Práctica de laboratorio # 6

MÉTODOS Y CLASES

En este laboratorio se pretende que el o la estudiante comprenda el uso y la importancia de la programación por objetos, especialmente en cuanto a la generalización de problemas mediante el uso de métodos y clases.

Una magnitud física es un valor asociado a una propiedad física o cualidad medible de un sistema físico, es decir, a la que se le pueden asignar distintos valores como resultado de una medición o una relación de medidas.[1] Para ello se utilizan las unidades de medidas.

De forma individual, realice un programa que sirva de convertidor de unidades, de modo que fácilmente se pueda usar para transformar una magnitud física expresada en una unidad, a otra unidad.

Ejercicio A. Creación de las clases vacías

1. Cree una clase con el nombre Convertidor. Esta clase va a funcionar como una calculadora inteligente, de modo que va a llevar implementado todos los cálculos necesarios para convertir magnitudes entre diferentes unidades.

2. Cree una clase con el nombre Pruebas. Esta clase va a contener el método main del programa, y además va a manejar la interacción con el usuario y las invocaciones principales a los métodos.

ⓘ Antes de continuar, un poco de información de repaso…

Recordemos lo que necesitamos conocer de un método para poder implementarlo:

(a) Alcance. Si es público o privado.

(b) Tipo de retorno. Si el método nos genera un resultado, debemos "responder" un valor. Se debe especificar en la declaración cuál es el tipo de dato que se va a retornar. El tipo es `void` si no retorna un valor.

(c) Nombre. La idea es que sea significativo de modo que pueda recordarlo fácilmente y que en serio represente lo que soluciona el método

(d) Parámetros. Son las entradas del algoritmo, los valores que se necesita conocer para poder hacer la operación. Van escritos entre paréntesis y separados por comas. Cada uno debe especificar el tipo de dato y ponerle un nombre para identificarlo en el cuerpo del método. Si no hay, se colocan paréntesis vacíos.

(e) Cuerpo del método. En un bloque de código, se programa el algoritmo. No debemos olvidar la instrucción `return` en caso de que haya un retorno. La idea es utilizar acá los parámetros en caso de que haya.

[1] Tomado de Wikipedia

Ejercicio B. Implementación de un método en la clase Convertidor

3. En Convertidor, cree un nuevo método que transforma una magnitud expresada en kilómetros a la misma expresada en metros. Recordemos para la declaración de un método lo que necesito saber:
- → Alcance: en este caso es público para que pueda ser llamado desde la clase Pruebas.
- → Retorno: el resultado del método es una magnitud en metros, por lo que es un número real (por ejemplo, tipo double)
- → Nombre: vigile que sea un nombre significativo. Para este caso un buen nombre podría ser convertir_km_m
- → Parámetros: la entrada del método en este caso sería una magnitud en kilómetros, por lo que también es un número real (por ejemplo, tipo double)

El cuerpo del método debe implementar el algoritmo para transforme el valor que recibe de parámetro en kilómetros, a metros.

4. Si se rinde, utilice el siguiente código, observando cuidadosamente cómo lo hace. Si usted no se rindió, use este ejemplo para comparar lo que usted implementó:

```
public double convertir_km_m (double km) {
      double m = 0.0;
      m = km * 1000;
      return m;
}
```

5. Haga una prueba del método. Para ello, vaya a la clase Pruebas, y dentro del main programe lo siguiente:
- **5.1** Declare e inicialice una variable de tipo double
- **5.2** Pida al usuario un número (puede tener decimales) que sea una medida en km.
- **5.3** Haga las conversiones necesarias para poder ser tratado como un número
- **5.4** Haga un llamado (o invocación) al método que convierte kilómetros a metros, en la instancia creada, y el resultado métalo en la variable creada en el punto 5.1.Recuerde que la invocación a un método se hace utilizando el nombre de la instancia, seguido de un punto, y el nombre del método con los parámetros. Por ejemplo, `p.ladrar()`
- **5.5** Imprima esa variable

Tome de referencia este ejemplo que convierte 2.57 km a metros.

```
double ejemplo = instancia.convertir_km_m (2.57);
JOptionPane.showMessageDialog(null, ejemplo);
```

6. Compile y corra el programa. Ingrese un valor con decimales en kilómetros y espere respuesta del programa. ¿Qué valor debería imprimirse? ¿Lo está haciendo bien? Corrija los errores necesarios antes de continuar.

7. Verifique que entiende realmente lo que está pasando en el programa. Haga al profesor las preguntas necesarias para aclarar conceptos. Por ejemplo: Clase y método, declaración del método y uso de parámetros, creación de una instancia e invocación a un método.

Ejercicio C. Implementación de varios métodos más

8. En Convertidor, de forma similar a como se hizo para pasar de kilómetros a metros, programe los siguientes métodos que retornan valores convertidos y reciben parámetros por convertir. Para todos ellos asuma números reales:

a) Pasar de metros a kilómetros

b) Pasar de cm a pulgadas

c) Pasar de pulgadas a cm

d) Pasar de °F a °K

e) Pasar de °K a °F

f) Pasar de minutos a días

g) Pasar de días a minutos

h) Pasar libras a gramos

i) Pasar de gramos a libras

$$1 \quad pul = 2.54\ cm$$

$$F = \frac{9(K - 273.15)}{5} + 32$$

$$1\ d = 24\ h$$

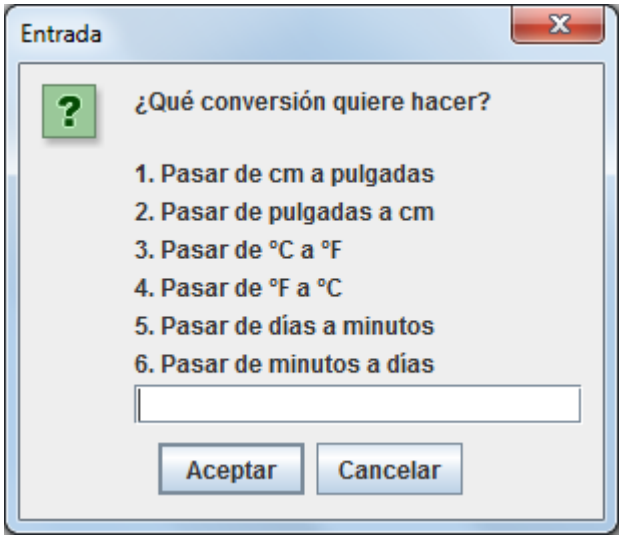
$$1 \quad h = 60\ min$$

$$1 \quad lib = 453.59\ g$$

9. Verifique que todo funcione. Puede hacer pruebas usando llamados desde el main. Recuerde que no necesita crear nuevas instancias de Convertidor cada vez que va a llamar un método. Más bien, puede usar la misma instancia.

Ejercicio D. Invocación de los métodos creados

10. En la clase Pruebas, cree un método sin parámetros llamado `preguntar` que muestre en una ventana un menú, donde pregunte por la conversión que quiere hacer, igual a la que se enseña a la derecha.[2] Use JOptionPane. Tome en cuenta que la entrada que digite el usuario debe guardarla e interpretarla.



11. Tome en cuenta que el método preguntar necesitaría crear una nueva instancia de Convertidor, para poder desde ahí hacer los llamados a los métodos.[3]

12. Dependiendo del número que ingresó, haga un llamado a la función[4] que corresponde, y que programó en el punto 8.

13. Agregue una opción al menú para salir, en el método `preguntar`. Modifique lo que sea necesario, para que el menú se muestre continuamente después de hacer una conversión. Hasta que se seleccione la opción de salir, el programa termina.

[2] Note que es un ejemplo de la forma de cómo debe verse, no sigue exactamente las conversiones que se pidieron en el laboratorio

[3] Otra opción podría ser crear métodos estáticos y hacer llamados desde la clase directamente, pero use esta opción si y solo si tiene 100% claro en qué consisten y cómo se usan los métodos estáticos.

[4] La palabra "función" se refiere a un método que retorna un valor.

14. En `Pruebas`, puede comentar o borrar el contenido del main que tenía, ponga dentro del main **solamente** lo siguiente:

14.1 Crear una instancia de Pruebas e inicializarla.

14.2 Con esa instancia, invocar el método `preguntar`

15. Hasta este punto, si corre el programa, debería mostrarse el menú, y al seleccionar una opción se convierte y muestra un valor. Verifique que eso ocurra y haga los cambios necesarios si no funciona así.

Ejercicio E. Manejo de excepciones.

16. Piense ¿Qué pasa si en alguna de las pruebas inserta en ejecución a la ventana de JOptionPane una letra? Ocurre un error de ejecución, una excepción. Si gusta, haga la prueba.

Utilice la estructura try – catch **en el método preguntar**, para evitar que el programa se caiga. Si ocurre una excepción, el programa debe notificar al usuario que ha ocurrido un error con el número insertado y volver al menú de conversiones.

Ejercicio F (Opcional). Contador de conversiones realizadas.

17. Haga que la clase Convertidor sea capaz de guardar y llevar conteo de cuántas operaciones se han realizado. Para ello deberá:

- Crear un atributo (privado) encargado de contar las operaciones de dicho convertidor.
- Crear el constructor de la clase que establece el valor inicial para el atributo, es decir, cero.
- Cerciorarse de que el conteo de operaciones se lleva correctamente (es decir, que se aumente el valor cada vez que se resuelve una operación).
- Crear un método en el Convertidor que retorna la cantidad de conversiones que están guardadas
- Modifique el método `preguntar` para que cuando el usuario decide salir, le muestre una ventana con la cantidad de operaciones que realizó antes de terminar el programa.

18. Entregue todas las clases para resolver el laboratorio antes de la fecha y hora descrita en la plataforma virtual. Suba un solo archivo, que sea una carpeta comprimida en .zip o .rar que dentro contiene los archivos .java que usó. (Solo se requieren los .java, no es necesario que incluya los .java~, ni los .class). El título de esta carpeta comprimida debe corresponder al número de carné, guión bajo, "Lab6". Por ejemplo, si el carné es *A12345*, el archivo debe llamarse "*A12345_Lab6*".

Rúbrica	
Definición de clases	5%
Implementación de métodos del convertidor	50%
Funcionamiento del menú	30%
Manejo de excepciones	10%
Uso y declaración del main	5%
Opcional: contador	5%
Total	**105%**

Sin embargo, la calificación máxima que se puede obtener es un 100.