0089

The Open University of Sri Lanka
Faculty of Engineering Technology
Department of Electrical and Computer Engineering

Study Programme	: Bachelor of Science Honours in Engineering
Name of the Examination	: Final Examination
Course Code and Title	: EEX3262 Object Oriented Programming
Academic Year	: 2022 - 2023
Date	: 17th February 2024
Time	: 09:30 -12:30 hrs
Duration	: 3 hours

General Instructions

1. Read all instructions carefully before answering the questions.
2. Answers should be written in answer book provided. Answer for each question should commence from a new page.
3. This question paper consists of **Four (4)** questions in **Six(06)** pages including this cover page.
4. Answer all questions.
5. This is a Closed Book Test **(CBT).**
6. Answers should be in clear handwriting.
7. Do not use red colour pen.

Question 1 (20 Marks)

State whether the following statements are true or false in Java Programming Language. Clearly explain why the statement is true or false. Use suitable examples where applicable.
(Note: Answers without the correct explanation will not score any marks)

1. It is possible to create instances of abstract classes.
2. Multiple inheritance is supported through the ***extends*** key word as shown in the following example: public class NewClass extends class1, class2{}
3. Using the ***continue*** statement in looping statement will make the program finish the current loop and start executing the statement following the loop.
4. Java interfaces may not contain any methods with implementation.
5. The ***static*** keyword can be applied to methods and variables in a Java program.
6. The concept of encapsulation strictly requires that all members of a class are declared as private.
7. A method in a class can be overloaded by changing its return type.
8. In Java a top-level class can be declared using the ***static*** keyword.
9. It is compulsory to define at least one constructor when creating a class.
10. It is possible to create sub-classes of a class declared with the keyword ***final,*** but these sub-classes must belong to the same package.

(2 x 10 = 20 Marks)

Question 2 (40 Marks)

Consider the following scenario:
A bank uses the following encrypting mechanism to encode confidential information that it transmits. All messages contain only capital letters, numbers and Space. Each character in the message is converted into an integer value.
The encrypted integer value for a character is obtained as follows:

- If the character to encrypt is a letter, first find the position of the character within the alphabet and add 15 to this value, square the resulting value and add a number called the key to get the encrypted value. Letter 'A' is in position 1, letter 'B' in position 2 and so on.

  For example, if the key is 13, then the encrypted value of letter 'B' would be $(15 + 2)^2 + 13 = 289 + 13 = 302$. Encrypted value of letter 'E' would be $(15 + 5)^2 + 13 = 400 + 13 = 413$.

- If the character is a number then add one to the number, find the square of the resultant number and add the value of the key to get the encrypted value for the characters.

  For example, if the key is 13, then the encrypted value of number 1 would be $(1 + 1)^2 + 13 = 17$. Encrypted value of 5 would be $(5 + 1)^2 + 13 = 49$

- The space character will be mapped to 13.

  In this scheme, if the bank wants to send the message "SEND RS100" then it will be encrypted to a sequence of numbers "1169 413 854 374 13 1102 1169 17 14 14" and transmitted.

The following shows a screenshot of the output of a program written for this purpose.

```
D: Code>java Encrypter
Enter message to encrypt:
SEND RS100000
Enrypted message is : [1169, 413, 854, 374, 13,  1102, 1169, 17, 14, 14, 14, 14, 14 ]
Decrypting the message : SEND RS100000
```

Carry out the following tasks:

(a) Create a class called ***Encrypter*** with the following characteristics:

(i) Instances of the class can be created without providing a value or by providing an encryption key as a parameter. If no value is provided as a parameter then the key will be 13.

(5 Marks)

(ii) Has a method called ***encryptChar*** that takes a character as the argument and returns the encrypted value of the character.

(10 Marks)

(iii) Has a method called ***decryptChar*** which takes an integer value and returns the decrypted character.

(10 Marks)

(b) Create a class called ***EncrypterTest*** that will test the functionality of the ***Encrypter*** class. The ***main*** method will read a message from the keyboard and encrypts each character using the ***encryptChar*** method. Each encrypted value is stored in a suitable data structure. Then the program will use the ***decryptChar*** method to decrypt the encrypted message.

(15 Marks)

Question 3: (20 Marks)

(a) Describe the role of each of the following keywords in Java and give examples of their use.

(i) import

(ii) extends

(iii) new

(iv) protected

(1.5 x 4 = 06 Marks)

(b) A student has created the following classes as part of a class exercise.

```
class Animal {
    private  int animalCount = 0;
    private String name;

    public Animal(String name) {
        this.name = name;
        animalCount++;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void makeSound() {
        System.out.println(this.name + "
makes a sound.");
    }

    public int getAnimalCount() {
        return animalCount;
    }
}
```

```
class Dog extends Animal {
    public Dog(String name) {
        super(name);
    }

    public void makeSound() {

System.out.println(getName() + "
barks.");
    }
}

class Cat extends Animal {
    public Cat(String name) {
        super(name);
    }

    public void makeSound() {

System.out.println(getName() + "
meows.");
    }
}
```

The student created a class called TestAnimal as given below to test the above classes

```
public class TestAnimal{
1.    public class TestAnimal{
2.        public static void main(String[] args) {
3.            Animal anAnimal = new Animal("Generic Animal");
4.            Dog rex = new Dog("Rex");
5.            Cat whiskers = new Cat("Whiskers");
6.            anAnimal.makeSound();
7.            rex.makeSound();
8.            whiskers.makeSound();
9.            System.out.println("Total animals created: " +
      anAnimal.getAnimalCount());
```

```
10.         }
11.       }
12.     }
```

(i) What is the output of the above program?

(03 marks)

(ii) The Student inserted the following statement after line no 2 in the code for TestAnimal class.

System.out.println("Total animals created: " + Animal.getAnimalCount());

When he tried to compile the program, the following error occurred.

```
D:\Programs\Java Code>javac TestAnimal.java
TestAnimal.java:49: error: non-static method getAnimalCount() cannot
be referenced from a static context
  System.out.println("Total animals created: " +
Animal.getAnimalCount());
                                                ^
1 error
```

(I) Explain the reason for the compilation error.

(03 marks)

(II) What are the changes that need to be made so the code for TestAnimal class runs with the newly included statement.

(08 marks)

Question 4:

(a) You are given a legacy program written by an unorganised programmer and asked to refactor the code. Explain 5 steps you would take to refactor the code for readability and maintainability. Use suitable examples to illustrate your answer.

(1 x 5 = 05 marks)

(b) Consider the following code segments and apply techniques for code refactoring where applicable. Your answer should include an explanation of why you considered refactoring is necessary.

(i)
```
public class UserAccess {
    private boolean isPremiumUser;
    private int age;

    public boolean grantAccess() {
        if (isPremiumUser) {
            return true;
        } else if (age > 18) {
            return true;
        } else {
            return false;
```

```
    }
  }
}
```

(ii)
```
public class Calculator {
  public void printSumAndAverage(int[] numbers) {
    int sum = 0;
    for (int i = 0; i < numbers.length; i++) {
      sum += numbers[i];
    }
    double average = sum / (double) numbers.length;
    System.out.println("Sum: " + sum + ", Average: " + average);
  }
}
```

(iii)
```
public class Order {
  private double price;
  private double discount;

  public void applyDiscount(boolean isHoliday) {
    if (isHoliday) {
      price -= price * 0.1; // 10% discount
      System.out.println("Discount applied.");
    } else {
      System.out.println("Discount applied.");
    }
  }
}
```

(5 x 3 = 15 marks)

---------- END OF PAPER ----------