INSTITUTO UNIVERSITARIO DE TECNOLOGÍA DE ADMINISTRACIÓN INDUSTRIAL
Extensión Puerto La Cruz

REGISTRO DE EVALUACIÓN

AUTORIZADO Coordinación de la Especialidad	AUTORIZADO Coordinación de Evaluación

EVALUACIÓN Nº 02

Prueba: Objetiva____ Ensayo_____ Mixta _____ Práctica____ Oral _____

Modalidad:

Trabajo: Campo_____ Investigación_____ Grupal_____ Individual _____

NOTA

Periodo: 2021-I_ **Fecha:** _____ **Nivel** _____ **Sección** __01___ **Ponderación** 100% **Objetivo** _____

Especialidad: INFORMÁTICA **Unidad Curricular: PROGRAMACIÓN IV** **Facilitador: ROYMER CENTENO.**

Estudiante: Apellidos y Nombres: Solangel Rodríguez **C.I.** 30.305.133

RECOMENDACIONES:

Lea cuidadosamente las instrucciones ☛ Medite su Respuesta ☛ Cuide su Redacción y Ortografía
Pendiente de la fecha y la hora límite para la entrega de esta evaluación. Sea responsable la Coordinación de Evaluación no se hace responsable por entregas de evaluaciones retardadas.
Consultas o dudas sobre esta evolución, hacerlo al docente por la sección de comentarios de Classroom
Cualquier actitud de su parte, que comprometa la confiabilidad de la prueba será sancionada.

instrucciones:

1. El presente examen debe de ser adjuntado con sus acordadas respuestas.
2. La realización del examen será en lápiz grafito deberá ser escaneado o fotografiado de manera Legible para ser adjuntado a la plataforma.
3. El archivo a subir debe ser uno sólo en formato .pdf y este debe contener tanto este examen original junto sus respuestas
RECOMENDACIÓN : saque captura de pantalla al examen y adjuntelo como imagen dentro del archivo junto con sus respuestas, de ser publicado en github debe realizar la invitación correspondiente para su evaluaciom
4. No se aceptarán entregas tardías sin justificación previa y comprobable ante la coordinación de evaluaciones.
5. El incumplimiento de alguna de las normas acarreará la suspensión parcial o total de la evaluación

PRUEBA 2 PROGRAMACIÓN IV.

1. Realizar un programa en c++ obedeciendo a la siguiente argumentación:

1.1Defina una clase Forma que tenga los siguientes miembros de datos: -Color -Coordenada del centro de la forma (objeto Punto) -Nombre de la forma (char *) Y, al menos, las siguientes funciones miembro: -Imprimir -Obtener y cambiar el color -Mover la forma (o sea, su centro) Defina una clase derivada Rectángulo que tenga los siguientes miembros como datos: -Lado menor. -Lado mayor. Y, al menos, las siguientes funciones miembro: -Imprimir. Debe imprimir qué se trata de un rectángulo mostrando su nombre, color, centro y lado. Debería usarse la función Imprimir de la clase base para realizar parte de este trabajo. -Calcular el área (lado menor * lado mayor). -Calcular el perímetro. (2 * lado menor + 2 * lado mayor). -Cambiar el tamaño del rectángulo. Recibe como parámetro un factor de escala. Así, por ejemplo, si el factor vale 2, el rectángulo duplicará su tamaño y si es 0,5 se reducirá a la mitad.

puntualidad: 10%

Se evaluará:
b)Utilización de las herramientas provistas en clase 10 %
c)Implementación del sistema de control de versiones para su código 10 %
a) la organización del código (Incluye utilización de estructuras de control y organización de los métodos según las prácticas presentadas en clase) (25%)

2. Defina una clase Cuadrado derivada de la clase Rectángulo del ejercicio anterior.

Se evaluará:
b)Utilización de las herramientas provistas en clase 10 %
c)Implementación del sistema de control de versiones para su código 10 %
a) la organización del código (Incluye utilización de estructuras de control y organización de los métodos según las prácticas presentadas en clase) (25%)

PRIMERA FORMA DE SOLUCIÓN
DOS ARCHIVOS
"Clases.h" con la declaración de las clases
"Evlauacion2.cpp" con la definición de las clases y clase Main
Con valores fijos de las propiedades de los objetos

Archivo de Declaración de Clases
"Clases.h"

```
// Ejercicio 1: Clase Punto ******************************************

using namespace std;

const double PI=3.1416;

class Punto {
    public:
       int x;
       int y;
       Punto() {x = 0; y=0;};
       Punto(int i, int j) {x = i; y = j;}
};

// Ejercicio 1: Clase Forma ******************************************

class Forma  {
    protected:
       string color;
       Punto centro;
       string nombre;
    public:
       Forma(string);
       Forma(string, Punto);
       virtual ~Forma() {};
       virtual void imprimir();
       virtual int area() {return(0);};
       string ObtenerColor();
       void EstablecerColor(string);
       void MoverCentro(Punto);
       friend ostream & operator << (ostream & salida, const Forma & f);
};

// Ejercicio 1: Clase Derivada Rectángulo ***********************************

class Rectangulo: public Forma {
    protected:
       int lado_menor;
       int lado_mayor;
    public:
       Rectangulo(string, int, int);
       Rectangulo(string, Punto, int, int);
       virtual void imprimir();
       virtual int area();
       virtual int perimetro();
```

```
        void CambiarTamano(double);
        friend ostream & operator << (ostream & salida, const Rectangulo & r);
};

// Ejercicio 2: Clase derivada Cuadrado de la clase rectángulo

class Cuadrado: public Rectangulo {
    public:
    Cuadrado(string, int);
    Cuadrado(string, Punto, int);
    virtual void imprimir();
    friend ostream & operator << (ostream & salida, const Cuadrado & col);
};
```

Archivo de Implementación
"Evaluacion2.cpp"

```
// Programa Principal Ejercicio 1 y 2-----------------------------------------------

#include <iostream>

#include "Clases.h"

//Definiciones de funciones de la clase Forma ----------------------------------------

Forma::Forma(string nomb)
{
    nombre = nomb;
}

Forma::Forma(string nomb, Punto pun): centro (pun)
{
    nombre = nomb;
}

void Forma::imprimir()
{
    cout << endl << nombre << " tiene un color " << color;
    cout << " con centro en el punto (" << centro.x << ", " << centro.y  << ")";
}

ostream & operator << (ostream & salida, const Forma & f)
{
    salida <<  f.nombre << " de color " << f.color << " con centro (";
    salida <<  f.centro.x << ", " << f.centro.y  << ")" << endl;
    return(salida);
}

void Forma::EstablecerColor(string col)
{
    color = col;
}

string Forma::ObtenerColor()
{
    return(color);
}

void Forma::MoverCentro(Punto pun)
{
    centro.x += pun.x;
    centro.y += pun.y;
}


//Definiciones de funciones de la clase derivada rectángulo---------------------------
```

```
Rectangulo::Rectangulo(string nomb, int menor, int mayor): Forma(nomb)
{
    lado_menor = menor;
    lado_mayor = mayor;
}

Rectangulo::Rectangulo(string nomb, Punto pun, int menor, int mayor): Forma(nomb, pun)
{
    lado_menor = menor;
    lado_mayor = mayor;
}

int Rectangulo::area()
{
    return (lado_menor * lado_mayor);
}

int Rectangulo::perimetro()
{
    return (2*lado_menor + 2*lado_mayor);
}

void Rectangulo::CambiarTamano(double factor_escala)
{
    lado_menor = (int) (lado_menor * factor_escala);
    lado_mayor = (int) (lado_mayor * factor_escala);
}

void Rectangulo::imprimir()
{
    Forma::imprimir();
}

ostream & operator << (ostream & salida, const Rectangulo & r)
{
    salida << Forma(r);
    salida << " y de lados " << r.lado_menor << " y " << r.lado_mayor << endl;
    return(salida);
}

//Definiciones de funciones de la clase derivada Cuadrado ----------------------------

Cuadrado::Cuadrado(string nomb, int lado): Rectangulo(nomb, lado, lado)
{
    ;
}

Cuadrado::Cuadrado(string nomb, Punto pun, int lado): Rectangulo(nomb, pun, lado, lado)
{
    ;
}

void Cuadrado::imprimir()
{
    Forma::imprimir();
}
```

```
ostream & operator << (ostream & salida, const Cuadrado & col)
{
    salida << Forma(col);
    salida << " y de lado " << col.lado_menor << endl;
    return(salida);
}

//Clase main ------------------------------------------------------------------------
int main ()
{
    //creación de objetos
    Punto pun(3,3);
    Forma forma("Forma", pun);
    Rectangulo rec("Rectangulo", pun, 2, 4);
    Cuadrado cuad("Cuadrado", pun, 4);
    cout << endl << "SE CREARON LOS OBJETOS PUNTO, FORMA, RECTANGULO Y CUADRADO";
    cout << endl << "-----------------------------------------------------------" << endl;

    //llamada a funciones de los objetos
    rec.EstablecerColor("azul");
    forma.EstablecerColor("negro");
    cuad.EstablecerColor("rojo");

    //Imprimiendo el estado de los objetos
    cout << endl << "DESCRIPCION DEL ESTADO DE LOS OBJETOS";
    cout << endl << "-------------------------------------" << endl;
    rec.imprimir();
    forma.imprimir();
    cuad.imprimir();
    cout << endl;

    // Cálculo del área de los objeto y prueba de MoverCentro y CambiarTamano
    cout << endl << "AREA DEL RECTANGULO DE DIFERENTES TAMANOS Y DEL CUADRADO";
    cout << endl << "-------------------------------------------------------" << endl;
    rec.MoverCentro(Punto(2, 2));
    cout << endl << "El area del Rectangulo 1 es " << rec.area();
    rec.CambiarTamano(3);
    cout << endl << "El area del Rectangulo 2  es " << rec.area();
    rec.CambiarTamano(0.5);
    cout << endl << "El area del Rectangulo 3 es " << rec.area();
    rec.CambiarTamano(2);
    rec.imprimir();
    cout << endl << "El area del Cuadrado es " << cuad.area() << endl;

    // Cálculo del perímetro de las figuras
    cout << endl << "PERIMETRO DEL RECTANGULO Y DEL CUADRADO";
    cout << endl << "---------------------------------------" << endl;
    cout << endl << "El perimetro del Rectangulo es " << rec.perimetro();
    cout << endl << "El perimetro del Cuadrado es " << cuad.perimetro();

    cout << endl;
    cout << endl;
}
```

SEGUNDA FORMA DE SOLUCIÓN
UN SOLO ARCHIVO
"Evaluacion22.cpp": Con declaración y definición de clases y Clase Main
Solicitando las propiedades de los objetos

```
#include <iostream>

using namespace std;

// Ejercicio 1: Declaración y definición de Clase Punto----------------------------------
--

class Punto {
    public:
        int x;
        int y;
        Punto() {x = 0; y=0;};
        Punto(int i, int j) {x = i; y = j;}
};

// Ejercicio 1: Declaración de Clase Forma  ------------------------------------------
-

class Forma  {
    protected:
        string color;
        Punto centro;
        string nombre;
    public:
        Forma(string);
        Forma(string, Punto);
        virtual ~Forma() {};
        virtual void imprimir();
        virtual int area() {return(0);};
        string ObtenerColor();
        void EstablecerColor(string);
        void MoverCentro(Punto);
        friend ostream & operator << (ostream & salida, const Forma & f);
};

//Ejercicio 1: Definiciones de funciones de la clase Forma ----------------------------

Forma::Forma(string nomb)
{
    nombre = nomb;
}

Forma::Forma(string nomb, Punto pun) : centro (pun)
{
    nombre = nomb;
}
```

```
void Forma::imprimir()
{
    cout << endl << nombre << " tiene un color " << color;
    cout << " con centro en el punto (" << centro.x << ", " << centro.y  << ")";
}

ostream & operator << (ostream & salida, const Forma & f)
{
    salida <<  f.nombre << " de color " << f.color << " con centro (";
    salida <<  f.centro.x << ", " << f.centro.y  << ")" << endl;
    return(salida);
}

void Forma::EstablecerColor(string col)
{
    color = col;
}

string Forma::ObtenerColor()
{
    return(color);
}

void Forma::MoverCentro(Punto pun)
{
    centro.x += pun.x;
    centro.y += pun.y;
}


// Ejercicio 1: Declaración de Clase Derivada Rectángulo --------------------------------

class Rectangulo: public Forma {
    protected:
        int lado_menor;
        int lado_mayor;
    public:
        Rectangulo(string, int, int);
        Rectangulo(string, Punto, int, int);
        virtual void imprimir();
        virtual int area();
        virtual int perimetro();
        void CambiarTamano(double);
        friend ostream & operator << (ostream & salida, const Rectangulo & r);
};

// Ejercicio 1: Definiciones de funciones de la clase derivada rectángulo----------------

Rectangulo::Rectangulo(string nomb, int menor, int mayor) : Forma(nomb)
{
    lado_menor = menor;
    lado_mayor = mayor;
}

Rectangulo::Rectangulo(string nomb, Punto pun, int menor, int mayor) : Forma(nomb, pun)
{
```

```
    lado_menor = menor;
    lado_mayor = mayor;
}

int Rectangulo::area()
{
    return (lado_menor * lado_mayor);
}

int Rectangulo::perimetro()
{
    return (2*lado_menor + 2*lado_mayor);
}

void Rectangulo::CambiarTamano(double factor_escala)
{
    lado_menor = (int) (lado_menor * factor_escala);
    lado_mayor = (int) (lado_mayor * factor_escala);
}

void Rectangulo::imprimir()
{
    Forma::imprimir();
}

ostream & operator << (ostream & salida, const Rectangulo & r)
{
    salida << Forma(r);
    salida << " y de lados " << r.lado_menor << " y " << r.lado_mayor << endl;
    return(salida);
}

// Ejercicio 2: Declaración de Clase derivada Cuadrado

class Cuadrado: public Rectangulo {
    public:
        Cuadrado(string, int);
        Cuadrado(string, Punto, int);
        virtual void imprimir();
        friend ostream & operator << (ostream & salida, const Cuadrado & col);
};

//Ejercicio 2: Definiciones de funciones de la clase derivada Cuadrado ------------------

Cuadrado::Cuadrado(string nomb, int lado) : Rectangulo(nomb, lado, lado)
{
    ;
}

Cuadrado::Cuadrado(string nomb, Punto pun, int lado) : Rectangulo(nomb, pun, lado, lado)
{
    ;
}

void Cuadrado::imprimir()
{
```

```
    Forma::imprimir();
}

ostream & operator << (ostream & salida, const Cuadrado & col)
{
    salida << Forma(col);
    salida << " y de lado " << col.lado_menor << endl;
    return(salida);
}

//Clase main ------------------------------------------------------------------------

int main ()
{
    int coordenada_x, coordenada_y, lmenor, lMayor,l;
    string colfor, colrec, colcuad;
    double factor;

    cout << endl << "CREACION DE LOS OBJETOS: " << endl;
    cout << "-----------------------" << endl;

    //1. Creación de un objeto Punto (pun)
    cout << endl << "Ingrese la coordenada x del centro de los objetos: " << endl;
    cin  >> coordenada_x;
    cout << "Ingrese la coordenada y del centro de los objetos: " << endl;
    cin  >> coordenada_y;
    Punto pun(coordenada_x,coordenada_y);
    cout << "Se creo el objeto Punto" << endl;

    //2. Creación de un objeto Forma (forma)
    Forma forma("Forma", pun);
    cout << endl << "Se creo el objeto Forma" << endl;

    //3. Creación de un objeto Rectapunto (rec)
    cout << endl << "Ingrese lado menor del rectangulo: " << endl;
    cin  >> lmenor;
    cout << "Ingrese lado mayor del rectangulo: " << endl;
    cin  >> lMayor;
    Rectangulo rec("Rectangulo", pun, lmenor, lMayor);
    cout << "Se creo el objeto Rectangulo" << endl;

    //4. Creación de un objeto Cuadrado (cuad)
    cout << endl << "Ingrese lado del cuadrado: " << endl;
    cin  >> l;
    Cuadrado cuad("Cuadrado", pun, l);
    cout << "Se creo el objeto Cuadrado" << endl;

    //5. Establecer el color de los objetos creados
    cout << endl << "ESTABLECER EL COLOR DE LOS OBJETOS";
    cout << endl << "---------------------------------" << endl;
    cout << "Ingrese color del objeto forma: " << endl;
    cin  >> colfor;
    forma.EstablecerColor(colfor);
    cout << "Ingrese color del objeto rectangulo: " << endl;
    cin  >> colrec;
    rec.EstablecerColor(colrec);
```

```
    cout << "Ingrese color del objeto cuadrado: " << endl;
    cin  >> colcuad;
    cuad.EstablecerColor(colcuad);

    //6. Imprimiendo el estado de los objetos
    cout << endl << "DESCRIPCION DEL ESTADO DE LOS OBJETOS";
    cout << endl << "-------------------------------------" << endl;
    rec.imprimir();
    forma.imprimir();
    cuad.imprimir();
    cout << endl;

    //7. Cálculo del área de los objetos rectángulo y cuadrado
    cout << endl << "AREA DEL RECTANGULO Y DEL CUADRADO";
    cout << endl << "----------------------------------" << endl;
    cout << endl << "El area del Rectangulo es " << rec.area();
    cout << endl << "El area del Cuadrado es " << cuad.area() << endl;

    //8. Cálculo del perímetro de los objetos rectángulo y cuadrado
    cout << endl << "PERIMETRO DEL RECTANGULO Y DEL CUADRADO";
    cout << endl << "---------------------------------------" << endl;
    cout << endl << "El perimetro del Rectangulo es " << rec.perimetro();
    cout << endl << "El perimetro del Cuadrado es " << cuad.perimetro() << endl;

    //9. Prueba de la función MoverCentro y CambiarTamano del Rectangulo
    cout << endl << "CAMBIAR CENTRO Y TAMANO DEL RECTANGULO";
    cout << endl << "--------------------------------------" << endl;

    cout << "Ingrese el cambio de x del centro del rectangulo: " << endl;
    cin  >> coordenada_x;

    cout << "Ingrese el cambio de y del centro del rectangulo: " << endl;
    cin  >> coordenada_y;

    rec.MoverCentro(Punto(coordenada_x,coordenada_y));

    cout << "Ingrese el factor de cambio de tamano del rectangulo: " << endl;
    cin  >> factor;
    rec.CambiarTamano(factor);
    rec.imprimir();
    cout << endl << "El area del nuevo rectangulo es " << rec.area();

    cout << endl;
    cout << endl << "PRESIONE UNA TECLA PARA CONTINUAR...." << endl;
    cin.get();
}
```