PRÁCTICO 0 –Java

1. Defina un clase de nombre Conversion que tenga:

    - Un método conversionAFahrenheitB que reciba como argumento un valor de temperatura en grados Celsius y devuelva como resultado su valor en grados Farenheit.

      Agregue dentro de la clase un método "main" el cual cree una instancia de la clase Conversión y realice pruebas de su correcto funcionamiento.

    Recuerde la fórmula de conversión: F = 9/5 * C + 32.

2. Genere una clase Solicitud que solicite números en entrada estándar cuando el usuario ingrese un -1 se terminará el mismo, deplegando la siguiente información:
    - mayor número introducido
    - menor número introducido
    - suma de todos los números
    - suma de los números positivos
    - suma de los números negativos
    - promedio de los números

    Para la solicitud de datos se deberá utilizar la clase Scanner, para tener de referencia revise los siguientes links:
    - Ejemplos de uso
        i. https://www.w3schools.com/java/java_user_input.asp
    - Javadocs
        i. https://docs.oracle.com/javase/8/docs/api/java/util/Scanner.html

3. Escriba una clase NumeroPrimo que tenga un método con la siguiente firma:
    - boolean esNumeroPrimo(int numero)
        i. Esta operación debe retornar true si el número ingresado es primo, falso en caso contrario.

    A su vez genere un método main que genere una instancia de NumeroPrimo y pida un número por entrada estandar y muestre en pantalla si es número primo o no.
    Recordar que un número primo es aquel que solo puede dividirse entre 1 y si mismo. Por ejemplo: 25 no es primo, ya que 25 es divisible entre 5, sin embargo, 17 si es primo.

4. Se debe generar una clase Matrices que tenga las siguientes operaciones:
    - public void desplegar(int[][] matriz);
        i. Despliega en salida estándar el contenido de la matriz. Elija el formato, respetando de colocar una linea por cada fila.
    - public boolean esCuadrada(int[][] matriz);
        i. Retorna true si la matriz es cuadrada, false en caso contrario.
    - public int filas(int[][] matriz);
        i. Retorna la cantidad de filas de la matriz
    - public int columnas(matriz);
        i. Retorna la cantidad de columnas de la matriz
    - public int[][] sumar(int[][] matrizA, int[][] matrizB);
        i. Retorna la matriz resultado de sumar matrizA + matrizB. En caso que las matrices tengan distintas dimensiones se debe retornar null.

5. Primer Parcial 2018.
"El Sistema Nacional Integrado de Salud reglamenta el derecho a la protección de salud de todos los habitantes del país, a través de un Seguro Nacional de Salud (SNS), financiado por el Fondo Nacional de Salud (Fonasa), extendiendo obligatoriamente la cobertura médica de los trabajadores a su núcleo familiar" extracto obtenido del sitio del Banco de Previsión Social.

Para ello cada trabajador realiza aportes personales obligatorios, que varían en función de la remuneración (sueldo) y la situación familiar del trabajador. Estos aportes se calculan como un % variable sobre las remuneraciones y se descuentan del sueldo (retenciones).

Estos aportes se realizan independiente del sueldo de la persona, ahora luego de culminar el año se verifica si lo aportado supera cierto tope mensual establecido. De ser así al trabajador le corresponde una devolución Fonasa que podrá cobrar.

Se desea realizar un simulador que le permita a un trabajador saber cuánto le corresponde de devolución Fonasa para cierto año. Con lo cual el trabajador deberá ingresar:

- 12 remuneraciones mensuales sin descuento correspondientes al año que desea consultar.
- Si tiene cónyuge a cargo (true o falso)
- Cantidad de hijos a cargo (numero entero mayor o igual a cero)

En base a esto se tienen que realizar 2 cálculos, el primero indica cuales son los aportes personales realizados mensuales y el segundo cuanto es el tope mensual que le corresponde. El cálculo de la devolución Fonasa surge como:

- Suma de aportes realizados mensuales – Tope mensual * 12
    - Si la devolución es menor o igual a cero, la devolución es cero.

Para calcular los aportes personales, se debe tomar la remuneración mensual y calcular el % de descuento correspondiente. Con lo que el aporte seria remuneración * X %, teniendo en cuenta que el valor del % se determina siguiendo las siguientes reglas:

Si la remuneración mensual es menor o igual a $ 2552,5 entonces:
Sin cónyuge a cargo (con o sin hijos) se descuenta un 3%
Con cónyuge a cargo (con o sin hijos) se descuenta un 5%
Si la remuneración mensual es mayor a $ 2552,5 entonces
Sin cónyuge a cargo entonces
Sin hijos se descuenta 4,5%
Con hijos se descuenta 6 %
Con cónyuge a carga entonces
Sin hijos se descuenta 6,5%
Con hijos se descuenta 8%

Para calcular el tope mensual se considera un valor fijo por persona a cargo. Este valor fijo se le llama CPE (Costo promedio equivalente) y tiene un valor de $ 3246,25. Por ejemplo si la persona no tiene cónyuge ni hijos a cargo, su tope mensual de aportes es 1 x $3246,25. Ahora si tiene cónyuge a cargo y no tiene hijos a cargo su tope mensual de aporte es 2 x $3246,25 y por último si tiene cónyuge a cargo y 3 hijos a cargo su tope mensual de aporte es de 5 x $ 3246,25.

Se debe crear una clase P2Ejercicio2 que contenga los siguientes métodos:

- public static double calcularDevolucion(double[] remuneraciones, boolean conyugeACargo, int hijosACargo)
    - El argumento remuneraciones es un vector de tamaño 12 con las remuneraciones mensuales.
    - El argumento conyugeACargo indica si se tiene un cónyuge a cargo o no.

  - El argumento hijosACargo indica la cantidad de hijos a cargo.
  - Devuelve el importe de devolución Fonasa total que le corresponde a la persona.
- public static void main(String[] args)
  - Se debe realizar pruebas de la función calcularDevolución demostrando su correcto funcionamiento en distintos escenarios. Colocar al menos 5 escenarios.

Algunos ejemplos serian:

1. **Remuneración mensual 110.000 sin cónyuge a cargo y con 1 hijo a cargo.**

```
double[] remuneraciones = {110000, 110000, 110000, 110000, 110000, 110000,
110000, 110000, 110000, 110000, 110000, 110000};
boolean conyugeACargo = false;
int hijosACargo = 1;

double devolucion = calcularDevolucion(remuneraciones, conyugeACargo,
hijosACargo);

System.out.println("Devolución es " + devolucion);
```

→ La salida es : Devolución es 1290.0

2. **Remuneración mensual 110.000 con cónyuge a cargo y con 1 hijo a cargo**

```
double[] remuneraciones = {110000, 110000, 110000, 110000, 110000, 110000,
110000, 110000, 110000, 110000, 110000, 110000};
boolean conyugeACargo = true;
int hijosACargo = 1;

double devolucion = calcularDevolucion(remuneraciones, conyugeACargo,
hijosACargo);

System.out.println("Devolución es " + devolucion);
```

→ La salida es : Devolución es 0.0

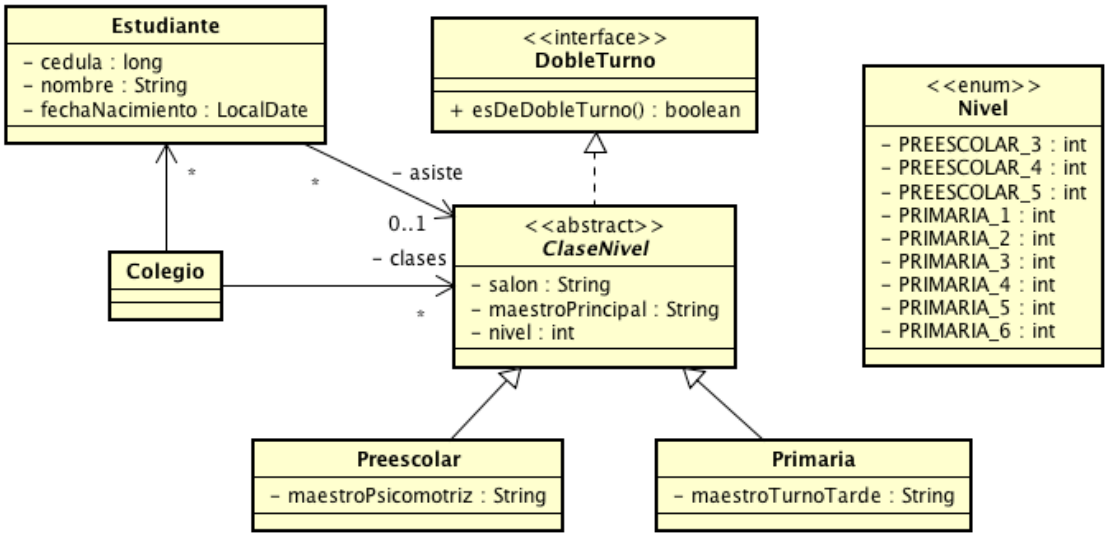
3. **Remuneración mensual 150.000 con cónyuge sin hijos a cargo**

```
double[] remuneraciones = {150000, 150000, 150000, 150000, 150000, 150000,
150000, 150000, 150000, 150000, 150000, 150000};
boolean conyugeACargo = true;
int hijosACargo = 0;

double devolucion = calcularDevolucion(remuneraciones, conyugeACargo,
hijosACargo);

System.out.println("Devolución es " + devolucion);
```

→ La salida es : Devolución es 39090.0

6. Segundo Parcial 2018

Una institución educativa de nivel preescolar y primaria desea informatizar sus registros de estudiantes. Para ello se desprende el siguiente modelo conceptual en UML que representa la realidad:



En resumen el colegio tiene un conjunto de estudiantes y un conjunto de clases. El colegio solo admite tener una clase por nivel. Cada estudiante pertenece a un nivel (preescolar y primaria). Para el caso de preescolar se tiene solo el turno de la mañana con un maestro principal, un maestro de psicomotricidad y un salón asignado. Para el caso de primaria se tiene un salón, un maestro principal en el turno de la mañana y un maestro para el turno de la tarde (en este colegio todo estudiante de primaria asiste doble turno).

Teniendo esto en cuenta se desea implementar las siguientes operaciones en la clase Colegio:

- void crearClasePreescolar(String salon, String maestroPrincipal, int nivel, String maestroPsicomotriz)
  - Esta operación crea una clase en el nivel preescolar con los datos indicados y lo asocia al colegio.
- void crearClasePrimaria(String salon, String maestroPrincipal, int nivel, String maestroTurnoTarde)
  - Esta operación crea una clase en el nivel primaria con los datos indicados y lo asocia al colegio.
- void crearEstudiante(long cedula, String nombre, LocalDate fechaNacimiento) throws EstudianteYaExiste
  - Esta operación crea un estudiante (sin asociarlo a ninguna clase)
  - Lanza la excepción check EstudianteYaExiste si ya existe un estudiante registrado con los datos pasados.
- void asociarEstudianteClase(long cedula, boolean esPrimaria, int nivel) throws EstudianteNoExiste
  - Esta operación asocia un estudiante existente a una clase. En el caso que el estudiante ya disponga de una clase asignada no debe realizar nada.
  - Lanza la excepción check EstudianteNoExiste si el estudiante que se quiere asociar no existe.
- int obtenerCantidadEstudiantesPrimaria()
  - Retorna la cantidad de estudiantes registrados en primaria.

Consideraciones:

- Notar que la clase ClaseNivel es abstracta.
- Las clases Preescolar y Primaria no deben permitir tener clases hijas.
- Hay una interfaz definida de nombre DobleTurno, la cual debe implementarse en primaria y preescolar, tomando como referencia que solo las clases de primaria tienen doble turno.

- Se definió un enumerado para el nivel de una clase, si lo desea puede optar por definir constantes.
- Todas las clases sin ser las excepciones se deben colocar en el paquete prog1.entities.
- Las clases EstudianteYaExiste y la clase EstudianteNoExiste se deben colocar dentro del paquete prog1.exceptions.
- La clase Main debe incluirse dentro del paquete prog1.
- Dentro de la clase Main se debe crear un método estático main que pruebe el correcto funcionamiento del sistema. Para ello se recomienda que se cree una instancia de colegio, registre un nivel de preescolar y un nivel de primaria, registre 3 estudiantes de primaria y 2 de preescolar y prueba que la función obtenerCantidadEstudiantesPrimaria retorna el valor esperado (3).