BSCCS2005: Practice Assignment with Solutions
Week 9

1. Consider the following Java code and choose the correct option.

```
import java.util.*;
public class Test{
   public static void main(String[] args){
      String[] arr=new String[10];
      arr[0]="Sun";
      arr[1]="Moon";
      Optional<String> op=Optional.ofNullable(arr[2]);
      op.ifPresent(n->System.out.println(n.toUpperCase()));
   }
}
```

- ◯ This program generates `ArrayIndexOutOfBoundsException`.
- ◯ This program generates `NullPointerException`.
- ◯ This program generates output:
  null
- [✓ This program does not generate any output.]

Solution: `ifPresent(Consumer<? super T> consumer)` method invoke the specified `consumer` with value if the value is present in the `Optional`, otherwise does nothing.

2. Consider the code given below.

```
import java.util.*;
import java.util.stream.Stream;
public class OptionalExample{
    public static void main(String[] args){
        Optional<Double> maxvalue =
            Stream.generate(Math::random)
            .limit(100)
            .max(Double::compareTo);
        var list = new ArrayList<Double>();
        maxvalue.ifPresentOrElse(
                v -> System.out.println("max value found"),
                () -> System.out.println("No max")
            );
    }
}
```

Choose the correct option regarding the code.

- ◯ Compilation failed.
- ✓ This program generates the output.
  `max value found`
- ◯ This program generates output:
  `No max`
- ◯ This program generates no output.

Solution: Stream generates the 100 random values between 0 and 1, max() will return the maximum value among all random values. `ifPresentOrElse()` will check whether maxrand contain the value or not, if present it will print `max value found` else will print `No max`.

3. Consider the following Java code and choose the correct option.

```
import java.util.*;
import java.util.stream.*;
public class Test{
   public static void main(String[] args){
      var i = 10;
      var list = new ArrayList<Integer>();
      while(i>1){
         list.add(i);
         i = i-1;
      }
      Stream<Integer> stream=list.stream();
      Integer[] num=stream.filter(n->n<5).map(n->n*n).toArray(Integer[]::new);
      for(var x=0;x<4;x++){
         System.out.println(num[x]);
      }
   }
}
```

- ○ This program generates compile time error because stream can not be converted into array.
- ○ This program generates output:
  2
  3
  4
- √ This program prints 16, 9 and 4 followed by `ArrayIndexOutOfBoundsException`.
- ○ This program generates output:
  4
  9
  16

4. Consider the following Java code and choose the correct option.

```
import java.util.*;
import java.util.stream.*;
public class Employee{
   int id;
   String name;
   int service;
   Employee(int id, String name, int service){
        this.id = id;
        this.name = name;
        this.service = service;
   }
   public int getId(){
        return id;
   }
   public String getName(){
        return name;
   }
   public int getService(){
        return service;
   }
   public String toString(){
        return "Employee{" + "id=" + id + ", Name=" + name + ", Service=" +
                service + '}';
   }
}
public class FClass{
    public static void main(String[] args){
        var employee=new ArrayList<Employee>();
        employee.add(new Employee(1,"Mercury",10));
        employee.add(new Employee(2,"Venus",5));
        employee.add(new Employee(3,"Earth",3));
        employee.add(new Employee(6,"Saturn",2));
        employee.add(new Employee(7,"Uranus",10));
        employee.add(new Employee(8,"Neptune",10));
        Map<Integer, List<Employee>> map = employee.stream().
                collect(Collectors.groupingBy(i->i.getService()));
        System.out.println(map.get(10).get(1));
    }
}
```

○ This program generates compile time error because stream cannot be converted to map.

- [✓ This program generates output:
  Employee{Id=7, Name=Uranus, Service=10}]
- ◯ This program generates `KeyNotFoundException`.
- ◯ This program generates output:
  Employee{id=1, Name=Mercury, Service=10}

Solution: `groupingBy(Function<? super T,? extends K> classifier)` method returns a `Collector` implementing a ”group by” operation on input elements of type T. It groups elements according to a classification function, and returns the results in a Map.

5. Consider the following Java code and choose the correct option.

```
import java.util.*;
import java.util.stream.*;
public class Planet{
    private String name;
    private int temp;
    public Planet(String name, int temp) {
        this.name = name;
        this.temp = temp;
    }
    public String getName() {
        return name;
    }
    public int getTemp() {
        return temp;
    }
}
public class FClass{
    public static void main(String[] args){
        List<Planet> planet=new ArrayList();
        planet.add(new Planet("Mercury",480));
        planet.add(new Planet("Venus",430));
        planet.add(new Planet("Earth",30));
        planet.add(new Planet("Mars",-25));

        Set<String> set = planet.stream().
                filter(n->n.getTemp()>450).
                map(x->x.getName()).collect(Collectors.toSet());

        Map<Integer, String> map = planet.stream().
                filter(x->x.getTemp()<0).
                collect(Collectors.toMap(a->a.getTemp(), b->b.getName()));

        set.forEach(System.out::println);
        for (Map.Entry entry : map.entrySet()){
            System.out.println("key: " + entry.getKey() +
            "; value: " + entry.getValue());
        }
    }
}
```

- ◯ This program generates output:
  null

```
Mercury
key: -25; value: Mars
```

✓ This program generates output:
```
Mercury
key: -25; value: Mars
```

○ This program generates output:
```
Mercury
key: 480; value: Mercury
key: 430; value: Venus
key: 30; value: Earth
key: -25; value: Mars
```

○ This program generates output:
```
Jupiter
Mercury
key: -25; value: Mars
```

6. Consider the code given below.
Assume that there is no file named "`E:\\Files\\earth.txt`".

```
import java.io.*;
public class Example {
    public static void main(String[] args) {
        try {
            var in=new FileInputStream("E:\\Files\\earth.txt");
            var din=new DataInputStream(in);
            System.out.println("Data from file:");
            System.out.println(din.readLine());
        }
        catch (FileNotFoundException e) {
            System.out.println("File does not exist.");
        }
        catch (IOException e) {
            System.out.println("Error in writing a file.");
        }
        finally {
            System.out.println("Program execution finished.");
        }
    }
}
```

Choose the correct option regarding the code.

- ◯ Compilation error.
- ◯ This program terminates abnormally due to unhandled exception(s).
- ◯ This program generates output:
```
Error in writing a file.
Program execution finished.
```
- ✓ This program generates output:
```
File does not exist.
Program execution finished.
```

Solution: While writing data into the file, if there is no file exist, program throws FileNotFoundException.
catch block with FileNotFoundException is executed and finally block also executed in above program.

7. Consider the code given below.
   Assume that the file "`E:\\Files\\books.txt`" contains the following text in it.
   `Hi I am already here.`

```
import java.io.*;
public class Example {
    public static void main(String[] args) {
        try {
            var out=new FileOutputStream("E:\\Files\\books.txt",false);
            var dout=new DataOutputStream(out);
            String data="Hello I have added  to you.";
            dout.writeBytes(data);
            System.out.println("Data written to file successfully");
        }
        catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch (IOException e) {
            e.printStackTrace();
        }
        finally {
            System.out.println("Program execution finished.");
        }
    }
}
```

   Choose the correct option regarding the code.

   - ○ Compilation error.
   - ✓ After program execution "`E:\\Files\\books.txt`" contains the following line of text in it.
     `Hello I have added to you.`
   - ○ After program execution "`E:\\Files\\books.txt`" contains the following line of text in it.
     `Hi I am already here.`
   - ○ After program execution "`E:\\Files\\books.txt`" contains the following line of text in it.
     `Hi I am already here.Hello I have added to you.`

Solution:

```
var out=new FileOutputStream("E:\\Files\\books.txt",false);
```

In the above statement, false indicates that you cannot append new data to existing data in a file.
only new data added to file by erasing existing data in a file.

8. Consider the code given below.
Assume that the file "`E:\\Files\\library.txt`" contains the following lines of text in it.

```
A library is a collection of books.
Library provides hard copies of documents.
Library provides digital access to materials.
```

```
import java.io.*;
import java.util.Scanner;
public class Example {
    public static void main(String[] args) {
        try {
            var in=new FileInputStream("E:\\Files\\library.txt");
            var scanner=new Scanner(in);
            System.out.println("Data from file:");
            System.out.println(scanner.nextLine());
            System.out.println(scanner.next());
            System.out.println(scanner.nextLine());
        }
        catch (FileNotFoundException e) {
            System.out.println("File does not exist.");
        }
        catch (IOException e) {
            System.out.println("Error in writing a file.");
        }
    }
}
```

Choose the correct option regarding the code.

- ◯ Compilation error.
- ◯ Program terminates abnormally due to unhandled exception(s).
- ✓ This program generates the output:
```
Data from file:
A library is a collection of books.
Library
 provides hard copies of documents.
```
- ◯ This program generates the output:
```
Data from file:
A library is a collection of books.
A
Library provides digital access to materials.
```

Solution: Assume books.txt contains the following lines of text in it.

```
A library is a collection of books.
Library provides hard copies of documents.
Library provides digital access to materials.
```

System.out.println(scanner.nextLine());
The above statement will read the first line.
System.out.println(scanner.next());
The above statement will read only one word from the second line.
System.out.println(scanner.nextLine());
The above statement will read reaming portion of the second line.

9. Consider the following Java code and choose the correct option.

```
import java.io.*;
public class X implements Serializable{
    String str="Moon";
}
public class Y extends X{
    transient String str2="Sun";
}
public class Test{
    public static void main(String[] args) throws Exception{
        ObjectOutputStream os = new ObjectOutputStream(new FileOutputStream
                                ("File.ser"));
        os.writeObject(new Y());
        FileInputStream fis=new FileInputStream("File.ser");
        ObjectInputStream ois=new ObjectInputStream(fis);
        Y e=(Y)ois.readObject();
        System.out.print(e.str+"\n"+e.str2);
    }
}
```

- ◯ This program throws compile time error because `Y` does not implement `Serializable`.
- ◯ This program compiles successfully but throws `NotSerializableException` at runtime.
- ◯ This program generates output:
  Moon
  Sun
- ✓ This program generates output:
  Moon
  null

Solution: The serializable nature is inheritable. Thus, eventhough a child class does not implements `Serializable`, the child class objects by default gets serialized if the parent class implements `Serializable`.

10. Consider the following Java code and choose the correct option.

```
import java.io.*;
public class Mail implements Serializable{
    String user="Moon@mail.sun";
    transient int pass=1234;
    private void writeObject(ObjectOutputStream oos) throws Exception{
        oos.defaultWriteObject();
        int encrypt=(pass*100)+10;
        oos.writeObject(encrypt);
    }
    private void readObject(ObjectInputStream ois) throws Exception{
        ois.defaultReadObject();
        int decrypt=(int)ois.readObject();
        pass=(decrypt/100);
    }
}
public class Test{
    public static void main(String[] args) throws Exception{
        FileOutputStream fos=new FileOutputStream("File.ser");
        ObjectOutputStream oos=new ObjectOutputStream(fos);
        oos.writeObject(new Mail());
        FileInputStream fis=new FileInputStream("File.ser");
        ObjectInputStream ois=new ObjectInputStream(fis);
        Mail m=(Mail)ois.readObject();
        System.out.println(m.user+"\n"+m.pass);
    }
}
```

- [✓ This program generates output:
  Moon@mail.sun
  1234]
- ◯ This program generates output:
  Moon@mail.sun
  1244
- ◯ This program generates output:
  Moon@mail.sun
  0
- ◯ This program generates output:
  Moon@mail.sun
  null