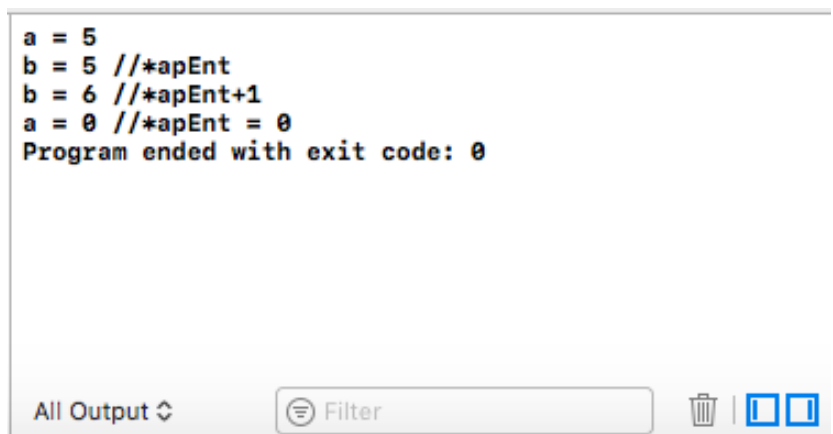
- a) Ejercicio 1: El programa nos muestra la declaración de una variable de tipo carácter, la cual almacena la letra 'x', y la declaración de un puntero de tipo carácter, el cual almacena la dirección de la variable previamente declarada. Posteriormente se imprime el carácter, su código ASCII y la dirección de memoria en la que está almacenada, todo haciendo uso únicamente del puntero asociado a la variable.

```
Carácter: x
Código ASCII: 120
Dirección de memoria: 1606416063
Program ended with exit code: 0
```



- b) Ejercicio 2: El programa nos muestra la declaración de dos variables de tipo entero corto cuyo nombre son a y b respectivamente, y un puntero de tipo de entero corto denominado apEnt, que almacena la dirección de la variable a. Haciendo uso del puntero, se guarda el valor de la variable a en la variable b, y se realizan algunas modificaciones para posteriormente imprimirse en pantalla, siempre haciendo uso del apuntador.

```
a = 5
b = 5 /*apEnt
b = 6 /*apEnt+1
a = 0 /*apEnt = 0
Program ended with exit code: 0
```



- c) Ejercicio 3: El programa nos muestra la declaración de un arreglo unidimensional con su respectivo puntero, ambos de tipo entero corto, en donde se observa como efectivamente, conforme a lo que vimos en clase, con solo asociar el apuntador con la primera posición del arreglo, ya se puede acceder a cualquier posición del arreglo utilizando únicamente el puntero. De igual forma, se comprueba que al solamente escribir el identificador del arreglo se hace alusión a la dirección de memoria de dicho arreglo, por lo que un arreglo se puede asociar con un puntero sin necesidad de utilizar el operador de dirección &.

```
Dirección del arreglo en la primera posición:  
5fbff6be  
Dirección del arreglo: 5fbff6be  
Dirección del apuntador: 5fbff6be  
Program ended with exit code: 0
```

All Output ↺

Filter



- d) Ejercicio 4: El programa nos muestra la declaración de un arreglo unidimensional y un puntero asociado con dicho arreglo, ambos de tipo entero. Existe la declaración de una tercera variable de tipo entero, sin embargo, esta variable no se ocupa en todo el programa por lo que la podemos descartar del programa.

Se observa como el valor de la primera posición del arreglo cambia cuando, al usar una función, se pasa un parámetro mediante valor y al usar otra función, se pasa un parámetro mediante referencia, ya que cuando se hace por referencia el valor de la variable cambiará sin importar el alcance que tenga esta.

```
Pasar valor: 55
55
128
Pasar referencia: 55
55
128
Valor final: 128
Program ended with exit code: 0
```

All Output

Filter



e)

Ejercicio 5: El programa nos muestra la declaración de un arreglo con su respectivo puntero, ambos de tipo entero corto. De esta manera, el puntero asociado con el arreglo puede recorrer todas las posiciones del arreglo, simplemente sumándole al puntero el índice correspondiente a la posición del arreglo a la que se quiera acceder.

```
5fbff6be
5fbff6c0
5fbff6c2
5fbff6c4
5fbff6c6
Program ended with exit code: 0
```

All Output

Filter



f) Ejercicio 6: El programa nos muestra la declaración de un arreglo bidimensional con su respectivo puntero, los dos de tipo entero. Al igual que el programa anterior, el puntero asociado con el arreglo puede recorrer todas las posiciones del arreglo, simplemente sumándole al puntero el índice correspondiente a la posición del arreglo a la que se quiera acceder, sin importar la dimensión que tenga el arreglo, ya que el puntero recorre posición por posición, sin considerar si el arreglo tiene filas, planos, etc.

```
5fbff6a0    5fbff6a4    5fbff6a8  
5fbff6ac    5fbff6b0    5fbff6b4  
5fbff6b8    5fbff6bc    5fbff6c0    Program ended  
with exit code: 0
```

All Output ↕

Filter



g) Ejercicio 7: Este programa realiza un cifrado César, el cual fue ideado por el comandante Julio César, para enviar órdenes a sus generales en los campos de batallas. El cifrado consistía en mover el abecedario tres posiciones, es decir que en este cifrado, el abecedario comenzaría en la letra D y terminaría en la letra C.

Se comienza desplegando el abecedario que se está utilizando, el cual no contiene a la letra ñ. El abecedario se despliega haciendo uso del apuntador de indirección y el nombre del arreglo únicamente, el cual indica la dirección donde se encuentra almacenado. Posteriormente se despliega un pequeño menú, con tres opciones a elegir.

- Opción 1: En esta opción tú puedes cifrar cualquier palabra. Para esta opción se llama a la función cifrar, a la cual se le pasa el parámetro por referencia al arreglo en donde se guarda la palabra introducida por el usuario.

Dentro de dicha función, se utiliza un ciclo for dentro de otro. El ciclo for 'exterior' es el encargado de ir letra por letra de la palabra introducida por el usuario, y se detiene hasta que se encuentra un carácter de tipo NULL. El ciclo for 'interior' es el encargado de verificar con que letra del abecedario coincide la letra de la palabra que se introdujo, y una vez que se encuentra, se imprime en pantalla la letra correspondiente al cifrado César. Una vez que se encuentra, se sale inmediatamente del ciclo for interior.

Opción 2: En esta opción tú puedes descifrar cualquier palabra. Para esta opción se llama a la función descifrar, a la cual se le pasa el parámetro por referencia al arreglo en donde se guarda la palabra introducida por el usuario. Dentro de dicha función, se utiliza un ciclo for dentro de otro. El ciclo for 'exterior' es el encargado de ir letra por letra de la palabra introducida por el usuario, y se detiene hasta que se encuentra un carácter de tipo NULL. El ciclo for 'interior' es el encargado de verificar con que letra del cifrado César coincide la letra de la palabra que se introdujo, y una vez que se encuentra, se imprime en pantalla la letra correspondiente al abecedario normal. Una vez que se encuentra, se sale inmediatamente del ciclo for interior.

- Opción 3: En esta opción, se puede salir del programa.

```
*** CIFRADO CÉSAR ***
ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC
Elegir una opción:
1) Cifrar
2) Descifrar.
3) Salir.
1
Ingresar la palabra a cifrar (en mayúsculas): ARTISTA
El texto ARTISTA cifrado es: DUWLVD

*** CIFRADO CÉSAR ***
ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC
Elegir una opción:
1) Cifrar
2) Descifrar.
3) Salir.
2
Ingresar la palabra a descifrar (en mayúsculas):
DUWLVD
El texto DUWLVD descifrado es: ARTISTA

*** CIFRADO CÉSAR ***
ABCDEFGHIJKLMNOPQRSTUVWXYZ
DEFGHIJKLMNOPQRSTUVWXYZABC
Elegir una opción:
1) Cifrar
2) Descifrar.
3) Salir.
3
Program ended with exit code: 0
```

## Ejercicios en clase

a) **Ejercicio 1:** Este programa modifica los valores de tres variables de tipo entero, de manera indirecta, es decir, modifica el valor de estas variables mediante tres apuntadores, cada uno que almacena la dirección de una variable. En el código se encontraron algunos errores, como la falta del operador de dirección para guardar la dirección de una variable dentro del apuntador, y también la falta o sobra del operador de indirección, el cual hace referencia al valor de la variable cuya dirección se encuentra almacenada dentro de un puntero.

Por último, se imprimen los valores de las variables ya modificados.



```
Los resultados son: 30, 300 y 50
Program ended with exit code: 34
```

The screenshot shows a dark-themed terminal window. The output text is white. At the bottom, there is a toolbar with the text 'All Output', a 'Filter' input field, a trash icon, and two window control icons.

b) **Ejercicio 2:** El programa guarda valores dentro de un arreglo de tipo entero de tres dimensiones, cuyo identificador es `arr1`, posteriormente se asocia el puntero `point` con dicho arreglo. Se le asigna el valor 1 a la variable `var` de tipo entero y dentro de tres ciclos `for` anidados, se empiezan a guardar valores, utilizando la variable `var`, dentro del arreglo, cuyo valor va incrementando de tres en tres, empezando desde el plano 0, renglón 0, columna 0.

Por último, se guardan valores del arreglo, a través de su respectivo puntero, en las tres variables `a`, `b` y `c`, respectivamente.

Dentro de estas variables, `a`, `b` y `c`, se guardan las posiciones del arreglo cuyos índices son:

- Para la variable `a`: `arr1[0][1][3]`
- Para la variable `b`: `arr1[1][1][1]`
- Para la variable `c`: `arr1[2][1][2]`

Para el último inciso, he decidido imprimir todos los valores dentro del arreglo, para comprobar y demostrar que el arreglo se llenó según se requería en las instrucciones, se imprime el índice del arreglo y el valor que se almacena en dicha posición del arreglo. Para poder llenar el arreglo conforme se pedía, una dificultad que se presenta es el de utilizar un solo ciclo con un solo acontador para llenar todo el arreglo, ya que se necesita utilizar a todo el arreglo sin distinción de columnas, renglones y planos, mientras que en las instrucciones del inciso nos piden que llenemos cada plano de cierta forma, por lo que opté por multiplicar el número de columnas con el número de renglones, para poder saber cuantas columnas y renglones existen por plano. De esta manera, comparé el valor límite de columnas y renglones por cada plano, para poder realizar el llenado correctamente.






```
El valor del arreglo en la posicion[0][0][0] es: 5
El valor del arreglo en la posicion[0][0][1] es: 10
El valor del arreglo en la posicion[0][0][2] es: 15
El valor del arreglo en la posicion[0][0][3] es: 20
El valor del arreglo en la posicion[0][1][0] es: 25
El valor del arreglo en la posicion[0][1][1] es: 30
El valor del arreglo en la posicion[0][1][2] es: 35
El valor del arreglo en la posicion[0][1][3] es: 40
El valor del arreglo en la posicion[0][2][0] es: 45
El valor del arreglo en la posicion[0][2][1] es: 50
El valor del arreglo en la posicion[0][2][2] es: 55
El valor del arreglo en la posicion[0][2][3] es: 60
El valor del arreglo en la posicion[1][0][0] es: 6
El valor del arreglo en la posicion[1][0][1] es: 12
El valor del arreglo en la posicion[1][0][2] es: 18
El valor del arreglo en la posicion[1][0][3] es: 24
El valor del arreglo en la posicion[1][1][0] es: 30
El valor del arreglo en la posicion[1][1][1] es: 36
El valor del arreglo en la posicion[1][1][2] es: 42
El valor del arreglo en la posicion[1][1][3] es: 48
El valor del arreglo en la posicion[1][2][0] es: 54
El valor del arreglo en la posicion[1][2][1] es: 60
El valor del arreglo en la posicion[1][2][2] es: 66
El valor del arreglo en la posicion[1][2][3] es: 72
El valor del arreglo en la posicion[2][0][0] es: 7
El valor del arreglo en la posicion[2][0][1] es: 14
El valor del arreglo en la posicion[2][0][2] es: 21
El valor del arreglo en la posicion[2][0][3] es: 28
El valor del arreglo en la posicion[2][1][0] es: 35
El valor del arreglo en la posicion[2][1][1] es: 42
El valor del arreglo en la posicion[2][1][2] es: 49
El valor del arreglo en la posicion[2][1][3] es: 56
El valor del arreglo en la posicion[2][2][0] es: 63
El valor del arreglo en la posicion[2][2][1] es: 70
El valor del arreglo en la posicion[2][2][2] es: 77
El valor del arreglo en la posicion[2][2][3] es: 84
Program ended with exit code: 3
```



c) Ejercicio 3: En este programa se le pide al usuario que ingrese dos valores, en donde el primer número es elevado al segundo número y ese resultado se guarda en la primera variable. Posteriormente, se hace la división del resultado anterior entre el valor del primer número ingresado y ese resultado se guarda en la segunda variable. La dificultad con el programa es que se necesita pasar valor por referencia a una función, por lo que cuando se almacena el valor del primer resultado de la potencia, en la misma primera variable, el valor se modifica y se sobrescribe, sin embargo se necesita volver a utilizar el valor de la primera variable, por lo que se necesita guardar ese valor otra variable dentro de la función, para poder utilizarlo cuando se requiere utilizar la operación de división.

Como la función no retorna ningún valor, dentro de la función se imprimen los resultados de las operaciones realizadas usando las dos variables.

```
Ingrese dos valores, separados por comas:
17,5
El primer numero 17 elevado al segundo numero 5 es:
  1419857
La division del resultado anterior 1419857 entre el primer
numero 17 es: 83521
Program ended with exit code: 175|
```

All Output   Filter   

d) Ejercicio 4:

- Evidencia de la implementación
- Conclusiones de la práctica