

Advanced Programming (CSE201), Endsem Quiz
Time allocated: 10:30am – 12:00noon (1.30 hours), Total Marks: 30
(No Submissions are allowed beyond 12.00noon)

Instructions:

- This is a closed book quiz.
- Only reasonable and clearly mentioned assumptions (if any) would be accepted.
- For justifications, please be as concise as possible (2-3 sentences only)
- **IIIT plagiarism policy is applicable if any such cases found**
- **Write your answers on a plain sheet that you can upload by taking a picture of the same (ensure low resolution so that the upload size is smaller)**
- **You can either submit your solutions on the below Google Form link**
https://docs.google.com/forms/d/e/1FAIpQLSc3vCUr1_sIIYetSf78XuX2p0jMvn9sUa2FhA8yomlpbJ5mew/viewform?usp=sf_link
or email the quiz solutions to “ap-m2020-submission@iiitd.ac.in” or upload on google classroom . Only one mode of submission!
- **Subject of the mail should be the Endsem Quiz.**
- **We will not consider any submission that is submitted beyond 12.00noon. It is your responsibility to ensure you email it or submit through google form or through google classroom on time. Please ensure you have proper internet connectivity as we are giving you sufficient extra time to send the email or upload your solutions either on google classroom or on google form.**

Question-1: Is / Are there any issue in this code (compilation / runtime exception / improper design)? If yes, then identify it and write the correct code. [7 marks]

```
1. import java.util.*;
2. public class A implements Cloneable {
3.     private int a1;
4.     public A (int a) { a1 = a; }
5.     public void print() {
6.         System.out.println(a1);
7.     }
8.     public A clone() throws CloneNotSupportedException {
9.         return (A) super.clone();
10.    }
11. }

1. public class B extends A {
2.     private List<Integer> b1 = new ArrayList<Integer>();
3.     public B (int b) { b1.add(b); }
4.     public void add(int x) {b1.add(x); }
5.     public void print() {
6.         for(int b : b1) {
7.             System.out.println(b);
8.         }
9.     }
10. }

1. public class C {
2.     public static void main(String[] args) {
3.         B b1 = new B(1);
4.         B b2 = b1.clone();
5.         b2.add(10);
6.         b1.print();
7.         b2.print();
8.     }
9. }
```

Question-2: Identify the issues with the following code in terms of bad object-oriented design choices. Why are they considered bad? [3 marks]

```
1.class University {
2.    public List<String> courseList;
3.    public List<String> studentList;
4.    protected List<String> facultyList;
5.    public List<String> enrollmentList;
6.    public String universityName;
7.    public String universityLocation;
8.
9.    public void registerForCourse(String student, String course) {
10.        // method elided
11.    }
12.    public boolean checkFacultyTeachingLoad() {
13.        // method elided
14.        return false;
15.    }
16.    public int getCoursesTaughtByFaculty(String faculty) {
17.        // method elided
18.        return 0;
19.    }
20.    public static String getAdminStaffQualifications() {
21.        // method elided
22.        return null;
23.    }
```

```
1. interface Faculty {
2.    public void teachCourse(String course);
3.    public void adviseStudents(List<String> students);
4.}
5. interface AssistantProfessor extends Faculty {
6.    public int getCourseLoadRestriction();
7.}
8. interface AssociateProfessor extends Faculty {
9.    public int getAdminLoadRestriction();
10.}
```

Question-3: Would there be any issues with the following code (assume there are no compilation or runtime errors). What could be the reason(s) for those issues and how would you resolve them? [3 marks]

```

1. public class ThreadDemo implements Runnable {
2.     public static void main(String[] args) {
3.         for(int i=0; i<100; i++) {
4.             Thread t = new Thread(new ThreadDemo(), "T"+i);
5.             int threadPriority = ThreadLocalRandom.current().nextInt(
6.                 Thread.MIN_PRIORITY, Thread.MAX_PRIORITY+1);
7.             t.setPriority(threadPriority);
8.             t.start();
9.         }
10.    }
11.    @Override
12.    public void run() {
13.        try {
14.            Thread.sleep(5000);
15.            synchMethod();
16.        } catch (InterruptedException e) {
17.            e.printStackTrace();
18.        }
19.    }
20.    public synchronized void synchMethod() {
21.        //Heavy computation is done here
22.        System.out.println(Thread.currentThread().getName() + " " +
23.            Thread.currentThread().getPriority());
24.        try {
25.            Thread.sleep(1000);
26.        } catch (InterruptedException e) {
27.            e.printStackTrace();
28.        }
29.    }
30.}

```

Question-4:

- a) In the following code, identify the design patterns that have been implemented or used. In each case, explain how you identified the design pattern. **[10 marks]**
- b) Draw the UML diagram based on the code. **[5 marks]**

```
1. abstract class Noname0 {
2.     public final void noMethodName0_0() {
3.         noMethodName0_1();
4.         noMethodName0_2();
5.         noMethodName0_3();
6.         noMethodName0_4();
7.     }
8.     public void noMethodName0_1() {
9.         System.out.println("This is noMethodName0_1 in Noname0");
10.    }
11.    public void noMethodName0_3() {
12.        System.out.println("This is noMethodName0_3 in Noname0");
13.    }
14.    public abstract void noMethodName0_2();
15.    public abstract void noMethodName0_4();
16. }
```

```
1.class Noname1 extends Noname0 {
2.    private static Noname1 noname1;
3.    private Noname1() {}
4.    public static synchronized Noname1 noMethodName1_0() {
5.        if(noname1 == null)
6.            noname1 = new Noname1();
7.        return noname1;
8.    }
9.    @Override
10.    public void noMethodName0_2() {
11.        String s1 = "abc";
12.        String s2 = new String("abc");
13.        System.out.println(s1 + "and " + s2 + " are equal");
14.        System.out.println("This is noMethodName0_2 in Noname1");
15.    }
16.    @Override
17.    public void noMethodName0_4() {
18.        System.out.println("This is noMethodName0_4 in Noname1");
19.    }
20. }
```

```

1.class Noname2 extends Noname0 {
2.    @Override
3.    public void noMethodName0_2() {
4.        System.out.println("This is noMethodName0_2 in Noname2");
5.    }
6.    @Override
7.    public void noMethodName0_4() {
8.        System.out.println("This is noMethodName0_4 in Noname2");
9.    }
10.}

1.class Noname3 {
2.    public Noname0 noMethodName3_0(int option) {
3.        if(option == 1)
4.            return Noname1.noMethodName1_0();
5.        else if(option == 2)
6.            return new Noname2();
7.        else
8.            return null;
9.    }
10.    public void noMethodName3_1(Collection<Noname2> collection) {
11.        Iterator<Noname2> collt = collection.iterator();
12.        while(collt.hasNext()) {
13.            System.out.println(collt.next());
14.        }
15.    }
16.}

```

Question-5: Convertible extends Car and Porsche extends Convertible. Car class has a method **drive** that is overridden in each child class mentioned above. The method **drive_car** below drives different types of Cars. Rewrite the declaration of this static method after performing a type erasure. [2 marks]

```
public static <T extends Car> void drive_car(T c) { c.drive(); }
```