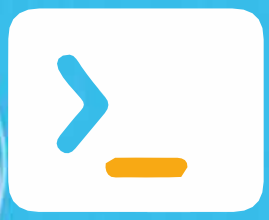

Argentina programa

Programación Orientada a Objetos con JAVA

Guía I –Parte 2

Ejercicios

EJERCICIOS DE APRENDIZAJE

Antes de comenzar con esta guía, les damos algunas recomendaciones:
Este módulo es uno de los más divertidos ya que vamos a comenzar a modelar los objetos del mundo real con el lenguaje de programación Java, esta vez, haciendo hincapié en la implementación de métodos en donde podremos aplicar las estructuras de programación vistas en PSEINT como selección e iteración. Es importante tener en cuenta que entender la programación orientada a objetos lleva tiempo y sobre todo PRÁCTICA, así que, a no desesperarse, con cada ejercicio vamos a ir entendiendo un poco más cómo aplicar este paradigma.

Te sugerimos ver los videos relacionados con este tema, antes de empezar los ejercicios, los podrás encontrar en tu aula virtual o en nuestro canal de YouTube.

Dificultad Baja Dificultad Media Dificultad Alta

1) En un nuevo proyecto crear una clase de nombre Numero, con un atributo de tipo entero, un constructor que permita inicializar dicho atributo, los métodos getter y setter y los siguientes métodos adicionales:

- **esPar:** este método retornará true si el valor guardado en el atributo es par, caso contrario retornará false.
- **esPositivo:** este método retornará true si el valor guardado en el atributo es mayor o igual a cero, caso contrario retornará false.
- **esMultiploDe:** este método recibirá un valor por parámetro y retornará true si el valor guardado en el atributo es múltiplo del valor recibido, caso contrario retornará false.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear un objeto Numero con el valor 1459
   Luego utilizando sus métodos:
b) Mostrar por consola si es par o no.
c) Mostrar por consola si es positivo o no.
d) Mostrar por consola si es múltiplo de 3.

Ver video ayuda Ejercicio 1.

2) En un nuevo proyecto crear una clase de nombre Cuadrilatero con los atributos largo y alto, un constructor que permita inicializar dichos atributos, sus respectivos getter y setter y los siguientes métodos adicionales:

- **calcularPerimetro()**: este método retornará el perímetro del cuadrilátero.
- **calcularArea()**: este método retornará la superficie del cuadrilátero.
- **esUnCuadrado()**: este método retornará true si este cuadrilátero es un cuadrado, caso contrario retornará false.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear un objeto Cuadrilatero con largo 70 y altura 50;
   Luego utilizando sus métodos:
a) Mostrar por consola su perímetro
b) Mostrar por consola su superficie.
c) Mostrar por consola si es un cuadrado o no.
d) Cambiar el estado de uno de sus atributos para que sea un cuadrado.

3) En un nuevo proyecto, crear una clase de nombre Circulo, con los atributos PI que es una constante con el valor 3.14 y radio; un constructor que permita inicializar el radio del círculo; los métodos “get” y “set” para radio y los siguientes métodos adicionales:

- **calcularArea()**: este método retornará la superficie del círculo.
- **calcularPerímetro()**: este método retornará el perímetro del círculo.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear un objeto Circulo con radio 5.75 ;
   Luego utilizando sus métodos:
b) Mostrar por consola su perímetro
c) Mostrar por consola su superficie.

4) En un nuevo proyecto, crear una clase de nombre Conversor con los siguientes métodos estáticos:

- **convertirAPesos()**: recibe un valor en dólares y retorna convertido a pesos argentinos.
- **convertirADolar()**: recibe un valor en pesos argentinos y retorna convertido a dólar.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:
Utilizando los métodos estáticos de la clase Conversor.

a) Convertir 500 dólares a pesos y mostrar por consola lo convertido.
b) Convertir 25700 pesos a dólares y mostrar por consola lo convertido.

5) En un nuevo proyecto, crear una clase de nombre Utilitario con una serie de métodos estáticos:

- **dibujaCuadrado()**: Este método recibe un número entero N, y dibujará un cuadrado de N elementos por lado utilizando el carácter "*". Por ejemplo, si el cuadrado tiene 4 elementos por lado se deberá dibujar lo siguiente:

```
* * * *
*     *
*     *
* * * *
```

- **elMayorEs():** Este método recibe 3 números enteros y mostrará por consola cual es el mayor. En caso de ser iguales lo deberá informar.
- **elDiaEs():** Se espera que este método reciba un numero entero entre 1 y 7 que corresponderá a un día de la semana, retornando el nombre que le corresponda, por ejemplo: 1 →Lunes, 2→Martes, 3→Miércoles, 4→Jueves, 5→Viernes, 6→Sábado, 7 →Domingo. Si recibe un valor distinto retornará "No existe ese día!!!".

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

Utilizando los métodos estáticos de Utilitario:

a) Dibujar un cuadrado de 5 elementos.
b) Mostrar el mayor entre (20,75 y 40)
c) Mostrar el nombre del día 5.

6) En un nuevo proyecto, crear una clase de nombre Triangulo con los atributos: lado1, lado2, lado3; un constructor que permita inicializar sus atributos; los métodos getter y setter; y los siguientes métodos adicionales:

- **esUnTriangulo()**: Sabiendo que la regla principal que da origen al triángulo tiene que ver con la longitud de sus lados. Esta plantea que la suma de dos de sus lados debe ser mayor a la longitud del tercer lado. Por lo tanto si los valores de los lados de este triángulo cumplen con la regla este método retornará true caso contrario false.
- **tipoTriangulo()**: Si es un triángulo y tiene sus tres lados iguales, este método retornará equilátero; si sus tres lados son distintos, retornará "escaleno"; caso contrario, es decir, si sólo dos de sus lados son iguales, retornará "isósceles"; en el caso de que no sea un triángulo, retornará "Con sus lados no se puede representar un triángulo!!!"

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear un objeto Triángulo válido.

Luego utilizando sus métodos:

b) Mostrar por consola que tipo de triángulo es.
c) Crear un objeto Triángulo inválido.

Luego utilizando sus métodos:

d) Mostrar por consola que tipo de triangulo es.

7) En un nuevo proyecto, crear una clase de nombre Robot con los atributos: batería inicializado en 500 unidades de energía y nombre; un constructor que permita inicializar únicamente a su atributo nombre; los métodos getter y setter para sus atributos y los siguientes métodos adicionales:

- **avanzar():** Este método recibirá la cantidad de pasos que deberá dar el robot, sabiendo que por cada 100 pasos consume 10 unidades de energía y que sólo avanzará si hay batería suficiente.
- **bateriaVacia()**: Este método retornará true, sólo cuando la batería quede con un valor menor o igual a cero.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear un objeto Robot de nombre "Tito"
   Luego utilizando sus métodos:
b) Hacerlo avanzar de a un paso hasta que se quede sin batería.

8) En un nuevo proyecto, crear una clase de nombre Calculo con los siguientes métodos estáticos:

- **esPrimo()**: Este método recibe un número entero y retorna true si el número recibido es primo, caso contrario retornará false.
- **valorAbsoluto()**: Este método recibe un número entero y retorna su valor absoluto.
- **calcularRices()**: Este método recibe los coeficientes a, b y c de una ecuación de segundo grado y mostrará por consola sus raíces y si no las tiene también lo informará.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

Utilizando los métodos estáticos de la clase Calculo:

a) Informar por consola si un número escogido por usted es primo o no.
b) Mostrar por consola el valor absoluto del valor -90;
c) Mostrar las raíces de la ecuación de segundo grado con los coeficientes: 1, 0 y 9.

9) En un nuevo proyecto, crear una clase de nombre Fecha con los siguientes atributos: dia, mes, año; un constructor que permita inicializar a todos sus atributos; los métodos getter y setter para sus atributos y los siguientes métodos adicionales:

- **esBisiesto()**: Este método retornará true si el año de esta Fecha es bisiesto; caso contrario retornará false.
- **calcularAños()**: Este método recibirá un objeto del tipo Fecha y retornará un entero que será la cantidad de años transcurridos entre esta Fecha y la que recibe por parámetro.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear dos objetos de tipo fecha.
   Utilizando sus métodos:
b) Mostrar por consola si la primera fecha y la segunda corresponden a años bisiestos.
c) Mostrar por consola, la cantidad de años transcurridos entre la primera fecha y la segunda.

10) En un nuevo proyecto, crear una clase de nombre Pensador con un atributo que almacenará un valor entero; un constructor que permita inicializar dicho atributo; los métodos get y set para dicho atributo y los siguientes métodos adicionales:

- **invertir()**: Este método retornará el valor guardado en el atributo con sus cifras invertidas. Por ejemplo si el valor guardado es 3915, retornará 5193
- **parAntes()**: Este método retornará el valor par próximo anterior al valor guardado.

- **parPosterior()**: Este método retornará el valor par próximo posterior al valor guardado.
- **primerDigito()**: Este método retornará el primer dígito del valor guardado.
- **ultimoDigito()**: Este método retornará el último dígito del valor guardado.

Luego desde la clase principal del proyecto (la que contiene el método main) se pide:

a) Crear una instancia de la clase Pensador.
b) Probar todos sus métodos y mostrar por consola los resultados obtenidos.

BIBLIOGRAFIA:

Christopher Alexander, “A Pattern Language”, 1978

Erich Gamma, “Design Patterns: Elements of Reusable OO Software”

Martin Fowler, “Analysis Patterns: Reusable Object Models”, Addison Wesley,

1997