Informe de Trabajo 03

Tema: Trabajo 03

Nota

Estudiante	Escuela	Asignatura
Victor Mamani Anahua vmamanian@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación II Semestre: II Código: 20230489

Trabajo	Tema	Duración
03	Trabajo 03	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 22 Diciembre 2023	Al 23 Diciembre 2023

1. Tarea

- Crear una clase base denominada Punto que conste de las coordenadas x e y. A partir de esta clase, definir una clase denominada Circulo que tenga las coordenadas del centro y un atributo denominado radio. Entre las funciones miembro de la primera clase, deberá existir una función distancia() que devuelva la distancia entre dos puntos, donde: Distancia = ((x2 – x1) 2 + (y2 – y1) 2) 1/2
- Utilizando la clase construida en el ejercicio 01, obtener una clase derivada Cilindro derivada de Circulo. La clase Cilindro deberá tener una función miembro que calcule la superficie de dicho cilindro. La fórmula que calcula la superficie del cilindro es S = 2r(l + r) donde r es el radio del cilindro y l es la longitud.
- Caso de estudio especial: herencia múltiple. Es un tipo de herencia en la que una clase hereda el estado (estructura) y el comportamiento de más de una clase base. (hay herencia múltiple cuando una clase hereda de más de una clase). Java no permite la herencia múltiple, pero se puede conseguir la implementación de la herencia múltiple usando interfaces. Implemente el siguiente diagrama de clases UML y consiga pruebas válidas.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.v
- Visual Studio Code.

- VIM 9.0.
- OpenJDK 64-Bits 19.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos.
- Actividades del Trabajo03.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/VictorMA18/fp2-23b.git`
- URL para el Trabajo03 en el Repositorio GitHub.
- `https://github.com/VictorMA18/fp2-23b/tree/main/Fase02/Trabajo03`

4. Actividades del Trabajo03

4.1. Ejercicio1

- En el primer programa creamos la clase Punto tiene atributos x e y, un constructor para asignar coordenadas, y un método para calcular la distancia entre dos puntos. La clase Circulo hereda de Punto e incluye un radio. En el programa principal (Ejercicio1), se crean instancias de Punto y Circulo, se calcula la distancia entre dos puntos y se imprime la información del círculo. La implementación demuestra la relación de herencia entre las clases

Listing 1: Las lineas de codigos del metodo creado:

```
class Punto{
   protected double x;
   protected double y;
   public Punto(double x, double y){ //Creamos un constructor para asiganar valores a
       los atributos de la clase Punto el cual nos podra asignar puntos como para un
       centro de nuestra figura o de un punti cualquiera
      this.x = x;
      this.y = y;
   }
   public double distancia(Punto fin){ //Metodo que nos permite ver la distancia entre
       puntos o hallar la distancia de un centro de nuestra figura hacia un punto
      double distanciax = this.x - fin.x;
      double distanciay = this.y - fin.y;
      double distanciaresultante = (distanciax * distanciax) + (distanciay * distanciay);
      return Math.sqrt(distanciaresultante);
   }
   public String toString(){ //Cree este metodo para que nos puedra mostrar los puntos
       que estamos calculando
      return "(" + this.x + " , " + this.y + ")";
   }
}
class Circulo extends Punto{
```

```
    protected double radio;
    public Circulo(double x, double y, double radio){ //Creamos este constructor el cual
        nos va a establecer las coordenadas que necesitamos como su centro y asignamos
        el valor del radio de este
       super(x,y); //Llamamos al constructor de la clase Punto ,la cual seria nuestro
           centro del circulo de lo cual heredamos de la clase Punto
       this.radio = radio;
    }
    public String toString(){ //Cree este metodo para que nos puedra mostrar las
        propiedades de este como su centro y su radio
       return ("El centro del circulo es : " + "(" + this.x + " , " + this.y + ")" + " y
           su radio es : " + this.radio);
    }
}
class Ejercicio1{
    public static void main(String[] args) {
       Punto puntoinicial = new Punto(0, 0);
       Punto puntofinal = new Punto(3, 4);
       double distanciaentrepuntos = puntoinicial.distancia(puntofinal);
       System.out.println("El punto inicial: " + puntoinicial.toString() + " y el punto
           final: " + puntofinal.toString());
       System.out.println("La distancia entre estos es: " + distanciaentrepuntos);
       Circulo circulo = new Circulo(2, 1, 5);
       System.out.println(circulo.toString());
    }
}
```

```
user@victus:~/fp2-23b/Fase02/Trabajo03$ java Ejercicio1
El punto inicial: (0.0 , 0.0) y el punto final: (3.0 , 4.0)
La distancia entre estos es: 5.0
El centro del circulo es : (2.0 , 1.0) y su radio es : 5.0
```

4.2. Ejercicio2

- En el segundo programa creamos la clase Cilindro, que extiende la clase Circulo. La clase Cilindro representa un cilindro tridimensional con un centro, un radio y una altura. Su constructor llama al constructor de la clase Circulo para establecer el centro y el radio y asigna la altura adicional. La clase incluye un método para calcular la superficie del cilindro y un método toString() que devuelve una representación en cadena que incluye las propiedades del cilindro. En el programa principal (Ejercicio2), se crea una instancia de Cilindro con coordenadas, radio y altura específicos, se calcula la superficie y se imprime la información del cilindro junto con su superficie.

Listing 2: Las lineas de codigos del metodo creado:

```
class Cilindro extends Circulo {
    protected double altura;
    public Cilindro (double x , double y , double radio , double altura){ //Creamos este
        constructor el cual nos va a establecer las coordenadas que necesitamos como su
        centro , radio y la longitud de este cilindro
       super(x, y, radio); //llamamos al constructor del cual heredamos esta clase que es
           Circulo lo cual nos ayudara en nuestra clase Cilindro para tener un centro y
           un radio definido
       this.altura = altura; //asignamos la altura
    }
```

```
    public double superficie(){
        double superficiecilindro = ((2 * (this.radio)) * (this.altura + this.radio));
        return superficiecilindro;
    }
    public String toString(){ //Cree este metodo el cual me ayuda a de
        return ("El centro del cilindro es : " + "(" + this.x + " , " + this.y + ")" + " ,
            su radio es : " + this.radio + " y su altura es: " + this.altura);
    }
}
class Ejercicio2 {
    public static void main(String[] args) {
        Cilindro cilindro = new Cilindro(2, 4, 5, 10);
        double superficiecilindro = cilindro.superficie();
        System.out.println(cilindro.toString());
        System.out.println("La superficie de este cilindro es : " + superficiecilindro);
    }
}
```

```
user@victus:~/fp2-23b/Fase02/Trabajo03$ javac Ejercicio2.java
user@victus:~/fp2-23b/Fase02/Trabajo03$ java Ejercicio2
El centro del cilindro es : (2.0 , 4.0) , su radio es : 5.0 y su altura es: 10.0
La superficie de este cilindro es : 150.0
```

4.3. Ejercicio2

- En el tercer programa creamos dos interfaces, Barco y Avion, cada una con un método específico (zarpar y planear, respectivamente). Luego, se implementa la herencia múltiple a través de la clase Hidroavion, que implementa ambas interfaces. En el programa principal (Ejercicio3), se crea un objeto Hidroavion, y se llaman a sus métodos zarpar y planear, demostrando la implementación exitosa de ambas interfaces. Además, se utilizan referencias de tipo Barco y Avion para hacer referencia al mismo objeto Hidroavion

Listing 3: Las lineas de codigos del metodo creado:

```
interface Barco {
    void zarpar();
}
interface Avion{
    void planear();
}
class Hidroavion implements Barco, Avion{ //Creamos las interfaces como una herdacion
     multiple la cual Hidroavion tiene esta condicion donde este puede aplicar metodos
     de las interfaces creadas
    public void zarpar(){ //Metodo de la interfaz Barco
        System.out.println("El Hidroavion esta implementando el metodo zarpar de la
            interfaz Barco");
    }
    public void planear() {
        System.out.println("El Hidroavion esta implementando el metodo planear de la
            interfaz Avion");
    }
}
class Ejercicio3 {
    public static void main(String[] args) {
        Hidroavion hidroavion = new Hidroavion();
```

```
        hidroavion.zarpar();
        hidroavion.planear();//Comprobando la heredacion multiple por interfaces la cual
            creamos a un objeto hidroavion que usara metodos de las interfaces de la cual
            se hereda
        Barco barco = hidroavion; //La referencia de tipo Barco para hacer referencia a
            una instancia de Hidroavion.
        Avion avion = hidroavion; //La referencia de tipo Avion para hacer referencia a
            una instancia de Hidroavion.
        barco.zarpar();
        avion.planear();
    }
}
```

```
user@victus:~/fp2-23b/Fase02/Trabajo03$ javac Ejercicio3.java
user@victus:~/fp2-23b/Fase02/Trabajo03$ java Ejercicio3
El Hidroavion esta implementando el metodo zarpar de la interfaz Barco
El Hidroavion esta implementando el metodo planear de la interfaz Avion
El Hidroavion esta implementando el metodo zarpar de la interfaz Barco
El Hidroavion esta implementando el metodo planear de la interfaz Avion
```

4.4. Estructura de Trabajo03

- El contenido que se entrega en este Trabajo03 es el siguiente:

```
/Trabajo03
|----Avion.class
|----Barco.class
|----Cilindro.class
|----Circulo.class
|----Ejercicio1.class
|----Ejercicio1.java
|----Ejercicio2.class
|----Ejercicio2.java
|----Ejercicio3.class
|----Ejercicio3.java
|----Hidroavion.class
|----Latex
       |----img
              |----Captura2.png
              |----Captura3.png
              |----Captura.png
              |----logo_abet.png
              |----logo_episunsa.png
              +----logo_unsa.jpg
       |----Informe.aux
       |----Informe.fdb_latexmk
       |----Informe.fls
       |----Informe.log
       |----Informe.out
       |----Informe.pdf
       |----Informe.synctex.gz
       +----Informe.tex
+----Punto.class
```

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

5.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Tabla 3: Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el trabajo hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	4	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	2	
Total		20		18	

6. Referencias

- https://drive.google.com/drive/u/1/folders/19TzLFO-T77qG7bOWmg5OH7FXAMD2CrJL