School of Science, Computing and Artificial Intelligence
The University of the West Indies, Five Islands

COMP2232 - Lab 2 - 2%

Item	Marks
section 1	10 marks
section 2	30 marks
section 2	20 marks

Section 1: Introduction

With code examples, explain what is meant by:

1. SOLID principles
   (Please be aware that we have not yet delved into the topics of interfaces and abstract classes.
   Some principles will incorporate these concepts, the purpose here is to highlight their relevance in the design and implementation of classes. We will cover these topics in the coming weeks.)

Section 2: Encapsulation, Classes and Objects

1. Write the code for an AirConditioner class with the following attributes:
   - brand (String) - default is null
   - location (String) - default is null

- acID (integer) - default is 0
- isOn (boolean) - default is false (AC is off) units (String) - default is Celsius
- mode (String) - default is Cool roomTemperature (double) - default is 24C desiredTemperature (double) - default is 18C

Ensure that information hiding principles are not violated!

2. Write accessor methods for the following attributes:
   - brand (String)
   - location (String)
   - acID (integer)
   - isOn (boolean)
   - units (String)
   - mode (String)
   - roomTemperature (double)
   - desiredTemperature (double)
3. Write mutator methods for the following attributes:
   - units (String) - value is either Fahrenheit or Celsius (default)
   - mode (int) - input value is either 1, 2 or 3 which corresponds to Fan, Dry or Cool respectively. (Hint: research switch statements in Java for a quick implementation)

   For all methods, print error messages if the input does not conform to the ranges specified.
4. Write the following methods to control the AirCondition object which return true if successful and false otherwise.
   - turnOff ( ): Boolean
   - turnOn( ): Boolean
5. Write a toString( ) method that prints the aggregated state of an AirConditioner object.
6. Write a main class called CoolingSystem with the following structure:

```
public class CoolingSystem{
  public static void main(String[ ] args){
       //code for 7 and onward goes here
    }
}
```

Brand	Location	acID	Turned on
Carrier	Atrium	34	No
Lennox	Lobby	93	No
Trane	Kitchen	67	Yes

7. In the CoolingSystem class:
   - Create three AirConditioner objects with the state specified in the table above.
   - Create an AirConditioner array and add the AirConditioner objects to the array.
   - Iterate through the array and print the state of each AirConditioner object in the array by invoking the toString( ) method on each object.
8. Write a private method with the following method signature: private double convertCelsiusToFahrenheit (double celciusValue) This method should convert a Celsius value to Fahrenheit.
9. Write a private method with the following method signature: private double convertFahrenheitToCelcius(double fahrenheitValue) This method should convert a Fahrenheit value to Celsius.
10. Write a mutator method for the following attribute in the AirConditioner class:
    desiredTemperature (double) - allowable range is 15 to 30 C otherwise the default is used.
    This method should cater for the user entering a value that is in either Celsius or Fahrenheit and doing the appropriate conversion.
11. Change the details of the AirConditioner objects as follows by invoking the appropriate methods of the class:

acID	Turned on	Units	Mode	Desired Temperature
34	Yes	C	Cool	19.5
93	Yes	F	Fan	16.3
67	No	C	Dry	24.0

Section 3: Class Relationships

1. Using the Die class given below, write a class called PairOfDice, composed of two Die objects. Include methods to set and get the individual die values, a method to roll the dice, and a method that returns the current sum of the two die values. Create a driver class called RollingDiceTwo to instantiate and use a PairOfDice object.

2. Use the Account class which simulate an ATM machine. Create 10 accounts in an array with id 0, 1, . . . , 9, and an initial balance of $100. The system prompts the user to enter an id. If the id is entered incorrectly, ask the user to enter a correct id. Once an id is accepted, the main menu is displayed as shown in the sample run. You can enter choice 1 for viewing the current balance, 2 for withdrawing money, 3 for depositing money, and 4 for exiting the main menu. Once you exit, the system will prompt for an id again. Thus, once the system starts, it will not stop.

```
public class Die
{
    private final int MAX = 6; // maximum face value
    private int faceValue; // current value showing on the die
    public Die(){
        faceValue = 1;
    }
    public int roll(){
        faceValue = (int) (Math.random() * MAX) + 1;
        return faceValue;
    }
    public void setFaceValue(int value){
        faceValue = value;
    }
    public int getFaceValue(){
        return faceValue;
    }
    public String toString(){
        String result = Integer.toString(faceValue);
        return result;
    }
}
```