United International University (UIU)

Dept. of Computer Science & Engineering (CSE)

Final Exam, Trimester: Fall 2022

Course Code: CSE-1115, Course Title: Object Oriented Programming

Total Marks: 40, Duration: 2 hours

Any examinee found adopting unfair means will be expelled from the trimester / program as per UIU disciplinary rules.

1. Answer the following questions: **3+4+3**
   a. Write about the advantage of using interface over abstract class. Is multiple inheritance possible in java? Explain.
   b. Find out the errors (if there are any) and correct them **without writing any code inside any class or interface.**

```
abstract class abs {
   abstract public void m1();
   public void m2(){}
}

Interface I1{
   public void m3();
   public void m4();
}

Interface I2{
   public void m5();
   public void m6();
}

class Abstraction extends abs implements I1, I2{
   public void m1(){};
   public void m3(){};
   public void m4(){};
   public void m5(){};

}
```

   c. Will the following code compile successfully? If yes, what will be the output of program? If no, why?

```
public abstract class A {
  abstract void m1(A a);
}
class B extends A {
  void m1(A a){
       System.out.println("One");
  }
}
class C extends B {
  void m1(B b) {
       System.out.println("Two");
       super.m1(new B());
       System.out.println("Three");
  }
}
class Test {
   public static void main(String[] args) {
   C c = new C();
   c.m1(new B());
    }
}
```

2. Answer the following questions: 5+5

a. Modify the following code so that it runs without error and write the output of the following program

```
public class Test{

	int TestA(){
		throw new Exception("Exception has been thrown from TestA");
		System.out.println("Don't forget me!");
		return 0;
	}

	void TestB(){
		int a = TestA();

		if(a == 0){
			System.out.println("Exception in TestB: a is zero.");
		}
		else{
			System.out.println("You should figure out the value of a: " + a);
			throw new Exception("Exception from else clause in TestB");
		}

		return;
	}

	boolean TestC(boolean val){
		if(val == false){
			throw new Exception("Exception in TestC");
		}

		return false;
	}
	 public static void main(String []args){
		try{

			TestC(false);
		}
		catch(Exception e){
			System.out.println(e);
			try{
				TestB();
			}
			catch(Exception e1){
				System.out.println(e1);
			}
		}
	}
}
```

b. You need to implement search function in the following code. The search function tries to find an element using linear search. If the element is not present in the array, then it throws *ElementNotFoundException.* Also, you need to throw *ArrayNotInitializeException* if the given array has not been instantiated.

```
public class ArrayList {

    int []arr;
    public boolean search(int element){
        // implement the code here
        for(int i = 0; i< arr.length; i++){
            // implement the code here
        }
        // implement the code here
    }
}
```

3. Answer the following Questions: **5+5**

a. You are given a file named "person.txt". Each line of the file contains information about a person, containing id, age and nationality (divided by forward slash). The file looks like this:

```
1/20/Bangladeshi
2/23/Argentinian
3/35/Portuguese
```

Write a function that will return the id of a person having maximum age (If there are multiple persons with maximum age, return any one of them).

b. Create and assign an object of Person using anonymous inner class for both of Line 7 and 8 to produce output "Hello, I'm an Engineer" from Line number 10 and "Hello, I'm a doctor" from Line number 11.

```
1  interface Person {
2    void introduce();
3  }
4
5  public class AnnonEx {
6    public static void main(String[] args) {
7      Person engineer; // Write your codes
8      Person doctor; // Write your codes
9
10     engineer.introduce(); // should print "Hello, I'm an Engineer"
11     doctor.introduce(); // should print "Hello, I'm a Doctor"
12   }
13 }
```

4. Answer the Following Questions (You must answer **a, b** and any one of the **c or d**): **3+4+3**

a. Remember how you raced with your friends in your childhood? Someone shouted: 3…2…1…GO! And you started running! Now, create a simple GUI application that shows **3…2…1…GO!** The GUI will have only 2 components in a Frame: **A Label and a Button**. The label will not show any text in the beginning. When you press the button the first time, the label will show "3…". The next time you press

the button, the label will show “2…”, then “1…” and lastly “GO!”. Some parts of the code are done for you, you will need to complete the rest (Consider appropriate classes are imported).

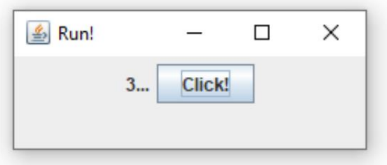
```
class Main {
    public static void main(String[] args) {
        JFrame fr = new JFrame("Run!");
        fr.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        fr.setSize(250, 100);

        JLabel label = new JLabel();
        JButton button = new JButton("Click!");

        // Write your code here.

        fr.setVisible(true);
    }
}
```

The GUI after you clicked the button once:



b. Analyze the given code and answer the following questions

i. In **main()** method we declare two objects of **BloodDonor**. So according to our logic, **member** value will be two doesnot matter **getMember()** method called by whom. But practically, we always get 1 as the output of **getMember()**. Now, change the declaration of the **member** variable in the BloodDonor class so that **getMember()** will provide output for as many objects as we create. You can’t change the remaining codes.

ii. After the above corrections, are there still any errors? If yes, then after correction, find the output. If not, then write "No Errors" and find the output.

```
class BloodDonor{
    private int member;
    public BloodDonor(){
        member++;
    }
    int getMember(){
        return member;
    }
    void donate(String d){
        final String donor;
        donor = d;

System.out.println("Donor is:
"+donor);
    }
}
```

```
public class Main
{
 public static void main(String[] args)
 {
  BloodDonor donor1 = new BloodDonor();
  BloodDonor donor2 = new BloodDonor();

System.out.println(donor1.getMember());
//it prints 1

System.out.println(donor2.getMember());
// it also prints 1 but actually both
should print 2
    donor1.donate("Mr. Ali");
    donor2.donate("Afsana");

  }

}
```

c. Consider the following code:

```
class Student{
    String name;
    double height;
    Student(String name, double height){
        this.name = name;
        this.height = height;
    }
}

public class ArrayListDemo{
    public static void main(String []args){
        // your code here
    }
}
```

i. Create an Arraylist of Student class inside the main function. Insert five Student objects into the Arraylist. Initialize the Student objects with random but proper values.

ii. Add necessary code to sort the Arraylist created in (i) according to height in descending order.

OR

d. Write two ways of implementing multi-threading.