CS1112: Programación II

Unidad 3: Punteros

Sesión de Laboratorio - 4B

Profesores:
María Hilda Bermejo mbermejo@utec.edu.pe
Jonathan Silva jsilva@utec.edu.pe
Jorge Villavicencio jvillavicencio@utec.edu.pe
Henry Gallegos hgallegos@utec.edu.pe
Ian Paul Brossard ibrossard@utec.edu.pe
Jose Chavez jchaveza@utec.edu.pe
Wilder Nina wnina@utec.edu.pe
José Fiestas jfiestas@utec.edu.pe

Material elaborado por:
Maria Hilda Bermejo, Ruben Rivas

Índice:

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

3.2 Unidad 3: Arreglos con punteros

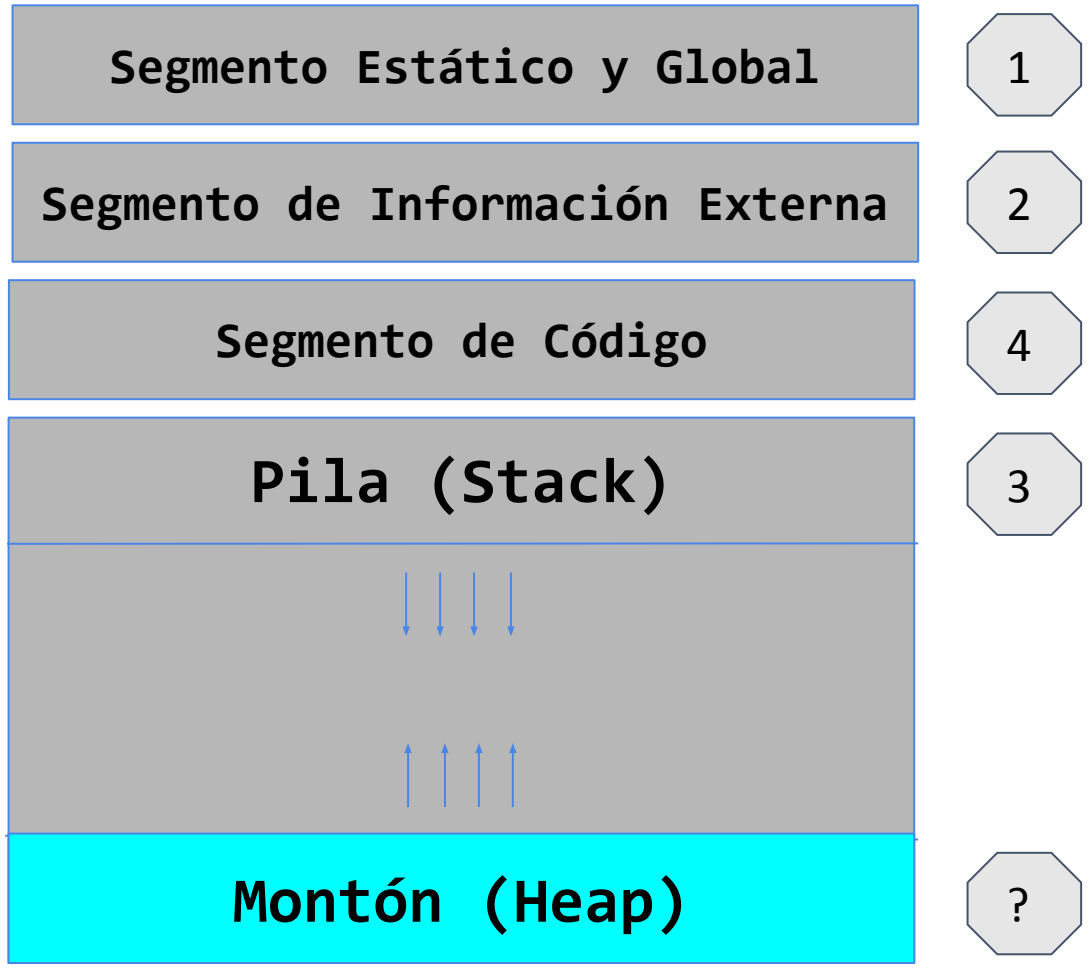
Logro de la sesión:

Al finalizar la sesión, los alumnos podrán:

- Desarrollar programas utilizando punteros para el acceso a valores por referencia.

Manejando memoria

Programa de C++ en la memoria primaria




```
#include <iostream>
using namespace std;

int varGlobal = 20;                          1

int main(int argc, char * argv[])            2
{
    int varLocal = 10;                       3
     int* ptrVarLocal = &varLocal;

    cout << varLocal << "\n";
    return 0;
}
```

4

Al Heap solo se puede acceder a través del uso de punteros.

Arreglos

Un array es una estructura de datos, que permita almacenar **elementos del mismo tipo**, los que se acceden por su posición.

Los arrays pueden ser **estáticos** y **dinámicos**.

Los arrays **estáticos** tiene un tamaño definido y su tamaño no puede variar en todo el programa. Para definir un array estático, se tiene que saber su tamaño previamente.

Un array **dinámico** es más flexible y su tamaño se puede decidir cuando se ejecuta el programa.

Un array estático se define así:

int A[10]={9,45,3,7,10,25,14,15,100,120};

Una vista lógica se muestra a continuación :

	0	1	2	3	4	5	6	7	8	9
A	9	45	3	7	10	25	14	15	100	120

Para hacer referencia al elemento se usa el índice:

```
cout << A[3];   //---- imprime el 7
cout <<A[1];   //-----imprime el 45
```

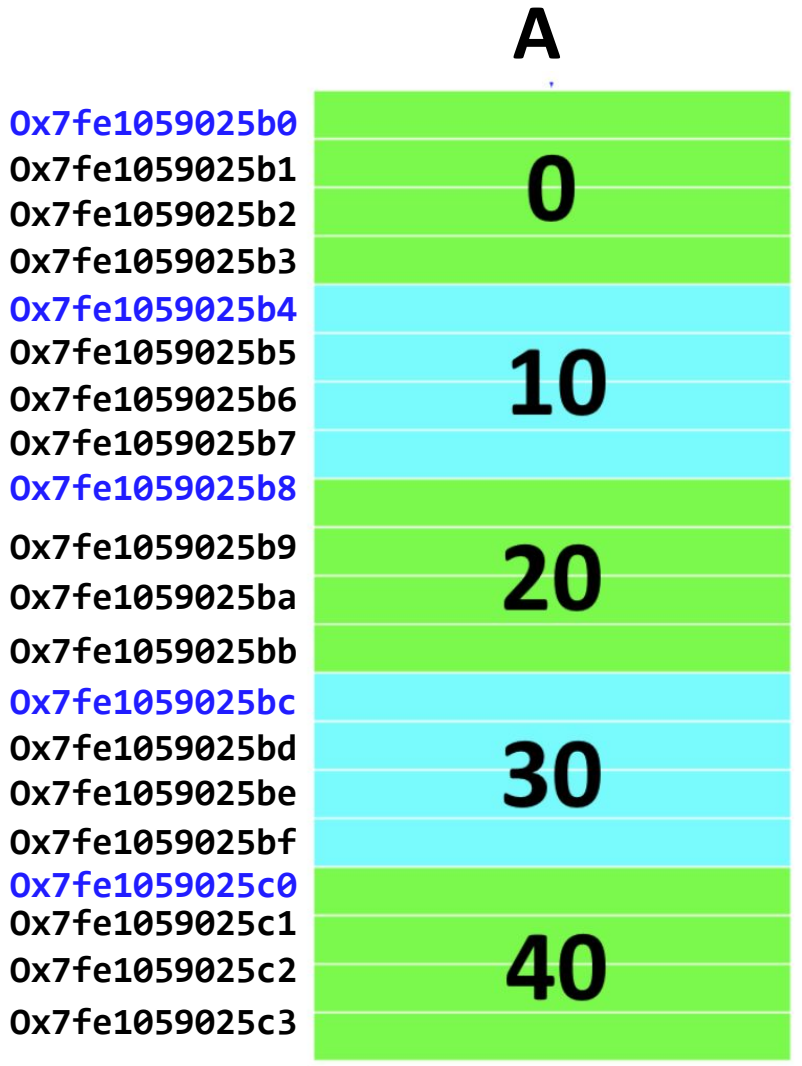
Definición:

```
int arr[10];                    // es un array de 10 elementos.

int a2[]={0,1,2};               //-- es un array de 3 elementos

int a3[5] = {0,1,2};            //-- equivale a    int a[]={0,1,2,0,0};


string a4[3]={“hi”,  ”bye”};    //--- equivale a:   string a4[]={“hi”,  “bye”, “”};

int a5[2]={0,1,2};              //--- es un error:  muchos inicializadores.

const unsigned sz=3;
int a1[sz]={0,1,2};  //-- define un array de 3 elementos con valores 0,1,2

int w;
float F[w];    //--- error:   w es una variable.
```

Cuando se define un array, los elementos están de manera contigua en la memoria del computador.

int A[5]={0,10,20,30,40};

A representa la dirección de memoria donde se encuentra el primer elemento del array.



Cuando se quiere imprimir los elementos de un array, no se puede realizar con una sola instrucción, se debe recorrer uno a uno los elementos del array.

```
#include <iostream>
using namespace std;

int main()
{ int  A[7]={ 71,72,73,74,75,76,77};

 //--- se recorre el array y se imprimen los datos
 for(int i=0; i<7; i++)
   cout << "A[" << i << "]=" << A[i] << "\n";

 //-- se imprimen las direcciones donde se encuentra cada elemento
 cout << "\nInicio del array " << A << "\n";

 cout << "\nImprimimos las direcciones de cada casillero del
          Arreglo A \n";
 for(int i=0; i<7; i++)
   cout << "&A[" << i << "]= " << &A[i] << "  Guarda el " << A[i] <<"\n";

 return 0;
}
```

Se imprime:

```
A[0]=71
A[1]=72
A[2]=73
A[3]=74
A[4]=75
A[5]=76
A[6]=77

Inicio del array 0x7fff52a26a00

Imprimimos las direcciones de cada
casillero del Arreglo A
&A[0]= 0x7fff52a26a00  Guarda el 71
&A[1]= 0x7fff52a26a04  Guarda el 72
&A[2]= 0x7fff52a26a08  Guarda el 73
&A[3]= 0x7fff52a26a0c  Guarda el 74
&A[4]= 0x7fff52a26a10  Guarda el 75
&A[5]= 0x7fff52a26a14  Guarda el 76
&A[6]= 0x7fff52a26a18  Guarda el 77
```

Ejemplo 1:

Escriba un programa que permita leer dos datos de tipo entero, las almacene en dos variables estáticas y luego el programa realizará lo siguiente:

1. Imprimir los números leídos
2. A través de una función llamada Intercambiar, intercambie el valor de las variables.
3. Imprimir nuevamente el valor de las variables.
4. Multiplicar los valores de las variables e imprimirlos

Ejemplo 2:

Realice un programa que lea la cantidad de elementos de un array.
Luego genere el array, llénelo con números aleatorios entre 0 y 999. e imprima el array
Genere a partir de ese array dos nuevos arrays,
el primero con los múltiplos de 5 y el segundo con los múltiplos de 7

Ejemplo 3:

Escriba un programa que permita leer como dato un número entero mayor a 15, el cual representará el número de alumnos de un salón de clase.
El programa luego deberá dimensionar un array para leer por teclado cada una de las N notas para luego hallar:

1. Imprimir solo las notas que están por encima del promedio
2. Imprimir la mayor nota
3. Imprimir el promedio eliminando la menor nota. Si el valor de la nota menor se repite se deberá eliminar todas las ocurrencias y realizar el cálculo del promedio con las notas restantes.

Diseñe el programa de tal manera que se utilicen funciones.

Resumen

En esta sesión aprendiste a:

1. Acceder al **stack** y **heap** de la memoria por medio de los punteros.
2. Usar **arreglos** que son tipos de datos compuestos que permiten almacenar una colección de datos del mismo tipo en forma secuencial y consecutiva.
3. Comprender que las variables almacenadas en la memoria automática o stack se reservan y liberan de forma automática.
4. Reservar y liberar de forma manual en la memoria dinámica o heap por medio de los operadores new y delete.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Bibliografía:

Deitel. P.J. and Deitel. H. M. (2016) C++ How to Program, Prentice Hall.

Stroustrup, Bjarne (2013). The C++ Programming Language, 4th Addison-Wesley.

Eckel, Bruce, 2000. Thinking in C++, Vol 1: Introduction to Standard C++, 2nd Edition, Prentice Hall

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

¡Nos vemos en la siguiente clase!

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA