|  |  |  |
| --- | --- | --- |
| **Question 1** |  |  |
| Convert the following for-loop to a while-loop.   |  | | --- | | for(int i = 0; i < 10; i++){  System.out.println(i + 1);  } | | | |
|  | | |
| **Question 2** |  |  |
| Create an application that allows students to choose for Belgium Campus.   1. The first step is to output the following line in the command prompt:   *Do you want to study at Belgium Campus? (y/n)* | |  |
| 1. Then you need to allow the user to enter a char into the command prompt (“y” or “n”). | |  |
| 1. Now you have to check wether the user inputed “y” or “n”. If the user entered “y” you go to step 3a, if he entered “n” you have to go to step 3b. | |  |
| * 1. Display the following line into the command prompt and end the application:   *That’s a smart choice! See you next year.* | |  |
| * 1. Now you should have entered a while loop. | |  |
| 1. Now you have to display:   *Are you sure? Because we think you are going to miss out.(y/n)* | |  |
| 1. Then you need to allow the user to enter a String into the command prompt (“y” or “n”). | |  |
| 1. You have to check wether the user entered “y” or “n”. If the user entered “y” he has to go to 6a, if he entered “n” you have to go to 6b. | |  |
| * 1. The user entered “y” so you have to display :   *Wrong choice!* | |  |
| * 1. The user entered “n” so you have to display :   *We thought so! You can make up your mind if you wish to.* | |  |
| 1. After this you have to go back to step 1. Don’t forget you have to stay in the same while loop. | |  |
| **Question 3** |  |  |
| You have to make a menu that allows the user to choice wether to start listen music or watch movie. The user’s choice has to be validated using a switch.[Use enum]   1. Listen music 2. watch movie 3. Enter number | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Question 4** | | |
| Write a program called **GradesAverage**, which prompts a user for the number of students, reads it from the keyboard, and saves it in an **int** variable called **numStudents**. The application should then prompt the user for the grades of each of the students and save them in an **int** array named **grades**. The program should validate to ensure that the entered grade is between 0 and 100. A sample session is as follows:   |  | | --- | | Enter the number of students: 3  Enter the grade for student 1: 55  Enter the grade for student 2: 108  Invalid grade, try again...  Enter the grade for student 2: 56  Enter the grade for student 3: 57  The average is 56.0 | | | |
|  | | |
| **Question 5** |  |  |
| Write a superclass called Shape (as shown in the class diagram), which contains: | |  |
| * Two instance variables color (String) and filled (boolean). | |  |
| * Two constructors: a no-arg (no-argument) or default constructor that initializes the color to "green" and filledto true, and a constructor that initializes the color and filled to the given values. | |  |
| * Getter and setter for all the instance variables. | |  |
| * A toString() method that returns "A Shape with color of xxx and filled/Not filled". | |  |
| Write a test program to test all the methods defined in Shape. | |  |
|  | |  |
| Write two subclasses of Shape called Circle and Rectangle, as shown in the class diagram. | |  |
| The Circle class contains: | |  |
| * An instance variable radius (double). | |  |
| * Three constructors as shown. The no-arg constructor initializes the radius to 1.0. | |  |
| * Getter and setter for the instance variable radius. | |  |
| * Methods getArea() and getPerimeter(). Here both can just return 0. | |  |
| * Override the toString() method inherited, to return "A Circle with radius=xxx, which is a subclass of yyy", where yyy is the output of the toString() method from the superclass. | |  |
| The Rectangle class contains: | |  |
| * Two instance variables width (double) and length (double). | |  |
| * Three constructors as shown. The no-arg constructor initializes the width and length to 1.0. | |  |
| * Getter and setter for all the instance variables. | |  |
| * Methods getArea() and getPerimeter(). The method getArea has to return the width multiplied with the length. The getPerimeter() method should return the width added with the length. | |  |
| * Override the toString() method inherited, to return "A Rectangle with width=xxx and length=zzz, which is a subclass of yyy", where yyy is the output of the toString()method from the superclass. | |  |
|  | |  |
| ExerciseOOP_ShapeAndSubclasses.png | |  |
| **Question 6** | | |
| Write a Java console program that will prompt a user to enter any year then tests if it is a leap year or not. Your program must contain a method called **LeapYearTest** which is called from the main class. | | |
| **Question 7** | | |
| What will be the output/ error of the following code snippets below, motivate your answers:   |  | | --- | | enum Enums  {      ONE, TWO, THREE, ONE, FOUR;  } |  |  | | --- | | enum Levels  {      private TOP,        public MEDIUM,        protected BOTTOM;  } |  |  | | --- | | enum MyEnums  {      FIRST, SECOND, THIRD, FOURTH;  }    public class Test  {     public static void main(String[] args)     {         MyEnums[] myEnums = new MyEnums[4];           for (int i = 0; i < myEnums.length; i++)         {             System.out.println(myEnums[i]);         }     }  } |  |  | | --- | | interface One  {      String s = "ONE";        void printS();  }    interface Two  {      String s = "TWO";        void printS();  }    enum Three implements One, Two  {      ONE      {          String s = "ONE";      },        TWO      {          String s = "TWO";      };        String s = "THREE";        public void printS()      {          System.out.println(s);      }  }    public class MainClass  {      public static void main(String[] args)      {          Three.ONE.printS();            Three.TWO.printS();      }  } |   **Does below code compile successfully? If not, why?**   |  | | --- | | class A  {      class B      {          enum C          {              D, E, F;          }      }  } |  |  | | --- | | enum MyEnum  {      ABC, XYZ, PQR  }    class MyClass extends MyEnum  {      int i;  } |  |  | | --- | | class A  {      void methodA()      {          enum Enums          {            }      }  } | | | |