National University of Computer & Emerging Sciences, Karachi
Spring-2023 School of Computing (BSCS, BSCS-R, BSSE, BSCY, BSAI)
FINAL EXAMINATION
22nd May 2023, 12:00 noon – 03:00 pm

Course Code: CS1004	**Course Name: Object Oriented Programming**
Instructors Name: Dr. Farooque Hassan Kumbhar, Dr. Abdul Aziz, Mr. Zain-ul-Hassan, Ms. Abeer Gauher, Mr. Basit Ali, Ms. Sobia Iftikhar, Ms. Aqsa Zahid, Ms. Sumaiyah, Ms. Abeeha Sattar, Ms Javeria Farooq, Mr. Shahroz Bakht, Ms. Eman Shahid	
Student Roll No:	**Section No:**

Instructions:

- Return the question paper and make sure to keep it inside your answer sheet.
- Read questions completely before answering. There are **5 questions, 6 sides on 3 pages**.
- In case of any ambiguity, you may make assumptions. But your assumption should not contradict any statement in the question paper.
- You are **not allowed to write** anything on the question paper (except your ID and section).

Time: 180 minutes. **Max Marks**: 110 Marks

Q1: [20 min, 20 Marks, CLO 1] what will be the output of the programs when they are executed? There are no compilation errors in the programs.

1.
```
class Point {
private int x;
private int y;
    public Point Point(int i, int j){
    x = i;
    y = j;
    System.out.println("Normal   Constructor
called”);
    }

    Point(Point t){
    y = t.y;
        System.out.println("Copy Constructor
called”);
    }

void display ( ){
     System.out.println("Value of x =" +x);
     System.out.println("Value of y =" +y);
    }

public static void main(String [ ] args)
{
   Point t1 =  new Point(10, 15);
   Point t2 = new Point (t1);
   Point t3 = t1;
   t3.display();

}
```

2.
```
class Test {
    private static int count;

    Test fun() {
        count++;
        System.out.println(count);
        return this;
    }


    public static void main(String[] args) {
        Test t = new Test();
        t.fun().fun().fun().fun();
        Test t2 = new Test();
        t2.fun().fun().fun().fun();

    }
}
```

3.
```
class Base {
    public static int count;

    public Base() {
        count++;
    }

    protected void finalize() {
        count--;
    }

    static void printCount() {
        System.out.println("Count:    "    +
count);
    }
}
class Derived extends Base {

    public Derived() {
        count++;
    }

    protected void finalize() {
        super.finalize( );
    }
}
class Test {
    public static void main(String[] args) {
        Base.printCount();
        Base obj1 = new Base();
        Derived obj2 = new Derived();
        Base.printCount();
        obj2.finalize();
        Derived.printCount();
        obj1.finalize();
        Base.printCount();

    }
}
```

4.
```
class A
{
    public A (){
        System.out.println("A's
constructor"); }
    public A (A a) {
        System.out.println("A'sCopy
constructor");
    }
}

class B
{
    A a;
    B(A a) {
        this.a = a;
        System.out.println("B's
constructor");
    }
}
class Main {

    public static void main(String[] args) {
        A a1 = new A();
        B b1 = new B(a1);

    }
}
```

Q2: [30 min, 20 Marks, CLO 2] considering the output given, complete the following code snippets.

1.
```
public static <T> void swap(){ - - - - -    }
public static void main(String[] args) {
        Integer[] intArray = {1, 2, 3, 4, 5};
        String[] stringArray = {"Hello", "World"};

        swap(intArray, 1, 3);
        swap(stringArray, 0, 1);

        System.out.println("After swap:");
        System.out.println("Integer Array: " + Arrays.toString(intArray));
        System.out.println("String Array: " + Arrays.toString(stringArray));
    }
```

```
Output:
After swap:
Integer Array: [1, 4,
3, 2, 5]
String        Array:
[World, Hello]
```

2.
```
class Testing { - - - - - -  }


    public static void main(String[] args) {
        Testing example = new Testing();

        example.sum(5, 10);
        example.sum(1, 2, 3);
    }
}
```

```
Output:
Sum of two integers:
15
Sum     of     three
integers: 6
```

3. class Base { - - - - - - - - }
class Derived extends Base { - - - - - - - }
 	
public static void main(String[] args) {
Base ptr = new Derived();
ptr.finalize();
}
}
	Output:
	"BaseDerivedDerivedBase"

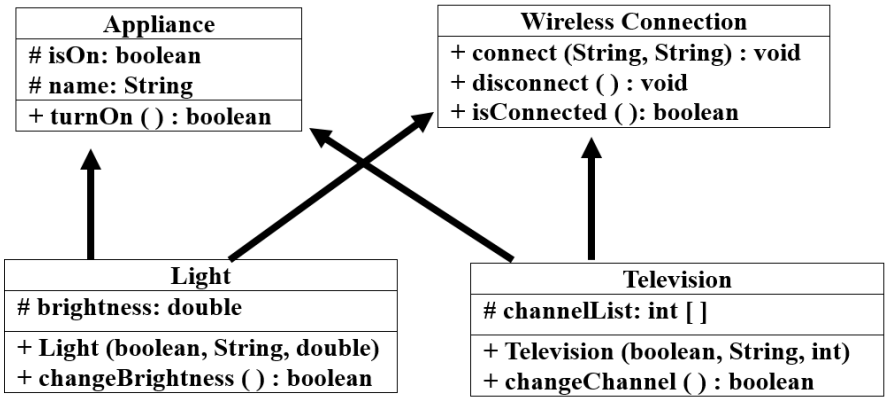
4. public class OuterClass {
private int outerData;
public OuterClass(int data) {
outerData = data;
}
 	
public void outerMethod() {
System.out.println("Outer method");
}
 	
public class InnerClass { - - - - - - - - }
 	
public static void main(String[] args) {
OuterClass outerObj = new OuterClass(10);
OuterClass.InnerClass innerObj = outerObj.new InnerClass(20);
 	
outerObj.outerMethod();
innerObj.innerMethod();
innerObj.accessOuterData();
}
}
	Output:
	Outer method
	Inner method
	Outer data: 10

5. abstract class Person {
protected String name,age;
Person(String name, String age) {
this. name= name;
this.age=age;
}
abstract void saveToFile(Student s);
}
class Student extends Person {
 	
private String university;
 	
Student(String name, String age, String university) {
super(name, age);
this.university = university;
}
 	
void saveToFile(Student s) { - - - - - - - - - }
 	
public static void main(String[] args) {
Student student = new Student("Ali", "21", "FAST NUCES");
student.saveToFile(student);
}
}
	Output:
	Data saved to file with details : Ali 21 FAST NUCES

```
6. class GenericStack<T> {
    private ArrayList<T> stack;
    public GenericStack() {
        stack = new ArrayList<>();
    }

    public void push(T item) { - - - - - -  }

    public static <E> void printStack(GenericStack<E> stack) { - - - - - -}


    public static void main(String[] args) {
        GenericStack<Integer> intStack = new GenericStack<>();
        intStack.push(1);
        intStack.push(2);
        intStack.push(3);
        System.out.println("Integer Stack:");
        printStack(intStack);

        GenericStack<String> stringStack = new GenericStack<>();
        stringStack.push("Hello");
        stringStack.push("World");
        System.out.println("String Stack:");
        printStack(stringStack);
    }}
```

```
Output:
Integer Stack:
3
2
1
String Stack:
World
Hello
```

Q3: [30-40 min, 20 Marks, CLO 3]



Your task is to design a smart home application that contain various types of devices. Implement the class diagram shown above, according to the description given in each part of the question.

a) You are supposed to design a class hierarchy according to the rules specify by the object oriented paradigm and its implementation in accordance to Java. **(4)**

b) The following functions are implemented in the specialized classes according to the description given: **(6)**

1. turnOn: turns on the specific appliance.
2. connect: takes in the name of network and password as parameters. If the name of the network and password matches within the list of networks and their passwords, then set connected as true, otherwise display “Failed to connect”.
3. disconnect: a device is disconnected automatically if the device is powered off or if there is no access point within (0m – 45m) range. Otherwise display “Failed to disconnect”.
4. isConnected: checks whether the device is connected or disconnected

c) The class “Light” has a function changeBrightness ( ) that changes the brightness according to the user’s needs. **(5)**

If the light is turned On and is connected by the wireless connection, then the user can either increase or decrease the brightness level. For example, if the brightness is set to 10 and the

user wants to change it to 20 then add 10 to the current brightness level. The brightness level can never be negative or larger than 20.

d) The class “Television” has a function changeChannel ( ) that changes the channel as the user wants. If the television is turned On and is connected by the wireless connection, change the channel only if the channel exists in the channelList. **(5)**

Q4: [30-40 min, 20 Marks, CLO 4]

As a JAVA programmer, you are required to develop a program that manages an e- commerce system. Users are able to purchase products according to their needs.

a) Create a generic class named “Product”. The product class has attributes name, price, manufacturer and stock level. Create a parameterized constructor that sets these attributes. **(6)**

b) Create a generic member inner class in Product named as “Shopping Cart” that has a list of products ArrayList<Product<T>> products as an attribute. Create a default constructor that initializes the arraylist. The class has the following functions: **(2)**

1. void addProduct(Product<T> product): adds a product that the user purchases. **(2)**
2. void removeProduct(Product<T> product) : asks the user which product he wants to remove from the shopping cart. If the product exists, remove it from the list, otherwise display “Product is not in the list”. **(2)**
3. double calculateTotalPrice(): calculates the total price of all the products purchased. The calculation is done as follows: **(2)**
    1. Ask the quantity to be purchased for a particular product.
    2. If the quantity does not exceed the stock level, then calculate the price or else display “Quantity exceeds stock level”.

c) Create another generic member class in Product named as “Order” that has OrderID and ShoppingCart<T> shoppingCart as attributes. The class has the following functions: **(2)**

1. generateOrderID ( ): The order ID is generated by taking in the first and last letter of the customer’s name and concatenating it with the last 2 digits of the customer’s phone number (11 digits). **(2)**
2. ShoppingCart<T> getShoppingCart(): displays all the products that are in the shopping cart. **(2)**

Q5: [40-50 min, 10+10+10 Marks, CLO 4] Consider a chatbot system designed to provide responses to users' queries. The system consists of four chatbot variants tailored for medical, technology, legal, and general queries. Your task is to implement an object-oriented program that fulfills the following requirements:

a) Design a User class to store user data, including attributes such as username, country, interest, and age. User have a method **Ask ()** which takes a string as a query and generates a specific response.

- The medical chatbot should respond if the query's prefix (first word) is "doc". The legal chatbot should respond if the query begins with "attorney". The technology chatbot should respond only if the query starts with "guru".
- If a query begins with "special", check the user's interest, and forward the query to the relevant chatbot variant based on their interest.
- If a query does not match the relevant prefix, it must throw a custom exception object of type "Bot_Exception" with an error message.

b) Implement a Chatbot class as the base class for all chatbot variants. Each chatbot variant (MedicalChatbot, TechnologyChatbot, LegalChatbot, GeneralChatbot) should be inherited from the Chatbot class. Make sure that each chatbot class should keep track of the number of instances created throughout the program.
    - Each chatbot variant should have a method **string generate_response (string query, User u)** to generate responses based on user queries. It should store the name of the most recent user that interacted with it and maintain a total user count, tracking the number of users who have ever interacted with it.
    - Provide functionality to access the total user count for each chatbot variant at any given time.

c) As specified in part A, when a chatbot variant throws a "Bot_Exception", an error message should notify the user that the query is invalid. Capture the username and query of the user causing the exception and write it to the "error_log.txt" file. The "error_log.txt" file should contain a list of usernames + queries for users who caused exceptions to be thrown. Also, you need to write a function “Analysis()” that will open the file “error_log.txt” in read mode and will perform the following analysis:
    - It will print the username who caused the maximum number of exceptions.
    - It will print the total count of words from each query stored.

Note: Ensure that your program demonstrates proper usage of object-oriented principles such as inheritance, encapsulation, exception handling, generics, and file handling. Implement appropriate methods and attributes in each class to fulfill the requirements outlined above.

BEST OF LUCK!