UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

Programación III
Práctica Calificada 1
Pregrado
2023-1
Profesor: José A. Chávez Álvarez
Lab 1.01

Indicaciones específicas:

- Esta evaluación contiene 8 páginas (incluyendo esta página) con 3 preguntas. El total de puntos son 20.
- El tiempo límite para la evaluación es 100 minutos.
- Cada pregunta deberá ser respondida en un solo archivo con el número de la pregunta.
  - `p1.cpp`
  - `p2.cpp`
  - `p3.cpp`
- Deberás subir estos archivos directamente a www.gradescope.com, uno en cada ejercicio. También puedes crear un .zip

Calificación:

Tabla de puntos (sólo para uso del professor)

Question	Points	Score
1	6	
2	7	
3	7	
Total:	20	

1. (6 points) **Problema 1**
   Implementar un programa que simule un vehículo detector de obstáculos. Para ello el programa debe tener los siguientes requerimientos:

   - Implementar la clase `VehiculoAutonomo` con dos atributos: posición en el eje X y velocidad en el eje X. El Constructor debe configurar estos atributos.
   - Implementar la clase `Obstaculo` con un atributo: posición en el eje X.
   - Implementar la sobrecarga al operador `>>` para simular el movimiento del vehículo.
   - Cuando el vehículo se encuentre a una distancia menor o igual a la distancia recorrida en una iteración, el vehículo debe girar en la dirección opuesta.

   El ejercicio debe ser validado con el siguiente bloque de código:

```
int main(){
    // posicion eje X = 0, velocidad = +1.3
    // ====================================
    // En cada iteracion el vehiculo se mueve +1.3. Por
    // ejemplo:
    // En tres iteraciones el vehiculo se encontrara en
    // la posicion x = 3.9

    VehiculoAutonomo v(0, 1.3);
    Obstaculo o1(10);

    // Avanzar
    v>>o1;

    Obstaculo o2(-20);
    v>>o2;
    return 0;
}
```

Para lo cual, el resultado del código debe ser el siguiente:

```
Avanzando... Posicion: 1.3
Avanzando... Posicion: 2.6
Avanzando... Posicion: 3.9
Avanzando... Posicion: 5.2
Avanzando... Posicion: 6.5
Avanzando... Posicion: 7.8
Avanzando... Posicion: 9.1
Obstaculo detectado!
Girando...
Avanzando... Posicion: 7.8
Avanzando... Posicion: 6.5
Avanzando... Posicion: 5.2
Avanzando... Posicion: 3.9
Avanzando... Posicion: 2.6
Avanzando... Posicion: 1.3
Avanzando... Posicion: 0.0
Avanzando... Posicion: -1.3
Avanzando... Posicion: -2.6
Avanzando... Posicion: -3.9
Avanzando... Posicion: -5.2
Avanzando... Posicion: -6.5
Avanzando... Posicion: -7.8
Avanzando... Posicion: -9.1
Avanzando... Posicion: -10.4
Avanzando... Posicion: -11.7
Avanzando... Posicion: -13
Avanzando... Posicion: -14.3
Avanzando... Posicion: -15.6
Avanzando... Posicion: -16.9
Avanzando... Posicion: -18.2
Avanzando... Posicion: -19.5
Obstaculo detectado!
Girando...
```

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Clases y Objetos	Desarrolla las clases con todos los atributos y métodos requeridos, se realiza un adecuado encapsulamiento, se utiliza adecuadamente los constructores que permiten copiar y asignar objetos, se realiza las sobrecargas adecuadas, se realiza un uso adecuado de la memoria dinámica (`6pts`)	Desarrolla las clases con todos los atributos y métodos requeridos, se realiza un adecuado encapsulamiento, se realiza las sobrecargas adecuadas, se realiza un uso adecuado de la memoria (`4pts`)	Desarrolla las clases con todos los atributos y métodos requeridos, se realiza las sobrecargas adecuadas, errores en el funcionamiento pasa algunas pruebas (`2pts`).	Contiene errores los cuales hacen que no funcione el programa (`1pts`)

2. (7 points) **Problema 2**
Implemente una jerarquía de clases, de tal modo que se pueda *dibujar* tres distintos objetos heredados con la misma función.

El ejercicio debe ser validado con el siguiente bloque de código:

```
// Implemente aqui las clases y su relacion de herencia, si
// es que existe:
// - Figure (Clase Base)
// - Circle
// - Rectangle
// - Triangle
// - Point2d

void print(Figure* f){
    f->print();
}

int main(){
    Figure* f[3];
    f[0] = new Circle(Point2d(1, 2), 3);
    f[1] = new Rectangle(Point2d(1, 2), Point2d(3, 4));
    f[2] = new Triangle(Point2d(1, 2), Point2d(3, 4),
       Point2d(5, 6));
    for(int i = 0; i < 3; i++){
        print(f[i]);
    }
    return 0;
}
```

Para lo cual, el resultado del código debería ser el siguiente:

```
Circle: (1, 2), radio: 3
Rectangle: (1, 2), (3, 4)
Triangle: (1, 2), (3, 4), (5, 6)
```

La rúbrica para esta pregunta es:

Criterio	Excelente	Adecuado	Mínimo	Insuficiente
Clases y Objetos	Desarrolla las clases con todos los atributos y métodos requeridos, se realiza un adecuado encapsulamiento, se utiliza adecuadamente los constructores que permiten copiar y asignar objetos, se realiza las sobrecargas adecuadas, se realiza un uso adecuado de la memoria dinámica (`7pts`)	Desarrolla las clases con todos los atributos y métodos requeridos, se realiza un adecuado encapsulamiento, se realiza las sobrecargas adecuadas, se realiza un uso adecuado de la memoria (`5pts`)	Desarrolla las clases con todos los atributos y métodos requeridos, se realiza las sobrecargas adecuadas, errores en el funcionamiento pasa algunas pruebas (`3pts`).	Contiene errores los cuales hacen que no funcione el programa (`1pts`)

3. (7 points) **Problema 3**

Implemente una función template que verifique la existencia (`true`) de al menos un par de parámetros los cuales sumados den 10. No utilizar contenedores (`vector`, arrays, `set`, etc).

El ejercicio debe ser validado con el siguiente bloque de código:

```
int main(){
    cout << check_sum(2,5,6,7,2,1,3) << endl;
    cout << check_sum(1,2,5,6,7,2,1,1) << endl;
    cout << check_sum(1,1,2,3,6,6,2,5,5) << endl;
}
```

Para lo cual, el resultado del código debería ser el siguiente:

```
1
0
1
```

La rúbrica para esta pregunta es:

Template de Funciones	Buen nivel de abstracción, uso adecuado de los diferentes parámetros de templates, selección correcta del tipo de template, selección correcta de la especialización o recursividad de templates. (`7pts`)	Buen nivel de abstracción, uso adecuado de los diferentes parámetros de templates. Selección correcta del tipo de template, uso de especializaciones o recursividad. (`5pts`)	Buen nivel de abstracción, uso adecuado de los diferentes parámetros de templates. selección correcta del tipo de template, errores en el funcionamiento pasa algunas pruebas. (`3pts`).	No se hizo un uso adecuado de los templates. (`1pts`)