Segundo Certamen
Tiempo: 100 minutos

1. Se tiene la siguiente aplicación Java. 30

   a) Genere la versión Applet equivalente.
   b) Muestre un archivo html que incluya esta Applet.
   c) ¿Con qué comando usted prueba su Applet usando appletviewer?
   d) ¿Cómo debe ser modificada para que vía el html se defina el texto del botón?
   e) ¿Cuál es el html correspondiente para que el texto del botón sea "Otro Botón"?

Obs: En lugar de re-copiar lo que se mantiene igual, indíquelo.

En archivo CreaBotones.java tenemos:

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;
public class CreaBotones
{
  public static void main(String[] args)
  {
    CreaBotonesFrame frame = new CreaBotonesFrame();
    frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    frame.setVisible(true);
  }
}
class CreaBotonesFrame extends JFrame
{
  public CreaBotonesFrame()
  {
    setTitle("Crea Botones");
    setSize(250, 150);
    CreaBotonesPanel panel = new CreaBotonesPanel();
    Container contentPane = getContentPane();
    contentPane.add(panel);
  }
}
class CreaBotonesPanel extends JPanel
{
  public CreaBotonesPanel()
  {
    creaBoton();
  }
  private void creaBoton()
  {
    JButton boton = new JButton("Crea Botón");
    boton.addActionListener( new ActionListener (){
      public void actionPerformed(ActionEvent event) {
        creaBoton();
      };
     });
    add(boton);
    validate();
  }
}
```

a) // 6 pts

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;

public class CreaBotonesApplet extends JApplet {
  public void init() {
    CreaBotonesPanel panel = new CreaBotonesPanel();
    Container contentPane = getContentPane();
    contentPane.add(panel);
  }
}
class CreaBotonesPanel extends JPanel {
  public CreaBotonesPanel() {
    creaBoton();
  }
  private void creaBoton() {
    JButton boton = new JButton("Crea Botón");
    boton.addActionListener( new ActionListener (){
       public void actionPerformed(ActionEvent event) {
         creaBoton();
       };
      });

    add(boton);
    validate();
  }
}
```

b) Sea este CreaBotonesApplet.html // 6 pts.

```
<html>
<body>
<applet code="CreaBotonesApplet.class"
   width="300" height="400">
</applet>
</body>
</html>
```

c) Estando en la línea de comando: $ appletviewer CreaBotonesApplet.html // 6 pts.

d) 6 pts.

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import javax.swing.*;

public class CreaBotonesAppletVar extends JApplet {
  public void init()   {
     String textoBoton=getParameter("Texto Boton");
    CreaBotonesPanel panel = new CreaBotonesPanel(textoBoton);
    Container contentPane = getContentPane();
    contentPane.add(panel);
  }
}
class CreaBotonesPanel extends JPanel {
  public CreaBotonesPanel(String texto)  {
     this.texto=texto;
    creaBoton();
  }
  private void creaBoton() {
    JButton boton = new JButton(texto);
    boton.addActionListener( new ActionListener (){
       public void actionPerformed(ActionEvent event) {
         creaBoton();
       };
      });

    add(boton);
    validate();
  }
  private String texto;
}
```

e) // 6pts.

```
<html>
<body>
<applet code="CreaBotonesAppletVar.class"
   width="300" height="400">
<param name="Texto Boton"  value="Otro Botón"/ >
</applet>
</body>
</html>
```

2.- 5 pts cada respuesta

20

a) ¿Qué diferencia existe entre ligado dinámico en Java y ligado dinámico en C++?
La principal diferencia es que en Java el ligado dinámico es la semántica normal en cambio en C++ no lo es y debe usarse el modificador (o calificador) virtual para señalar ligado dinámico.

b) ¿Qué diferencia existe en C++ entre clases (class) y estructuras (struct)?
La única diferencia es que en clases la omisión del calificador de visibilidad (o alcance) equivale a poner todo privado en cambio en estructuras la omisión del calificador de visibilidad equivale a public.

c) ¿Mencione y explique dos interpretaciones para la palabra reservada "const"?
const se usa para indicar:
i) que un campo miembro (atributo) o parámetro mantiene su valor constante. En este caso se asegura que cierto valor de memoria no será alterado.
ii) que un método no altera ninguno de los atributos del objeto. En este caso el la implementación del método está impedido de cambiar los atributos del objeto.

d) C++ permite sobrecargar operadores ~~en~~ usando métodos (funciones miembros de una clase) o como funciones globales. Dé un ejemplo en que estemos obligados a usar esta segunda forma de sobrecarga.

Un ejemplo de esto es cuando deseamos enviar a la salida estándar un objeto usando un el código:
cout << objeto;
En este caso no sirve sobrecargar el operador << en la clase del objeto, pues eso funcionaría si la instrucción fuera: objeto << cout; Para lograr lo natural indicador antes deberíamos sobremontar el operador << en la clase ostream que es una clase estándar del lenguaje. Luego para poder mantener la sintaxis natural es debemos hacer la sobrecargo vía una función global. (Esto se explicó en clase)

3.- 20

Implemente la función template o platilla ***suma(....)*** que retorna la suma "+" de los elementos de un arreglo. Su creación permitiría usarla en situaciones como:

```
// aquí se omite código en que se declara A, F, n y m
int s_i = suma( A, n);  // donde A es un arreglo de enteros y n el tamaño del arreglo.
float s_f = suma (F, m); // cuando F es un arreglo de float y m es el tamaño del arreglo.
```

Para que esta plantilla pueda ser usada con arreglos de objetos ¿qué requerimiento debe satisfacer la clase de estos objetos?
Obs: La suma ~~señala~~ es una operación cuyo valor es del mismo tipo que los operandos.

```
template <class T>  // 4pts.
T suma ( T const data[], int n) {   // 2+2+2 pts.
  T sum = 0;  // require asignación de 0  // cuerpo 6 pts
  for (int i=0; i<n; i++)
    sum+=data[i];
  return sum;
};
```

La clase de los objetos del arreglo debe incluir la sobrecarga del operador +=, del operador asignación de un objeto de la misma clase, y asignación de entero. // 4 pts.

4.- Se tiene la siguiente clase incompleta: 30

```
class ArregloDinamico {
```

```
public:
   ArregloDinamico(int size);
  // otros métodos pedidos
private:
   ArregloDinamico(); // para que otros no puedan crear instancias de tamaño indefinido
   int * arreglo;
   int size;
}
```

Incorpore e implemente el constructor indicado, el destructor, y sobrecarga operador asignación. Si fuera necesario algo, además de estas funciones miembros, agréguelas en la definición e implementación para el correcto funcionamiento de la clase.

Cuando hay atributos punteros se debe implentar operador asignación, constructor copia y destructor. Esto debido a la semántica de C++ para hacer la copia o asignación de objetos. El no hacerlo genera errores en los programas por cambios inesperados de los objetos asignados. Esto se explicó en clases.

En archivo ArregloDinamico.h // 2pts

```
#ifndef ARREGLO_DINAMICO  // esto es opcional
#define ARREGLO_DINAMICO
class ArregloDinamico {
public:
  ArregloDinamico(int size);
  ArregloDinamico(const ArregloDinamico &a); // constructor copia   4 pts.
  const ArregloDinamico & operator=(const ArregloDinamico & a); //sobre carga 3 pts.
                                                                 // operador asignación
  ~ArregloDinamico(); // destructor 3 pts.

private:
    ArregloDinamico();  // para que otros no puedan
                        // crear instancias de tamaño indefinido
    int * arreglo;
    int size;
};
#endif
```

En archivo ArregloDinamico.cpp // 2pts.

```
#include <cstdlib>
#include "ArregloDinamico.h"    // 2pts, en general consideré 6 pts por
                                // estructura global del programa.

/***************Constructores ************/
ArregloDinamico:: ArregloDinamico(){
  arreglo = NULL;
  size=0;
}
ArregloDinamico::ArregloDinamico(int size){  // 3 pts
  arreglo = new int[size];
  this->size=size;
}
ArregloDinamico::ArregloDinamico(const ArregloDinamico &a){ // 4 pts.
  if (arreglo != NULL) delete [] arreglo;
  size=a.size;
 arreglo = new int[size];
  for(int i=0; i<size; i++)
    arreglo[i]=a.arreglo[i];
}
/********** Sobre carga operador asignación *****************/
const ArregloDinamico & ArregloDinamico::operator=(const ArregloDinamico & a){//
4pts.
```

```
    if (arreglo != NULL) delete [] arreglo;
    size=a.size;
    arreglo = new int[size];
    for(int i=0; i<size; i++)
      arreglo[i]=a.arreglo[i];
    return *this;
  }

/*************** Destructores *********************/
ArregloDinamico::~ArregloDinamico(){  // 3 pts.
 if (arreglo != NULL) delete[] arreglo;
}
```