Programme Code: TU857
Shared with: N/A
Module Code: CMPU 2016
CRN:23094

TECHNOLOGICAL UNIVERSITY DUBLIN

CITY CAMPUS - GRANGEGORMAN

TU857 – BSc in Computer Science (Infrastructure)

Year 2

SEMESTER 2
EXAMINATIONS 2023/24

Object Oriented Programming

Internal Examiners:
Dr. Bianca Schoen-Phelan
Dr. Sunder Ali Khowaja
Dr. Paul Doyle

External Examiner:
Ms. Caroline McEnroy

Exam Duration: 3 hours

Instructions: There are two parts to this exam. Please answer 2 out of 3 questions from Part A Python, and 2 out of 3 questions from Part B Java.

Part A – Python

1. **(a)** Describe the difference between a Python **list** and a Python **tuple** in Python. Give an example of a situation where a tuple might be more appropriate than a list. (5 marks)

(b) In your own words explain the difference between a **class** and an **object** in OOP? Illustrate your answer with a Python example. Explain your code illustration. (8 marks)

(c) Provided is the following Python code in Listing 1:

```
class Vehicle:
    def __init__(self, name):
        self.name = name

    def start(self):
        print(f"{self.name} vehicle started")

class Car(Vehicle):
    def start(self):
        super().start()
        print(f"{self.name} car started")
```

Listing 1: Python code example.

(i) Explain what happens when the **start** method is called on an instance of **Car**. (6 marks)

(ii) Modify the **Car** class so that it also includes a **stop** method which is not present in **Vehicle**. (6 marks)

2. **(a)** Explain the difference between a **for** loop and a **while** loop in Python. When would you use one over the other? (5 marks)

(b) In your own words explain the concept of inheritance in OOP. Provide a Python example. Explain your code choices. (8 marks)

Question 2 continues on the next page.

2. **(c)** Provided is the following Python code in Listing 2:

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

class Employee(Person):
    def __init__(self, name, age, employee_id):
        super().__init__(name, age)
        self.employee_id = employee_id
```

Listing 2: Python code example.

(i) Considering a focus on OOP software development, identify and explain any issues with the **Person** class in Listing 2. (6 marks)

(ii) Re-write the **Person** class provided in Listing 2 to present an improved version that corrects any issues you have identified in question 2)c)i). (6 marks)

3. **(a)** Explain what an exception is in Python and how exception handling works. Provide a basic example of using a **try-except** block. (5 marks)

(b) In your own worlds explain what abstraction in OOP is, and how it is different from encapsulation. (8 marks)

(c) **(i)** In your own words explain the purpose and importance of unit testing in software development. What role does the **unittest** framework play in Python for achieving this? (6 marks)

(ii) Consider a fictional Python module named **`math_operations.py`** that contains a function **`def multiply(x, y).`**

Describe how you would structure a test case for this function using the **unittest** framework. Include the following in your answer:

1. The import statement required to use **unittest**.
2. The general structure of a test case class.
3. The setup method, if necessary.
4. How would you name the test method for testing the **multiply** function.
5. The kind of assertions you would use to verify that **multiply** works as expected. (6 marks)

Part B – Java

1. **(a)** Describe the distinctions among a Class, Variables, and Methods. Provide a basic code snippet illustrating the differentiation of these terms. (6 marks)

(b) In your own words explain the difference between an **object** and a **constructor** in OOP. Illustrate your answer with a Java example. Explain your code illustration. (6 marks)

(c) Provided is the following Java code in Listing 1:

```
class Vehicle {
    int maxSpeed = 120;
}

class Car extends Vehicle {
    int maxSpeed = 180;

    void display()
    {
       System.out.println("Maximum Speed: "
                          + super.maxSpeed);

       System.out.println("Maximum Speed of this
                          car is: " + maxSpeed);
    }
}

class Demo {
        public static void main(String[] args)
        {
           Car obj = new Car();
           Obj.display();
         }
}
```

Listing 1: Java code example.

(i) Explain what happens when the **display** method is called on an instance of **Car**. (6 marks)

Question 1 (c) continues on the next page.

1. **(c)** **(ii)** Modify the **Car** class so that it also includes a **stop and brakes** method which is not present in **Vehicle**. (7 marks)

2. **(a)** Explain the difference between **overriding and overloading** in Java. Give example Java codes for exhibiting the characteristics of overriding and overloading. (Hint: Given the animal class below, you can extend it in a class named cat and exhibit the overriding characteristic in the main class.) (8 marks)

```
class Animal {
      void eat()
       {  System.out.println("eat() method of base
                              class");
           System.out.println("eating…");}
}
```

(b) In your own words explain the concept of inheritance in OOP. Provide a Java example. Explain your code choices. (8 marks)

Question 2 continues on the next page.

2. **(c)** Provided is the following Java code in Listing 2:

```
interface Bird {
      int age;
      string name;
      public abstract void sing();
}

class Eagle implements Bird {
      public void sing(){
              System.out.println("Singing!");
      }
}

public class Listing2 {
      public static void main(String[] args) {
       Eagle e = new Eagle();
       System.out.println(e.age);
       System.out.println(e.sing());
     }
}
```

Listing 2: Java code example.

(i) Considering a focus on OOP, identify and explain any issues with the **code** in Listing 2. (3 marks)

(ii) Re-write the **code in Listing 2 such that**: the int and name are removed. Make the interface Bird a marker interface and run the method Sing(); only if the Eagle is an instance of the Bird. (6 marks)

3. **(a)** What are the different types of layouts in Java (refer to Java Swing), define at least three? (9 marks)

Consider the following code in Listing 3. What changes will you make to visualize something and what output will you get? What changes will you make to the code to observe all the labels?

```
import javax.swing.*;

public class DemoFrame {
         public static void main(String[] args) {
                 test obj = new test();
           }
}

class test extends JFrame {
             public test() {
                JLabel labeltest = new JLabel("Hello
                                   World");
                JLabel labeltest1 = new JLabel("Welcome
                                   to TUD");
                JLabel labeltest2 = new JLabel("TU 857
                                   class");

                add(labeltest);
                add(labeltest1);
                add(labeltest2);

           }
}
```

Listing 3. Java code example. (9 marks)

(b) In your own words, explain the difference between "List" and "Set" in Java through examples and syntaxes. Limit your code to declaration only. You do not need to add print statements. (7 marks)

(c) **(i)** What is the main benefit of using the properties file? An example of where can we use properties file? Provide an example Java code. (5 marks)

(ii) How would you convert an ArrayList to Array and Array to ArrayList?
How will you reverse a List?
Justify your answer with Java code and syntax. (4 marks)