|  |
| --- |
| EXAMEN FINAL  **CONVOCATORIA EXTRAORDINARIA**  **FP A DISTANCIA**  CFGS: Desarrollo de Aplicaciones Web  Módulo: Programación  Curso: 1º  Docente: Borja Martín Herrera |

Alumno/a: FRANCISCO DOMINGUEZ PEZO

|  |  |
| --- | --- |
| Preguntas elegidas: | 1,3. |

**Preguntas abiertas (elija 2 de las siguientes preguntas).**

1. Supongamos que vamos a automatizar la forma de rellenar un formulario. Para ello vamos a crear una clase Formulario que tendrá un método main.

Dentro de este main se le mostrará al usuario un menú con las siguientes opciones:

1. Rellenar nombre.

2. Rellenar apellidos.

3. Rellenar DNI.

4. Rellenar e-mail.

5. Finalizar.

Mientras que el usuario no elija la opción 5, se le seguirá mostrando el menú. Si el usuario elije la opción 5, solo se le dejará salir del programa en caso de que haya rellenado todos los datos, en caso contrario se mostrará un mensaje por pantalla indicando que faltan datos por rellenar. (1,5 puntos)

En cada opción se realizarán las siguientes comprobaciones:

- Opción 1: se introducen únicamente letras. En caso contrario lanzaremos la excepción TipoDatoIncorrectoException. (0,18 puntos)

- Opción 2: se introducen únicamente letras. En caso contrario lanzaremos la excepción TipoDatoIncorrectoException. (0,18 puntos)

- Opción 3:

- Si la longitud es distinta a 9 dígitos lanzaremos LongitudDNINoValidaException. (0,36 puntos)

- Si el último caracter es distinto de una letra lanzaremos UltimoDigitoNoLetraException. (0,36 puntos)

- Si los primeros 8 dígitos no son numéricos lanzaremos NumeracionContieneLetrasException. (0,36 puntos)

- Opción 4: comprobaremos que se ha introducido un caracter "@" y un "." En caso contrario lanzaremos la excepción EmailIncorrectoException. (0,36 puntos)

Al salir del programa se mostrarán por pantalla un resumen con los datos rellenados. (0,2 puntos)

*Recordar los métodos Character.isDigit() y Character.isLetter() que comprueban si un caracter que se le pasa por parámetro es un dígito o una letra, respectivamente.*

*Recordar el método .contains() que tienen los objetos tipo String que indica si contiene la cadena de caracteres que se le pasa por parámetro.*

1. Supongamos que se desea implementar una aplicación para calcular el perímetro de diferentes figuras geométricas. Las figuras que se deben soportar son: círculos, rectángulos y triángulos. Para ello, se definirá una clase Figura2D de la que heredarán las figuras anteriores y una interfaz FiguraGeometrica con un método para calcular el perímetro que implementará Figura2D.

**Vamos paso por paso:**

1) Crea las clases Círculo, Rectángulo y Triángulo. Circulo tendrá como atributos un nombre y el radio, tendrá un constructor que recibe sus atributos por parámetro y los getters y setters de los atributos. Rectángulo tendrá como atributos un nombre, la longitud de la base y la longitud de la altura, un constructor que recibe sus atributos por parámetro y los getters y setters de los atributos. Triángulo tendrá como atributos un nombre y la longitud de los tres lados, un constructor que recibe sus atributos por parámetro y los getters y setters de los atributos.

Recuerda que el perímetro del cículo es 2 \* Math.PI \* radio. El de un triángulo y el de un rectángulo es la suma de la longitud de sus lados.

Además, se definirán en cada clase los métodos toString que imprimirá por pantalla todos los atributos del objeto y el método equals que indicará que un objeto es igual a otro únicamente si son iguales todos sus atributos (1,5 puntos)

2) Nos hemos dado cuenta de que las 3 clases comparten un atributo en común así que crearemos una clase Figura2D que tendrá como atributo el nombre de la figura, un constructor que recibe este atributo como parámetro y los getters y setters de dicho atributo. Modificaremos las clases creadas en el punto 1 para que hereden de Figura2D, recuerda que habrá que modificar algunos métodos para no repetir código. (0,5 puntos)

3) Crea la interfaz FiguraGeometrica con el método "double calcularPerimetro()". (0,25 puntos)

4) Modifica Figura2D para que sea una clase abstracta que implemente FiguraGeometrica, deja el método calcularPerimetro() como abstracto. (0,25 puntos)

5) Modifica Círculo, Rectángulo y Triángulo para que implementen el método calcularPerimetro(). (0,25 puntos)

6) Crea una clase Prueba con una función main en la que se declara un array unidimensional o un ArrayList de tipo Figura2D, que contenga círculos, rectángulos y triángulos. Recorre la estructura y muestra por pantalla para cada figura su nombre y su perímetro. (0,1 puntos con array multidimensional; 0,25 puntos con ArrayList)

1. Supongamos que queremos implementar una clase "Estudiante" en Java que tenga como atributos el nombre, la edad y tres notas.

Crea la clase Estudiante con los atributos nombrados, un constructor que reciba por parámetro todos los atributos y los inicialice al valor dado, los getters de todos los atributos y un método que calcule la nota media del estudiante. Define también el método toString que imprimirá por pantalla todos los atributos del objeto y el método equals que indicará que un objeto tipo Estudiante es igual a otro únicamente si son iguales todos sus atributos. (1,5 puntos)

En la misma clase crea un método main en el que se inicialicen 2 estudiantes con los siguientes datos:

*Nombre: Juan Edad: 27 Nota1: 3.2 Nota2: 5.6 Nota3: 8.3*

*Nombre: Ana Edad: 35 Nota1: 7.4 Nota2: 6.5 Nota3: 9*

Estos dos estudiantes se añadirán a un array unidimensional o un ArrayList. Además, se mostrará por pantalla el nombre, edad y nota media de cada estudiante. (0,25 puntos si se usa array unidimensional; 0,5 puntos si se usa ArrayList).

Queremos leer los datos del fichero de texto adjunto "estudiantes.txt" en el que cada línea contiene los datos de un estudiante separados por comas: el nombre, la edad y las tres notas.

Para ello, implementaremos un método estático dentro de la clase Estudiante que reciba el nombre del fichero como parámetro y devuelva un array unidimensional (en este caso de dimensión 2, el número de líneas que tiene el fichero "estudiantes.txt") o un ArrayList de objetos de la clase Estudiante. Si el fichero no existe, se lanzará una excepción de tipo IOException. (1 punto si se usa array unidimensional; 1,25 puntos si se usa ArrayList).

La firma de dicho método sería la siguiente:

*public static ArrayList<Estudiante> leerEstudiantes(String nombreFichero) throws IOException*

*public static Estudiante[] leerEstudiantes(String nombreFichero) throws IOException*

Por último, modificaremos el main anterior añadiendo el uso del método para obtener la lista de estudiantes y la mostraremos por pantalla. (0,25 puntos)

*Recordar que para leer ficheros de datos podemos utilizar: BufferedReader br = new BufferedReader(new FileReader("estudiantes.txt"));*

Para subir los ejercicios del examen, realizalo en un fichero llamado como nombre\_examen.zip donde estén todos los proyectos que incluyen el examen