RESPUESTAS Actividades

Unidad 1 – Punteros: Ejercicios

Se sugiere el uso del editor / compilador C++ **“Zinjal” o “ C-Free”**

1 • Explique el significado de las siguientes expresiones de código C++.

a) char *m; Puntero Tipo Char	f) int *a[30]; Arreglo de punteros de 30 posiciones.
b) double x,y; double *px, *py; x,y Variables numéricas px, py Punteros a datos numéricos double.	g) float f(int *a, int * b); Función con parametros tipo puntero.

c) int a=25;
int *k=&a;
a variable tipo entero, k puntero que apunta a la dirección de a.
	h) int (*pfunc)(void);
	
	pfunc: puntero a función sin parámetros que retorna un entero.

d) float (*q)[20]; Puntero a 1er elemento de un Arreglo de 20, de tipo flotante.	i) char [3]={“rojo”,”gris”,”azul”} Error, una alternativa seria: int main(void){ char p[3][5]={"rojo\0","gris\0","azul\0"}; puts(p[0]);puts(p[1]);puts(p[2]);}
e) float x,y; float *px , *py=&y; x, y: variables numéricas. px, py: punteros, py apunta a la dirección de y.	j) float(*pfunc)(int *a,char b); pfunc: puntero a funcion que retorna un float, con parámetros a puntero tipo entero (pasaje por referencia) y b variable tipo char (pasaje por valor).

2 • Usando la sintaxis de C++ escriba el código que crea necesario para:

- Declarar un puntero a datos de tipo flotante y otro puntero a datos de doble precisión.

```
Float *a;
Double *b;
```

- Declarar un puntero a una lista de 10 arreglos contiguos de 15 elementos enteros cada uno. Reserve la memoria necesaria.

```
//U1E2a_CV
#include <iostream>
#include <cstdlib>
using namespace std;

int main(int argc, char *argv[]) {
  int (*p)[15] = new int [10][15]; //Creo un Arreglo Dinámico
  int x,y;
  cout<<endl<<"Cargo el arreglo dinamico con valores entre 10 y 20."<<endl;
```

```
  for (x=0; x<10; x++){
        for(y=0; y<15; y++){
              *(*(p+x)+y) = rand()%10+100;//asigno valores
              cout<<*(*(p+x)+y)<<"  ";//Muestro las posiciones
        };
        cout<<endl;//Separo las filas
  };
  delete p; //Libero la memoria ocupada por el arreglo p
  return 0;
}
```

- Declarar un arreglo de punteros para representar una matriz de 10x30 elementos flotantes.

```
//U1E2c_CV
#include <iostream>
#include <iomanip.h>
using namespace std;

int main(void){
  float *f[10]; //cantidad de punteros (filas)
  for (int fila=0; fila<10; fila++) {
        // reservamos espacio para 30 datos flotantes por cada fila
        f[fila]=(float *) malloc(30*sizeof(float)); //Reserva
        for (int columna=0; columna<30; columna++){
              //Carga con valores aleatorios
              *(*(f+fila)+columna)= (float)(rand()/100*0.22);
              cout<<setw(6)<< *(*(f+fila)+columna)<<"  ";
        };};
return 0;
}//fin
```

- Declarar un puntero a una función que acepte 2 parámetros de doble precisión y no devuelva resultado alguno.

```
void (*pfuncion) (double x, double y) ;
```

```
//U1E2c Codigo Ejemplo
#include <iostream>
using namespace std;
void suma(double x, double y)
{cout<<"x= "<<x<<"  y= "<<y<<" x+y= "<<x+y<<endl;};

void resta(double x, double y)
{cout<<"x= "<<x<<"  y= "<<y<<" x-y= "<<x-y<<endl;};

void mifuncion( double a, double b, void (*pfuncion)(double x1, double y1)){
  pfuncion(a, b);};

int main(void){
  double g=6, f=2;
  mifuncion(g, f, suma);
  mifuncion(g, f, resta);
  return 0;
```

```
}
```

- Declarar una función que tenga otra función como argumento y devuelva un puntero a un entero. La función argumento debe tener 2 parámetros enteros y devuelve un carácter.

Int *pfunc1 (char func2 (int a, int b));

3 • Observe la porción de código C++ del recuadro de la derecha y determine la salida que se obtiene de los flujos de salida **cout** propuestos.
Considere que la variable **a** se ha almacenado en memoria a partir de la dirección **0000FF09**.

```
.....
int a=90;
int *p=&a;
int b=(*p)++;
int *q=p+2;
cout<<p<<" "<<*p;
cout<<q<<" "<<*q;
cout<<a<<" "<<b;
p++; b=*(q--)-1;
a=(*p++)+1;
cout<<a<<"  "<<b;
.....
```

```
int main(void)
{int a=90;
int *p=&a;  // *p=90 y p tiene la direccion de a
cout<<"&a= "<<&a<<"  a= "<<a<<"  p= "<<p<<"  *p=
 "<<*p<<endl<<endl;
int b= (*p)++; // b=90 despues  *p=91 a=91
cout<<"int b= (*p)++  ##"<<endl;
cout<<"b= "<<&b<<"  b= "<<b<<"  p= "<<p<<"  *p= "<<*p<<endl<<endl;
int *q=p+2;    // q = direccion de p+2 bytes
cout<<"int *q=p+2; ##"<<endl;
cout<<"p+2= "<<p+2<<endl;
cout<<"p= "<<p<<" "<<"*p= "<<*p<<endl;
cout<<"q= "<<q<<" "<<"*q= "<<*q<<endl;
cout<<"a= "<<a<<" "<<"b= "<<b<<endl<<endl;
p++; //p = direccion de p+4 bytes en la próxima línea.
cout<<"p++ ##"<<endl;
cout<<"p= "<<p<<"  *p= "<<*p<<endl<<endl;
b=*(q--)-1; //b = contenido de q - 1 y después q = q-4 bytes
cout<<"b=*(q--)-1;  ##"<<endl;
cout<<"b= "<<&b<<"  b= "<<b<<endl;
cout<<"q= "<<q<<"  *q= "<<*q<<endl<<endl;
a=(*p++)+1; //a = direccion de p + 1 byte después p = p+4 bytes
cout<<"a=(*p++)+1;  ##"<<endl;
cout<<"a= "<<a<<" "<<"b= "<<b<<endl;
cout<<"p= "<<p<<"  *p= "<<*p<<endl<<endl;
}
```

4 • Analice el código C++ del recuadro de abajo para responder lo siguiente:
Qué tipo de parámetros actuales se emplean para llamara a **func** ?.
Se emplea un puntero que referencia a un arreglo de enteros.

Qué tipos de parámetros formales se definen en **func**?.
Puntero a entero.

```
....
void func(int *p);
{int i, sum=0;
for (i=0;i<6;i++)
    sum+=*(p+i);
cout<<"sum="<<sum<<endl;}

int main(void)
{int x[6]={12,34,56,78,39,90};
 .....
 func(x);
 .....
```

Qué tipo de información devuelve la función **`func`** ?.
Ninguna (void)

Cuál es la salida que se obtiene en el programa correspondiente al código propuesto para **`func`** ?.
Muestra por pantalla la suma del arreglo de 6 elementos. (con el código del cuadro)

5 • A continuación se declara un arreglo **a** de 10 elementos enteros. El elemento inicial **`x[0]`** se ubica en la dirección de memoria 000011E4:
int a[10]={110, 120, 130, 140, 150, 160, 170, 180, 190, 200};
Determine el valor que representan las expresiones siguientes:

X;	000011E4
(x+4);	01-02-03-04-05-06-07-08-09-10-11-12-13-14-15-16 E5,E6,E7,E8,E9,EA,EB,EC,ED,EF,F0,F1,F2,F3,F4 000011F4
*x;	110
*(x+3);	140
*x+3;	113

6. Utilizando notación de punteros, generar un arreglo lineal de N elementos numéricos enteros, con valores aleatorios entre 1000 y 1500, y muestre en pantalla la dirección de memoria del mayor elemento. N es un dato ingresado por el usuario.

```
//© Programación Orientada a Objetos- F.I.C.H. - U.N.L.
//Autor: Prof. Gerardo Sas.
//sitio web: http://e-fich.unl.edu.ar/moodle/
//Guia de estudios Nº 1 - Ejercicio Nº 6
#include <iostream>
#include <cstdlib>
using namespace std;

int main(int argc, char *argv[]) {
  int n, *mayor;
  cout<<"ingrese la cantidad de elementos: ";
  cin>>n;
  cout<<"Generando "<<n<<" elementos"<<endl;
  int *arreglo = new int[n];
  mayor = arreglo;
  for (int x= 0;x < n; x++){
     *(arreglo+x) = rand()%500+1000;
     cout<<"Arreglo["<<x<<"]= "<<*(arreglo+x)<<endl;
     if(*(arreglo+x)>*mayor)
          mayor= arreglo+x;
  }
  cout<<"La posicion del mayor ("<<*mayor<<") es: "<<mayor<<endl;
  delete(arreglo);
  return 0;
}
```

7. Amplíe el programa anterior para que luego de generar el arreglo aleatorio, permita ingresar un valor M que debe ser insertado en la posición 32 de dicho arreglo y muestre el vector modificado.

```
//© Programación Orientada a Objetos- F.I.C.H. - U.N.L.
//Autor: Prof. Gerardo Sas.
//sitio web: http://e-fich.unl.edu.ar/moodle/
//Guia de estudios Nº 1 - Ejercicio Nº 7
#include <iostream>
#include <cstdlib>
using namespace std;

int main(int argc, char *argv[]) {
  int n;
  cout<<"ingrese la cantidad de elementos: ";
  cin>>n;
  int p;
  do{
     cout<<"ingrese la Posicion de insercion: ";
     cin>>p;
  }while(p>n);
  int m;
  cout<<"ingrese el numero a insertar en la posicion "<<p<<": ";
  cin>>m;
  cout<<"Generando "<<n<<" elementos"<<endl;
  int *arreglo = new int(n+1);//agrego 1 posicion mas
  p--;//Porque comienza en 0
  for (int x= 0;x < n; x++){
     *(arreglo+x) = rand()%500+1000;
     cout<<"Arreglo["<<x<<"]= "<<*(arreglo+x)<<endl;
  }
  cout<<"Insertando numero "<<m<<" en posicion "<<p<<endl;
  for (int x = n+1;x > p; x--){
     *(arreglo+x) = *(arreglo+x-1);
  }
  *(arreglo+p)= m;
  for (int x= 0;x < n+1; x++){
          cout<<"Arreglo["<<x+1<<"]= "<<*(arreglo+x)<<endl;
  }
  delete(arreglo);
  return 0;
}
```

8. Modifique el ejercicio 6 para que el usuario pueda luego ingresar un entero C y se inserten esa cantidad de ceros al final del vector. Para realizar esto reserve dinámicamente un nuevo arreglo que contenga lugar para guardar los datos anteriores y los ceros que serán insertados, luego copie en la nueva memoria los datos del vector viejo y agregue los ceros. La memoria del primer arreglo debe ser liberada y el puntero del arreglo debe quedar apuntando a la nueva memoria con los ceros agregados. Finalmente, muestre el arreglo resultante.

```
//© Programación Orientada a Objetos- F.I.C.H. - U.N.L.
//Autor: Prof. Gerardo Sas.
//sitio web: http://e-fich.unl.edu.ar/moodle/
//Guia de estudios Nº 1 - Ejercicio Nº 8

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

int main(int argc, char *argv[]) {
  int n;
  cout<<"ingrese la cantidad de elementos: ";
  cin>>n;
  int c;
  cout<<"ingrese el numero de ceros a insertar: ";
  cin>>c;
  cout<<"Generando "<<n<<" elementos"<<endl;
  int *arreglo = new int(n);
  int t= n+c;
  int *aux = new int(t);
  for (int x= 0;x < n; x++){
     *(arreglo+x) = rand()%500+1000;
     cout<<"Arreglo["<<x<<"]= "<<*(arreglo+x)<<endl;
  }
  cout<<"Insertando ceros"<<endl;
  for (int x = 0;x <n; x++){
     *(aux+x) = *(arreglo+x);
     //cout<<"Arreglo["<<x+1<<"]= "<<*(aux+x)<<endl;
  }
  for(int x= n; x < n+c; x++){
     *(aux+x) = 0;
     //cout<<"Arreglo["<<x+1<<"]= "<<*(aux+x)<<endl;
  }
  arreglo = aux;
  for (int x = 0;x < t; x++){
     cout<<"Arreglo["<<x+1<<"]= "<<*(arreglo+x)<<endl;
  }
  delete(arreglo,aux);
  return 0;
}
```

9. Usando notación de punteros genere aleatoriamente una matriz de números reales de doble precisión de 10 filas por 6 columnas y determine e informe:

a) El promedio de la fila que el usuario ingrese como dato
b) La suma de cada columna

```
//© Programación Orientada a Objetos- F.I.C.H. - U.N.L.
//Autor: Prof. Gerardo Sas.
//sitio web: http://e-fich.unl.edu.ar/moodle/
//U1E9_CV
#include <iostream>
#include <iomanip>
#include <cstdlib>
using namespace std;
int main(void){
  const int filas=10;
  const int columnas=6;
  int f,c,i,j;
  double prom=0;
  //Declaro un arreglo dinamico de 10 filas y 6 columnas
  double (*a)[6]= new double[10][6];
  //recorro el arreglo de 0 a 9 y las columnas de 0 a 5, asignandole valores
  for (i=0;i<filas;i++)
  {for (j=0;j<columnas;j++)
  {*(*(a+i)+j)= (rand()%(100));
  cout <<setw(8)<< *(*(a+i)+j) << " ";//Muestro por pantalla
  };
  cout<<endl;
  };
  cout<<endl<<endl;
  //Terminada la carga ingreso valores para fila
  cout<<"Ingrese la fila a promediar (1-10): ";
  cin>>f;
  cout<<endl;
  prom=0;
  for (j=0;j<columnas;j++)
  prom += *(*(a+(f-1))+j);//Observe que f es 1 menor.
  prom /= columnas;//Calculo promedio
  cout<<"El promedio de la fila: "<<f<<" es "<<prom<<endl;
  for (c=0; c<columnas; c++)//Sumo cada columna
  {int sum=0;
  for (i=0;i<filas;i++)
  sum += *(*(a+i)+c);
  cout<<"La suma de la columna: "<<c+1<<" es "<<sum<<endl;
  };
  delete []a;//Libero la memoria .-
}//Fin del programa.-
```

10. Escriba una función que utilice punteros para buscar e informar la dirección de un entero dentro de un arreglo. Se pasan como parámetros el arreglo y el entero a buscar. Si el dato no se encuentra, devolver la dirección de memoria nula (NULL)

```
//© Programación Orientada a Objetos- F.I.C.H. - U.N.L.
//Autor: Prof. Gerardo Sas.
//sitio web: http://e-fich.unl.edu.ar/moodle/
//U1E10_CV
#include <iostream>
#include <iomanip>
#include <stdlib.h>
#include <time.h>

using namespace std;
//xx es la direccion del arreglo.
//num es el  numero a buscar.
//lar es el largo del arreglo.
int *buscar(int *xx, int num, int lar)
{ int c;
for (c=0;c<lar;c++)
  if (num == *(xx+c)) return (xx+c);
//devuelve la direccion del numero encontrado o NULL
return NULL;}
//*** Cuerpo del programa principal ***
int main(){
  //Genero el vector con el metodo de punteros
  int i, largo= 10;
  //int *x=(int*) malloc( 10*sizeof(int) ); //asi seria con malloc()
  int *x= new int [largo];        //Cargo el vector con datos menores que 100
  srand( time(NULL));//Inicializa rand con un valor tomado del reloj
  for(i=0; i<10; i++)//carga el vector
     *(x+i)=rand()%(100);
  //Muestro el dato y la posicion de memoria que ocupa, casteo a long
  for( i=0; i<10; i++)//casteo a long la direccion de memoria
     cout<<setw(5)<< *(x+i)<<"   "<<long(x+i)<<endl;
  //Ingreso el dato a buscar
  int numero;   cout<<endl<<endl;
  cout<<"Ingrese el numero a buscar: ";
  cin>>numero; cout<<endl;
  //Llamo a la funcion buscar y dependiendo del resultado muestro el msj
  int *p= buscar(x,numero,largo);
  if  (p != NULL) //casteo a long la direccion de memoria
     cout<<"Direccion del dato encontrado: "<<long(p)<<"  Dato: "<<*p<<endl;
  else
     cout<<"NULL - Dato no encontrado";
  cout<<endl<<endl;
  delete []x;//Libero la memoria
}//Fin del programa
```

11. Escriba la función Plural cuyo prototipo se expone a continuación. Debe retornar en el mismo parámetro, el plural correspondiente a la palabra. Considere agregar 's si la palabra termina en vocal y 'es si termina en consonante. Utilice notación de punteros para realizar todas las operaciones sobre la cadena

void plural (char *p);

```
//© Programación Orientada a Objetos- F.I.C.H. - U.N.L.
//Autor: Prof. Gerardo Sas.
//sitio web: http://e-fich.unl.edu.ar/moodle/
//U1E10_CV
#include <iostream>
#include <iomanip>
#include <stdlib.h>
#include <time.h>

using namespace std;
//xx es la direccion del arreglo.
//num es el  numero a buscar.
//lar es el largo del arreglo.
int *buscar(int *xx, int num, int lar)
{ int c;
for (c=0;c<lar;c++)
  if (num == *(xx+c)) return (xx+c);
//devuelve la direccion del numero encontrado o NULL
return NULL;}
//*** Cuerpo del programa principal ***
int main(){
  //Genero el vector con el metodo de punteros
  int i, largo= 10;
  //int *x=(int*) malloc( 10*sizeof(int) ); //asi seria con malloc()
  int *x= new int [largo];      //Cargo el vector con datos menores que 100
  srand( time(NULL));//Inicializa rand con un valor tomado del reloj
  for(i=0; i<10; i++)//carga el vector
  *(x+i)=rand()%(100);
  //Muestro el dato y la posicion de memoria que ocupa, casteo a long
  for( i=0; i<10; i++)//casteo a long la direccion de memoria
  cout<<setw(5)<< *(x+i)<<"   "<<long(x+i)<<endl;
  //Ingreso el dato a buscar
  int numero;          cout<<endl<<endl;
  cout<<"Ingrese el numero a buscar: ";
  cin>>numero; cout<<endl;
  //Llamo a la funcion buscar y dependiendo del resultado muestro el msj
  int *p= buscar(x,numero,largo);
  if  (p != NULL) //casteo a long la direccion de memoria
  cout<<"Direccion del dato encontrado: "<<long(p)<<"  Dato: "<<*p<<endl;
  else
  cout<<"NULL - Dato no encontrado";
  cout<<endl<<endl;
  delete []x;//Libero la memoria
}//Fin del programa
```