CSE 1007
Java programming

Theory
Digital Assessment 2

TOPIC: Exception handling, generic classes, serialisation and de-serialisation

---

Name: Makesh Srinivasan
Registration number: 19BCE1717
Slot: A1 + TA1
Date of submission: 9 May 2021 Sunday
Faculty: Prof. Pradeep K

Questions:

1) Kumar is working in a shopping Mall and is responsible for collecting the vehicle parking charges throughout the day as shown in the following table based on the type of vehicle and parking time duration. Help Kumar by writing a java program to find the parking amount for each vehicle and total amount collected for the entire day. The program requires Kumar to enter the following input for each vehicle.

- Type of vehicle (C/T) as a single character,
- In-time in 24 hours format(hh mm) Ex 15 30
- Out-time in 24 hours format(hh mm) Ex 20 45

Sometimes, Kumar may enter wrong character as the type of vehicle, wrong In-time and Out-time (he misinterprets and type in-time as out-time and vice versa) during the peak hours. Create and Raise an exception in such cases and allow him to continue entering the input after catching the exception.

Type of vehicle (Character)	Upto 3 hours	For each hour after 3 hours
Car (C)	100	30
Two wheeler(T)	40	10

2) Read the Aadhar number and Mobile Number of an employee. If the Aadhar number does not contain exactly 12 characters or if the Mobile Number does not contain exactly 10 characters, throw an IllegalArgumentException. If the Mobile Number contains any character other than a digit, raise a NumberFormatException. If the Aadhar number contains any character other than digits and alphabets, throw a NoSuchElementException. If they are valid, print the message ‘valid’ else ‘invalid’. Write a java program for the above scenario with an appropriate exceptions.

3) Write a java program for Generic classes and methods.

4) Write a java program for Serialisation and deserialisation of objects

---

Q1) 1) Kumar is working in a shopping Mall and is responsible for collecting the vehicle parking charges throughout the day as shown in the following table based on the type of vehicle and parking time duration. Help Kumar by writing a java program to find the parking amount for each vehicle and total amount collected for the entire day. The program requires Kumar to enter the following input for each vehicle.

- Type of vehicle (C/T) as a single character,
- In-time in 24 hours format(hh mm) Ex 15 30
- Out-time in 24 hours format(hh mm) Ex 20 45

Sometimes, Kumar may enter wrong character as the type of vehicle, wrong In-time and Out-time (he misinterprets and type in-time as out-time and vice versa) during the peak hours. Create and Raise an exception in such cases and allow him to continue entering the input after catching the exception.

Type of vehicle (Character)	Upto 3 hours	For each hour after 3 hours
Car (C)	100	30
Two wheeler(T)	40	10

Input format	Output format
First enter n (number of cars/two-wheelers) For each vehicle, enter the type, in-time and out-time when prompted in the same order. type can be “C”,”c”,”T” or “t”; in and out time are in the format “hh mm”	Prints the elapsed time followed by the bill generated
Sample input:	**Expected output:**
n = 1 vehicle type = t in-time = 05 30 out-time = 08 45	Time elapsed: 195 Bill: 50

Q1	

```java
import java.util.Scanner;

class InvalidOutTIme extends Exception{
    InvalidOutTIme(String s){
        super(s);
    }
}
class InvalidInAndOutTime extends Exception{
    InvalidInAndOutTime(String s){
        super(s);
    }
}
class Price{
    static float car_base = 100;
    static float car_hourly = 30;
    static float two_wheeler_base = 40;
    static float two_wheeler_hourly = 10;
    public static float get_base(char c){
        if(c == 'C' || c == 'c'){
            return car_base;
        } else {
            return two_wheeler_base;
        }
    }
    public static float get_hourly_price(char c){
        if(c == 'C' || c == 'c'){
            return car_hourly;
        } else {
            return two_wheeler_hourly;
        }
    }
}
```

Q1	

```
class Vehicle{
    char vehicle_type;
    Hour_glass in_time;
    Hour_glass out_time;
    int total_time;
    Vehicle(char vehicle_type, Hour_glass in_time, Hour_glass out_time){
        this.vehicle_type = vehicle_type;
        this.in_time = in_time;
        this.out_time = out_time;
        total_time = out_time.elapsedTime(in_time);
    }
    public float get_bill(){
        float price = 0;
        float buffer_time = total_time;
        Double x = Math.ceil(buffer_time/60);
        if(x > 3){
            x -= 3;
            price += Price.get_base(vehicle_type);
            price += x * Price.get_hourly_price(vehicle_type);
        } else {
            if(total_time != 0){
                price += Price.get_base(vehicle_type);
            } else {
                price = 0;
            }
        }
        return price;
    }
}
class Hour_glass{
    String time;
    int hour;
    int minute;
    Hour_glass(String time){
        this.time = time;
        hour = Integer.parseInt(time.substring(0, 2));
        minute = Integer.parseInt(time.substring(3));
    }
    public int isGreaterThan(Hour_glass t2){…
    public int elapsedTime(Hour_glass t2){
        int elapsed_time = 0;
        if(this.isGreaterThan(t2) == 1){
            int x = hour - t2.hour;
            int y = minute - t2.minute;
            x = x * 60;
            if(y > 0){
                x -= 60;
                x = x + 60-(y*-1);
            } else {
                x = x + y;
            }
            elapsed_time = x;
        }
        System.out.println("Elapsed time: "+ elapsed_time + " minutes");
        return elapsed_time;
    }
}
```

Q1	

```java
class da2q1{
    public static void validate_out_time(Hour_glass out_time, Hour_glass in_time) throws InvalidInAndOutTime, InvalidOutTIme{
        if(out_time.isGreaterThan(in_time) < 0){
            throw new InvalidOutTIme("Incorrect Out-time. Perhaps, you have entered IN-time instead of OUT-time?");
        } else if(out_time.isGreaterThan(in_time) == 0){
            throw new InvalidInAndOutTime("IN and OUT time are same!");
        }
    }
    public static void drawline(String symbol){
        for(int i = 0; i < 75; i++){
            System.out.print(symbol);
        }
        System.out.println("");
    }
    Run | Debug
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        drawline("*");
        System.out.print("Enter the number of vehicles: ");
        int n = input.nextInt();
        for (int i = 0; i < n; i++) {
            System.out.println("\nVehicle "+ (i+1)+ ") ");
            System.out.print("Enter the type of vehicle: ");
            char type = input.next().charAt(0);
            System.out.print("Enter the in-time in 24Hr format (hh mm): ");
            input.nextLine();
            Hour_glass in_time = new Hour_glass(input.nextLine());
            System.out.print("Enter the out-time in 24Hr format (hh mm): ");
            Hour_glass out_time = new Hour_glass(input.nextLine());
            drawline("_");
            try{
                validate_out_time(out_time, in_time);
            } catch (InvalidOutTIme timeswap){
                System.out.println(timeswap.getMessage());
                System.out.print("Do you want to swap IN and OUT time? (y/n): ");
                char option = input.next().charAt(0);
                if(option == 'y'){
                    Hour_glass temp = out_time;
                    out_time = in_time;
                    in_time = temp;
                } else {
                    System.out.println("You have entered 'n' (no) indicating error in your entry. Entry cancelled");
                    i--;
                    drawline(" ");
                    continue;
                }
            } catch (InvalidInAndOutTime samefields){
                System.out.println(samefields.getMessage());
                System.out.print("Vehicle entered and exited at the same time. Do you want to continue processing? (y/n): ");
                char option = input.next().charAt(0);
                if(option == 'n'){
                    System.out.println("You have entered 'n' (no) indicating error in your entry. Entry cancelled");
                    i--;
                    drawline(" ");
                    continue;
                }
            }
            Vehicle vehicle = new Vehicle(type, in_time, out_time);
            float bill = vehicle.get_bill();
            System.out.println("Bill: " + bill);
            drawline("*");
        }
        input.close();
    }
}
```

Q1	
	Output:

```
********************************************************************************
Enter the number of vehicles: 5

Vehicle 1)
Enter the type of vehicle: c
Enter the in-time in 24Hr format (hh mm): 05 30
Enter the out-time in 24Hr format (hh mm): 08 30
________________________________________________________________________________
Elapsed time: 180 minutes
Bill: 100.0
********************************************************************************

Vehicle 2)
Enter the type of vehicle: t
Enter the in-time in 24Hr format (hh mm): 05 30
Enter the out-time in 24Hr format (hh mm): 08 45
________________________________________________________________________________
Elapsed time: 195 minutes
Bill: 50.0
********************************************************************************

Vehicle 3)
Enter the type of vehicle: t
Enter the in-time in 24Hr format (hh mm): 12 30
Enter the out-time in 24Hr format (hh mm): 01 30
________________________________________________________________________________
Incorrect Out-time. Perhaps, you have entered IN-time instead of OUT-time?
Do you want to swap IN and OUT time? (y/n): y
Elapsed time: 660 minutes
Bill: 120.0
********************************************************************************

Vehicle 4)
Enter the type of vehicle: t
Enter the in-time in 24Hr format (hh mm): 12 30
Enter the out-time in 24Hr format (hh mm): 01 30
________________________________________________________________________________
Incorrect Out-time. Perhaps, you have entered IN-time instead of OUT-time?
Do you want to swap IN and OUT time? (y/n): n
You have entered 'n' (no) indicating error in your entry. Entry cancelled


Vehicle 4)
Enter the type of vehicle: c
Enter the in-time in 24Hr format (hh mm): 12 30
Enter the out-time in 24Hr format (hh mm): 12 30
________________________________________________________________________________
IN and OUT time are same!
Vehicle entered and exited at the same time. Do you want to continue processing? (y/n): y
Elapsed time: 0 minutes
Bill: 0.0
********************************************************************************

Vehicle 5)
Enter the type of vehicle: c
Enter the in-time in 24Hr format (hh mm): 12 30
Enter the out-time in 24Hr format (hh mm): 12 30
________________________________________________________________________________
IN and OUT time are same!
Vehicle entered and exited at the same time. Do you want to continue processing? (y/n): n
You have entered 'n' (no) indicating error in your entry. Entry cancelled
```

Q2) Read the Aadhar number and Mobile Number of an employee. If the Aadhar number does not contain exactly 12 characters or if the Mobile Number does not contain exactly 10 characters, throw an IllegalArgumentException. If the Mobile Number contains any character other than a digit, raise a NumberFormatException. If the Aadhar number contains any character other than digits and alphabets, throw a NoSuchElementException. If they are valid, print the message 'valid' else 'invalid'. Write a java program for the above scenario with an appropriate exceptions.

Input format	**Output format**
Read n (number of employees) Enter phone number followed by the Aadhar number when prompted	Prints if the entires are valid or invalid. If invalid, before printing invalid, it prints the errors
Sample input:	**Expected output:**
n = 1 99191hajko 1234asdf	IllegalArgumentException: Incorrect number of characters NumberFormatException: There are non-digits in the phone number Invalid

Q2

```java
import java.util.Scanner;

class NoSuchElementException extends Exception{
    NoSuchElementException(String s){
        super(s);
    }
}
class Employee{
    String phone;
    String aadhar;
    boolean valid = true;
    Employee(String phone, String aadhar){
        this.phone = phone;
        this.aadhar = aadhar;
    }
    public void validate_3() throws NoSuchElementException{
        char[] aadhar_array = aadhar.toCharArray();
        for (char c : aadhar_array) {
            if(!(Character.isDigit(c) || Character.isAlphabetic(c))){
                valid = false;
                throw new NoSuchElementException("NoSuchElementException: There are invalid characters in your aadhar");
            }
        }
    }
    public void validate_1(){
        if(aadhar.length() != 12 || phone.length() != 10){
            valid = false;
            throw new IllegalArgumentException("IllegalArgumentException: Incorrect number of characters");
        }
    }
    public void validate_2(){
        char[] phone_array = phone.toCharArray();
        for (char c : phone_array) {
            if(!Character.isDigit(c)){
                valid = false;
                throw new NumberFormatException("NumberFormatException: There are non-digits in the phone number");
            }
        }
    }
}
```

Q2

```
class da2q2{
    public static void drawline(String symbol){
        for(int i = 0; i < 50; i++){
            System.out.print(symbol);
        }
        System.out.println("");
    }
    Run | Debug
    public static void main(String args[]){
        System.out.println("");
        drawline("*");
        int n = 0;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter the number of employees: ");
        n = input.nextInt();
        Employee[] emploee = new Employee[n];
        for (int i = 0; i < emploee.length; i++) {
            String phone, aadhar;
            System.out.println("\nEmployee " + (i+1) + ")");
            System.out.print("Enter phone number: ");
            phone = input.next();
            System.out.print("Enter aadhar number: ");
            aadhar = input.next();
            emploee[i] = new Employee(phone, aadhar);
        }
        drawline("_");
        System.out.println("\nVALIDATION:\n");
        for (int i = 0; i < emploee.length; i++) {
            System.out.println("Employee " + (i+1) +": ");
            try{
                emploee[i].validate_1();
            } catch(Exception e){
                System.out.println(e.getMessage());
            }
            try{
                emploee[i].validate_2();
            } catch(Exception e){
                System.out.println(e.getMessage());
            }
            try{
                emploee[i].validate_3();
            } catch(Exception e){
                System.out.println(e.getMessage());
            }
            if(emploee[i].valid){
                System.out.println("Valid");
            } else {
                System.out.println("Invalid");
            }
            drawline("_");
        }
        input.close();
    }
}
```

Q2	
	Output:

```
**************************************************
Enter the number of employees: 5

Employee 1)
Enter phone number: 12341l3
Enter aadhar number: @no8g8g

Employee 2)
Enter phone number: 12345678900
Enter aadhar number: 12123lqno8g8g

Employee 3)
Enter phone number: 123456789w
Enter aadhar number: 12123lno8g8g

Employee 4)
Enter phone number: 1234567890
Enter aadhar number: 12123@no8g8g

Employee 5)
Enter phone number: 1234567890
Enter aadhar number: 12123qno8g8g
--------------------------------------------------

VALIDATION:

Employee 1:
IllegalArgumentException: Incorrect number of characters
NumberFormatException: There are non-digits in the phone number
NoSuchElementException: There are invalid characters in your aadhar
Invalid

--------------------------------------------------
Employee 2:
IllegalArgumentException: Incorrect number of characters
Invalid

--------------------------------------------------
Employee 3:
NumberFormatException: There are non-digits in the phone number
Invalid

--------------------------------------------------
Employee 4:
NoSuchElementException: There are invalid characters in your aadhar
Invalid

--------------------------------------------------
Employee 5:
Valid

--------------------------------------------------
Srinivasans-MacBook-Pro:DA 2 makeshsrinivasan$
```

Q3) Write a java program for Generic classes and methods.

In this question, I have implemented generic class and method in a logic calculator. It performs AND, OR, NAND, NOR, XNOR, XOR and NOT operations and the output image shows the working of this system more clearly.

Input format	Output format
Enter operand 1 and 2 as INT or DOUBLE Enter one of the Logical operator mentioned in the menu	Prints the result as a binary string
Sample input:	**Expected output:**
Enter operand 1: 9.25 Enter operand 2: 8.5 Enter Logical operator: OR	1001.11

Q3	

da2q3.java > ...

```java
import java.util.Scanner;
class da2q3 {

    public static void drawline(String symbol){
        for(int i = 0; i < 50; i++){
            System.out.print(symbol);
        }
        System.out.println("");
    }
    Run | Debug
    public static void main(String[] args) {
        int running = 1;
        Scanner input = new Scanner(System.in);
        drawline("*");
        System.out.println("Available logical operations in this calculator");
        System.out.println("AND, OR, NOT, XOR, NAND, NOR, XNOR");
        drawline("_");
        while(running == 1){
            System.out.print("Enter operand 1: ");
            String op1 = input.next();

            System.out.print("Enter operand 2: ");
            String op2 = input.next();

            System.out.print("Enter Logical operator: ");
            String operator = input.next();

            if(op1.contains(".") || op2.contains(".")){
                double x = Double.parseDouble(op1);
                double y = Double.parseDouble(op2);
                Calculator<Double, String> ob = new Calculator<>(x,y,operator);
                System.out.print("Calculating "+ob.get_operator()+" operation between "+op1+" (");
                System.out.println(ob.get_op1_type(x)+") and "+op2+" ("+ob.get_op2_type(y)+")");
                System.out.println("\nGeneric calculator returns "+ob.get_operator()+" operation: ");
                ob.calculate();
            } else {
                int x = Integer.parseInt(op1);
                int y = Integer.parseInt(op2);
                Calculator<Integer, String> ob = new Calculator<>(x,y,operator);
                System.out.print("Calculating "+ob.get_operator()+" operation between "+op1+" (");
                System.out.println(ob.get_op1_type(x)+") and "+op2+" ("+ob.get_op2_type(y)+")");
                System.out.println("\nGeneric calculator returns "+ob.get_operator()+" operation: ");
                ob.calculate();
            }
            System.out.println("");
            drawline("_");
            System.out.print("\nDo you want to calculate more? (y/n)");
            char confirmed = input.next().charAt(0);
            if(confirmed == 'n'){
                running = 0;
            }
        }
        input.close();
    }
}
```

Q3	

```
class Calculator<T extends Number, U> {
    private T operand_1;
    private T operand_2;
    private U operator;
    char[] a;
    char[] b;
    char[] result;
    Calculator(T op1, T op2, U operator) {
        this.operand_1 = op1;
        this.operand_2 = op2;
        this.operator = operator;
        if(operand_1.getClass().getName() == "java.lang.Integer"){
            a = Integer.toBinaryString(operand_1.intValue()).toCharArray();
            b = Integer.toBinaryString(operand_2.intValue()).toCharArray();
        } else {
            a = toBinary(operand_1.doubleValue(), 4).toCharArray();
            b = toBinary(operand_2.doubleValue(), 4).toCharArray();
        }
        result = new char[a.length];
    }
    Calculator(T op, U operator) {
        this.operand_1 = op;
        this.operator = operator;
        if(operand_1.getClass().getName() == "java.lang.Integer"){
            a = Integer.toBinaryString(operand_1.intValue()).toCharArray();
        } else {
            a = toBinary(operand_1.doubleValue(), 4).toCharArray();
        }
        result = new char[a.length];
    }
    public <V> String get_op1_type(T op_1) {
        return op_1.getClass().getName();
    }
    public <V> String get_op2_type(T op_2) {
        return op_2.getClass().getName();
    }
    public U get_operator(){
        return operator;
    }
```

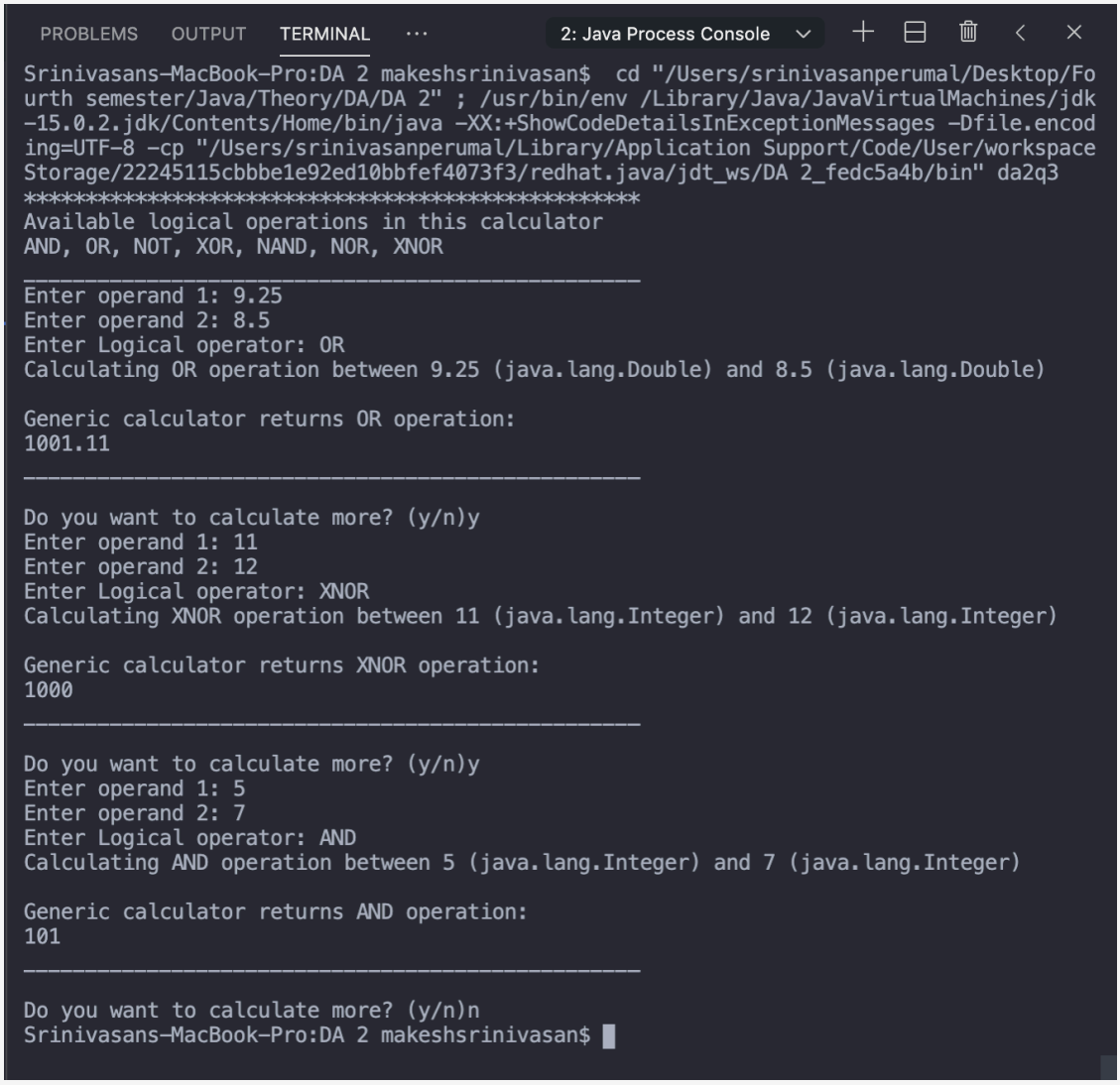
Q3	

```
95      public void calculate(){
96          char[] res = new char[32];
97          if(operator.toString().equals("AND")){
98              res = AND(a, b);
99          } else if(operator.toString().equals("OR")){
100             res = OR(a, b);
101         } else if(operator.toString().equals("NOT")){
102             res = NOT(a);
103         } else if(operator.toString().equals("XOR")){
104             res = XOR(a, b);
105         } else if(operator.toString().equals("XNOR")){
106             res = NOT(XOR(a, b));
107         } else if(operator.toString().equals("NOR")){
108             res = NOT(OR(a, b));
109         } else if(operator.toString().equals("NAND")){
110             res = NOT(AND(a, b));
111         }
112         display(res);
113     }
114     private void display(char[] array){
115         for (int i = 0; i < array.length; i++) {
116             System.out.print(array[i]);
117         }
118     }
119     private char[] AND(char[] a, char[] b) {
120         for (int i = 0; i < b.length; i++) {
121             if(a[i] == b[i] && a[i] == '1'){
122                 result[i] = '1';
123             } else if(a[i] == '.'){
124                 result[i] = '.';
125             } else {
126                 result[i] = '0';
127             }
128         }
129         return result;
130     }
131
132     private char[] OR(char[] a, char[] b) {
133         for (int i = 0; i < a.length; i++) {
134             if(a[i] == '1' || b[i] == '1'){
135                 result[i] = '1';
136             } else if(a[i] == '.'){
137                 result[i] = '.';
138             } else {
139                 result[i] = '0';
140             }
141         }
142         return result;
143     }
144
```

Q3

```
    private char[] NOT(char[] a) {
        for (int i = 0; i < a.length; i++) {
            if(a[i] == '1'){
                result[i] = '0';
            } else if (a[i] == '.'){
                result[i] = '.';
            } else {
                result[i] = '1';
            }
        }
        return result;
    }

    private char[] XOR(char[] a, char[] b) {
        for (int i = 0; i < b.length; i++) {
            if(a[i] != b[i]){
                result[i] = '1';
            } else if(a[i] == '.'){
                result[i] = '.';
            } else {
                result[i] = '0';
            }
        }
        return result;
    }

    public static String toBinary(double d, int precision) {
        long wp = (long) d;
        return wholeToBinary(wp) + '.' + fractionalToBinary(d - wp, precision);
    }

    private static String wholeToBinary(long l) {
        return Long.toBinaryString(l);
    }

    private static String fractionalToBinary(double num, int precision) {
        StringBuilder binary = new StringBuilder();
        while (num > 0 && binary.length() < precision) {
            double r = num * 2;
            if (r >= 1) {
                binary.append(1);
                num = r - 1;
            } else {
                binary.append(0);
                num = r;
            }
        }
        return binary.toString();
    }
}
```

Q3	
	Output:


PROBLEMS
OUTPUT
TERMINAL
2: Java Process Console
Srinivasans-MacBook-Pro:DA 2 makeshsrinivasan$ cd "/Users/srinivasanperumal/Desktop/Fourth semester/Java/Theory/DA/DA 2" ; /usr/bin/env /Library/Java/JavaVirtualMachines/jdk-15.0.2.jdk/Contents/Home/bin/java -XX:+ShowCodeDetailsInExceptionMessages -Dfile.encoding=UTF-8 -cp "/Users/srinivasanperumal/Library/Application Support/Code/User/workspaceStorage/22245115cbbbe1e92ed10bbfef4073f3/redhat.java/jdt_ws/DA 2_fedc5a4b/bin" da2q3
****
Available logical operations in this calculator
AND, OR, NOT, XOR, NAND, NOR, XNOR
Enter operand 1: 9.25
Enter operand 2: 8.5
Enter Logical operator: OR
Calculating OR operation between 9.25 (java.lang.Double) and 8.5 (java.lang.Double)
Generic calculator returns OR operation:
1001.11
Do you want to calculate more? (y/n)y
Enter operand 1: 11
Enter operand 2: 12
Enter Logical operator: XNOR
Calculating XNOR operation between 11 (java.lang.Integer) and 12 (java.lang.Integer)
Generic calculator returns XNOR operation:
1000
Do you want to calculate more? (y/n)y
Enter operand 1: 5
Enter operand 2: 7
Enter Logical operator: AND
Calculating AND operation between 5 (java.lang.Integer) and 7 (java.lang.Integer)
Generic calculator returns AND operation:
101
Do you want to calculate more? (y/n)n
Srinivasans-MacBook-Pro:DA 2 makeshsrinivasan$

Q4) Write a java program for Serialisation and deserialisation of objects

A student_details txt file is created to store the details of students (name, age, registration number, address and phone number). In serialisation, the student objects are serialised into an array list and saved in a .Txt file. In de-serialisation, the array list is de-serialised and converted back into student objects. If the student is graduating in 2023 or joined the UG programme in 2019, the details of the student are printed to the screen.

Serialisation

Input format	**Output format**

`Enter name`
`Enter age`
`Enter registration number`
`Enter phone number`
`Enter address`
 	
`And finally, if you want to add more objects, type y (yes) when prompted`
	NIL (a serialised Txt file is created and saved in the same directory)

Sample input:	**Expected output:**

`Enter name: Makesh`
`Enter age: 19`
`Enter registration number: 19BCE1717`
`Enter phone number: 9876098765`
`Enter address: Earth`
 	
`Do you want to calculate more? (y/n)y`
	Serialised file as .Txt in the folder

De-serialisation

Input format	**Output format**

`NIL`	`The students that graduate in 2023 or joined UG in 2019 are:`
	
	`Name`
	`Age`
	`Registration number`
	`Phone`
	`Address`

Sample input:	**Expected output:**

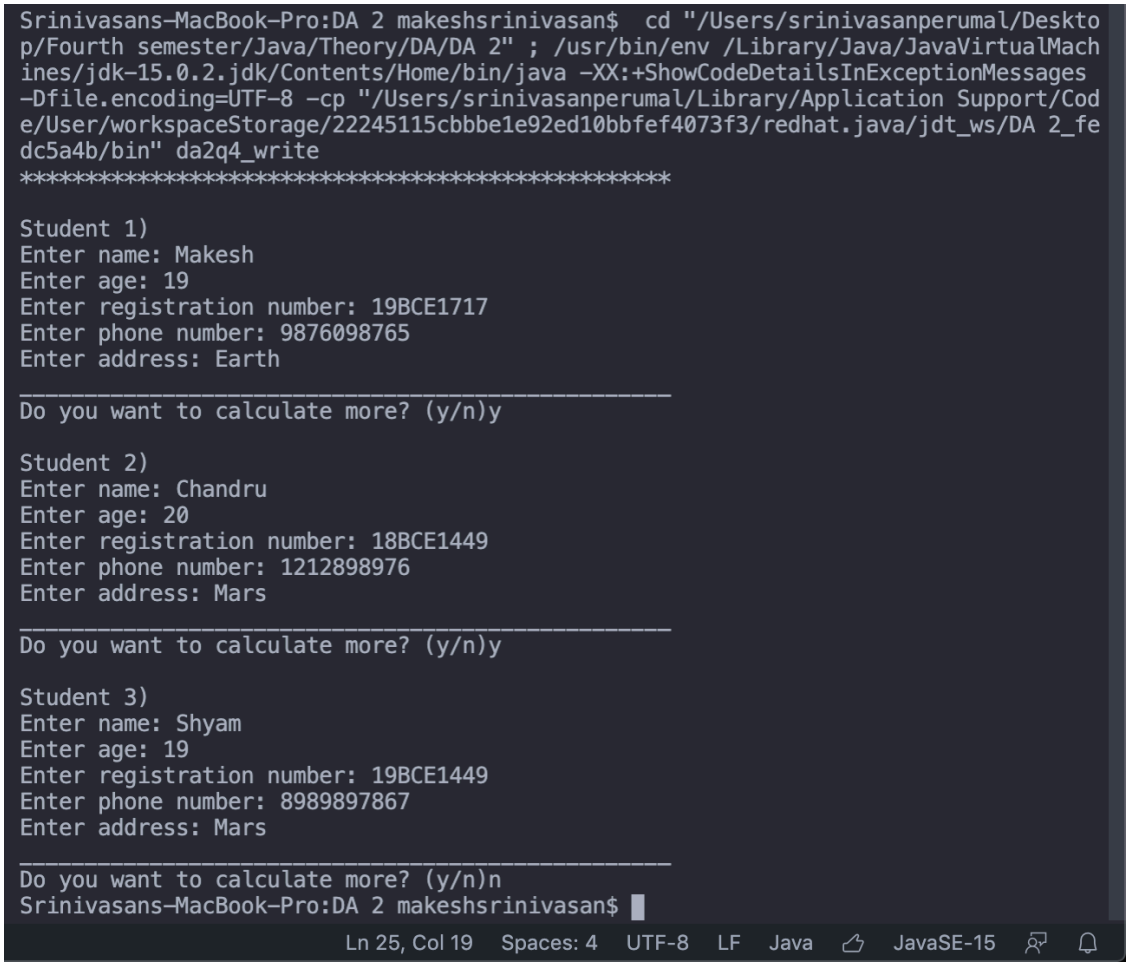
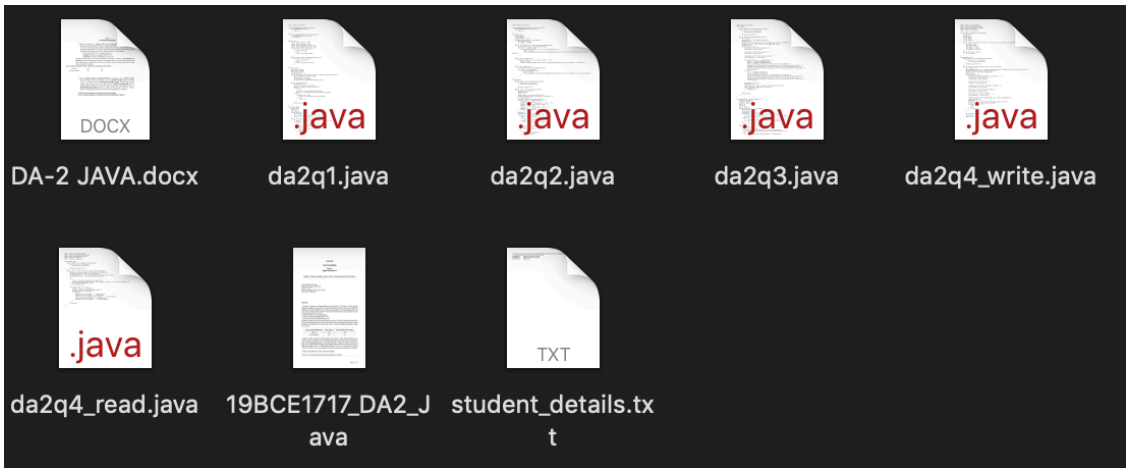
`NIL`	`The students that graduate in 2023 or joined UG in 2019 are:`
	`Name: Makesh`
	`Age: 19`
	`Registration number: 19BCE1717`
	`Phone: 9876098765`
	`Address: Earth`

Q4	Serialisation

da2q4_write.java > da2q4_write

```java
import java.io.Serializable;
import java.io.FileOutputStream;
import java.io.ObjectOutputStream;
import java.util.Scanner;
import java.util.ArrayList;

class Student implements Serializable{
    String name;
    int age;
    String regno;
    String phone;
    String address;

    public Student(String name, int age, String regno, String phone, String address) {
        this.name = name;
        this.age = age;
        this.regno = regno;
        this.phone = phone;
        this.address = address;
    }
    public String get_regno(){
        return regno;
    }
}
```

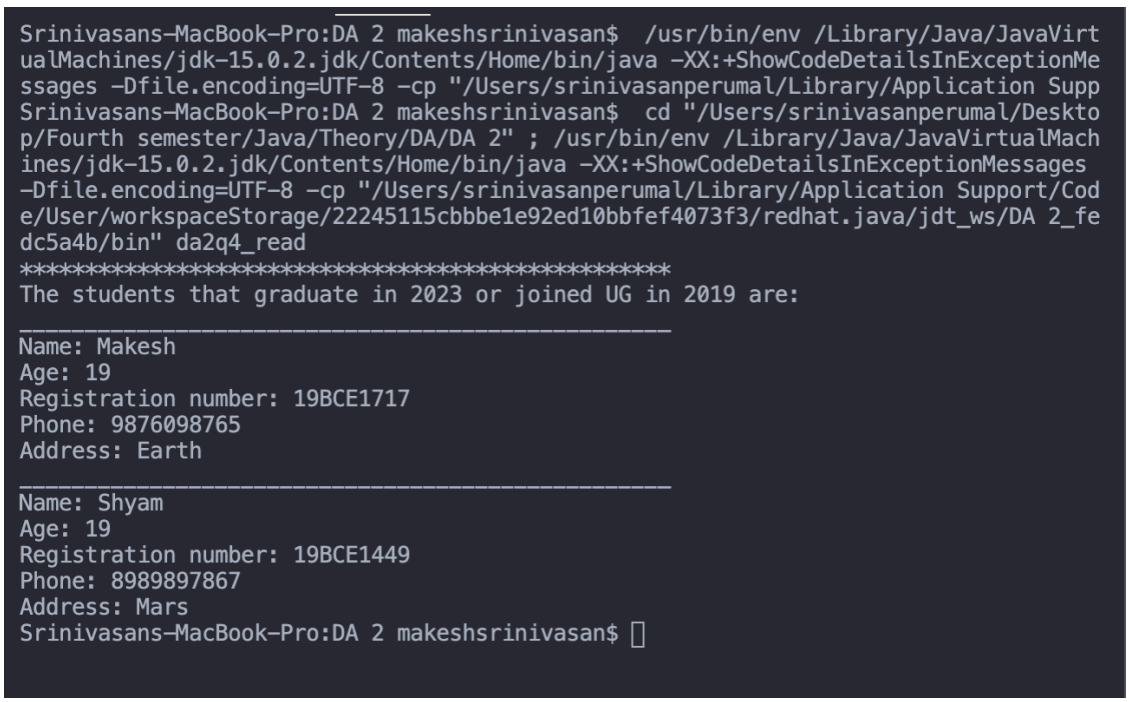
```java
}
class da2q4_write{
    public static void drawline(String symbol){
        for(int i = 0; i < 50; i++){
            System.out.print(symbol);
        }
        System.out.println("");
    }
    Run | Debug
    public static void main(String[] args) throws Exception{
        int running = 1;
        ObjectOutputStream out = new ObjectOutputStream(new FileOutputStream("student_details.txt"));
        ArrayList<Student> array = new ArrayList<>();
        Scanner input = new Scanner(System.in);
        drawline("*");
        while(running >= 1){
            System.out.println("\nStudent " + running + ") ");
            System.out.print("Enter name: ");
            String name = input.next();

            System.out.print("Enter age: ");
            int age = input.nextInt();

            System.out.print("Enter registration number: ");
            String regno = input.next();

            System.out.print("Enter phone number: ");
            String phone = input.next();

            System.out.print("Enter address: ");
            String address = input.next();

            Student student = new Student(name, age, regno, phone,address);
            array.add(student);
            drawline("_");
            System.out.print("Do you want to calculate more? (y/n)");
```

Q4	Serialisation
	Output:
	
	File in the directory:
	

Q4	De-serialisation

da2q4_read.java > da2q4_read

```java
import java.util.regex.*;
import java.util.ArrayList;

class da2q4_read{
    public static void drawline(String symbol){
        for(int i = 0; i < 50; i++){
            System.out.print(symbol);
        }
        System.out.println("");
    }
    Run | Debug
    public static void main(String[] args) throws Exception{
        Pattern p = Pattern.compile("[1][9]([A-Z]){3}[0-9]{4}");
        ArrayList<Student> array = new ArrayList<>();
        FileInputStream fin = new FileInputStream("student_details.txt");
        ObjectInputStream in = new ObjectInputStream(fin);

        try{
            array = (ArrayList<Student>)in.readObject();
            drawline("*");
            System.out.println("The students that graduate in 2023 or joined UG in 2019 are:");
        } catch (FileNotFoundException e){
            e.getStackTrace();
        }
        for (int i = 0; i < 3; i++) {
            Student student = array.get(i);
            Matcher m = p.matcher(student.get_regno());
            boolean matches = m.matches();
            if(matches){
                drawline("_");
                System.out.println("Name: " + student.name);
                System.out.println("Age: " + student.age);
                System.out.println("Registration number: " + student.regno);
                System.out.println("Phone: " + student.phone);
                System.out.println("Address: " + student.address);
            }
        }
        in.close();
    }
}

```

Q4	De-serialisation
	Output:
	
	Only Shyam and Makesh are 19BCE students and Chandru is a 18BCE student. Hence only Makesh and Shyam are mentioned here