

# JAVA-2

YUKTI SHARMA

1. Create Java classes having suitable attributes for Library management system. Use OOPs concepts in your design. Also try to use interfaces and abstract classes.

```
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.Date;

interface Student {

    void studentInfo(String name, int s_id, String city);

    void batchInfo(String batch);

}

public abstract class Ques1 implements Student {
    String name;
    int s_id;
    String city;
    String batch;

    @Override
    public void studentInfo(String n, int s, String c) {
        name = n;
        s_id = s;
        city = c;
        System.out.println("Student name is " + name + ", " +
            "\nStudent id is: " + s_id + " \nStudent belongs to: " + city);
    }

    @Override
    public void batchInfo(String b) {
        batch = b;
        System.out.println(name + " is of batch:- " + batch);
    }
}
```

```

@Override
public void batchInfo(String b) {
    batch = b;
    System.out.println(name + " is of batch:- " + batch);
}

abstract public void BookDetails(int bookid, String bookName);

class Library extends Ques1 {
    String book;
    int bookid;

    @Override
    public void BookDetails(int bookid, String bookName) {
        book = bookName;
        this.bookid = bookid;
        System.out.println("Book issued is:-\nBook id- " + bookid + "\nName:- " + book);
    }

    public void dateissued(String date2) {
        SimpleDateFormat formatter = new SimpleDateFormat( pattern: "dd/MM/yyyy");
        try {
            Date date = formatter.parse(date2);
            System.out.println("Date of issue is: " + date);
        } catch (ParseException e) {
            e.printStackTrace();
        }
    }
}

class Main {

    public static void main(String[] args) {
        Library obj = new Library();
        obj.studentInfo( n: "Yukti", s: 1121, c: "delhi");
        obj.batchInfo( b: "B.Tech-IT-2019");
        obj.BookDetails( bookid: 101, bookName: "DigitalLogic");
        obj.dateissued( date2: "31/03/2015");
    }
}

```

```
Main x
/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java
Student name is Yukti ,
Student id is: 1121
Student belongs to: delhi
Yukti is of batch:- B.Tech-IT-2019
Book issued is:-
Book id- 101
Name:- DigitalLogic
Date of issue is: Tue Mar 31 00:00:00 IST 2015

Process finished with exit code 0
```

## 2. WAP to sorting string without using string Methods?.

```
import java.util.Scanner;

public class Ques2 {
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter the string: ");
        String input1= sc.nextLine();
        String input= input1.replace( target: " ", replacement: "");
        int j = 0;
        char temp = 0;

        char[] chars = input.toCharArray();

        for (int i = 0; i < chars.length; i++) {
            for (j = 0; j < chars.length; j++) {
                if (chars[j] > chars[i]) {
                    temp = chars[i];
                    chars[i] = chars[j];
                    chars[j] = temp;
                }
            }
        }

        for (int k = 0; k < chars.length; k++) {
            System.out.print(chars[k]);
        }
    }
}
```

```
Ques8 x Ques2 x
/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/
Enter the string:
deabylkfghy
abdefghklly
Process finished with exit code 0
```

### 3. WAP to produce NoClassDefFoundError and ClassNotFoundException exception.

```
public class Ques3 {
    public static void main(String[] args) {
        Second b = new Second();
    }
}

class Second{
    //this is another class
}
```

```
n: Ques8 x Ques3 x
/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
Exception in thread "main" java.lang.NoClassDefFoundError: Second
    at Ques3.main(Ques3.java:3)
Caused by: java.lang.ClassNotFoundException: Second
    at java.net.URLClassLoader.findClass(URLClassLoader.java:382)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:424)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:349)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:357)
    ... 1 more

Process finished with exit code 1
```

```

public class Ques3b {
    public static void main(String[] args) {
        try
        {
            Class.forName("oracle.jdbc.driver.OracleDriver");
        } catch (ClassNotFoundException e)
        {
            e.printStackTrace();
        }
    }
}

```

```

/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java ...
java.lang.ClassNotFoundException: oracle.jdbc.driver.OracleDriver
    at java.net.URLClassLoader.findClass(URLClassLoader.java:382)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:424)
    at sun.misc.Launcher$AppClassLoader.loadClass(Launcher.java:349)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:357)
    at java.lang.Class.forName0(Native Method)
    at java.lang.Class.forName(Class.java:264)
    at Ques3b.main(Ques3b.java:7)

Process finished with exit code 0

```

**4. WAP to create singleton class.**

```

class Singleton
{
    private static Singleton instance = null;
    public String s;
    private Singleton() { s = "Singleton class is running"; }

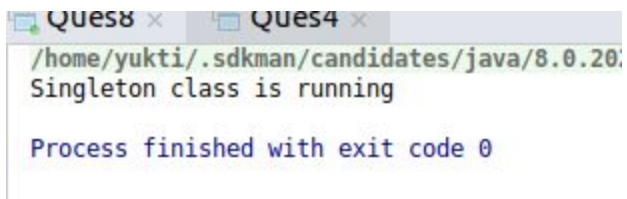
    public static Singleton getInstance()
    {
        if (instance == null)
            instance = new Singleton();

        return instance;
    }
}

public class Ques4
{
    public static void main(String args[])
    {
        Singleton x = Singleton.getInstance();
        System.out.println(x.s);
    }
}

```

output-



```

Ques8 x  Ques4 x
/home/yukti/.sdkman/candidates/java/8.0.20:
Singleton class is running

Process finished with exit code 0

```

**5. WAP to show object cloning in java using cloneable and copy constructor both.**



```

public class Ques5 {
    int a;
    int b;

    public Ques5(int a, int b){
        this.a=a;
        this.b=b;
    }

    public Ques5(Ques5 ref){
        a=ref.a;
        b=ref.b;
    }

    public static void main(String[] args) throws CloneNotSupportedException {
        A obj = new A();
        obj.i=1;
        obj.j=3;

        A obj1=(A)obj.clone();
        System.out.println("through cloneable\n "+obj.i+ " "+ obj1.i+"\n "+obj.j+ " "+ obj1.j);

        Ques5 o = new Ques5( a: 4, b: 6);
        Ques5 o1= new Ques5(o);
        System.out.println("\n through copy constructor\n"+o.a+" "+o.b);
        System.out.println(o1.a+" "+o1.b);
    }
}

```

```

class A implements Cloneable {
    int i;
    int j;
    @Override
    public Object clone() throws CloneNotSupportedException
    {
        return super.clone();
    }
}

```

Quest

```

/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/
through cloneable
1 1
3 3

through copy constructor
4 6
4 6

Process finished with exit code 0

```

## 6. WAP showing try, multi-catch and finally blocks.

```
public class Ques6 {  
    public static void main(String args[]){  
        try{  
            int array[]=new int[2];  
            array[4]=60/0;  
  
            System.out.println("First print statement in try block");  
        }  
  
        catch(ArithmeticException e){  
            System.out.println("Warning: ArithmeticException");  
        }  
  
        catch(ArrayIndexOutOfBoundsException e){  
            System.out.println("Warning: ArrayIndexOutOfBoundsException");  
        }  
  
        catch(Exception e){  
            System.out.println("Warning: Some Other exception");  
        }  
  
        finally {  
            System.out.println("execution of finally block");  
        }  
  
        System.out.println("Out of try-catch block...");  
    }  
}
```

output-

```
Ques8 x Ques6 x  
/home/yukti/.sdkman/candidates/java/8.0.202-amz  
Warning: ArithmeticException  
execution of finally block  
Out of try-catch block...  
  
Process finished with exit code 0
```

## 7. WAP to convert seconds into days, hours, minutes and seconds.



```

import java.util.Scanner;
public class Ques7 {

    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.println("Enter input seconds");
        long seconds= sc.nextLong();
        System.out.println(seconds+" seconds equals to:==>\n");
        ConvertSectoDay(seconds);
    }

    static void ConvertSectoDay(long n)
    {
        long day = n / (24 * 3600);

        n = n % (24 * 3600);
        long hour = n / 3600;

        n= n% 3600;
        long minutes = n / 60 ;

        n =n% 60;
        long seconds = n;

        System.out.println(day + " " + "days " + hour
            + " " + "hours " + minutes + " "
            + "minutes " + seconds + " "
            + "seconds ");
    }
}

```

## Output-

```

/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java ..
Enter input seconds
878896546
878896546 seconds equals to:==>

10172 days 9 hours 55 minutes 46 seconds

Process finished with exit code 0

```

8. WAP to read words from the keyboard until the word done is entered. For each word except done, report whether its first character is equal to its last character. For the required loop, use a  
a)while statement

```

import java.util.Scanner;

public class Ques8 {
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.println("enter word");
        String word = sc.next();
        while(!word.equals("done"))
        {
            if(word.charAt(0)==(word.charAt(word.length()-1)))
            {
                System.out.println("first and last character are equal");
            }

            else {
                System.out.println("first and last character are not equal");
            }
            System.out.println("enter other word");
            word= sc.next();
        }
    }
}

```

output-

```

/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java
enter word
helloh
first and last character are equal
enter other word
hey
first and last character are not equal
enter other word
lyruhl
first and last character are equal
enter other word
done

Process finished with exit code 0

```

**b)do-while statement**

```

import java.util.Scanner;

public class Ques8b {
    public static void main(String[] args) {
        Scanner sc= new Scanner(System.in);
        System.out.println("enter word");
        String word = sc.next();
        do
        {
            if(word.charAt(0)==(word.charAt(word.length()-1)))
            {
                System.out.println("first and last character are equal");
            }

            else {
                System.out.println("first and last character are not equal");
            }
            System.out.println("enter other word");
            word= sc.next();
        } while(!word.equals("done"));
    }
}

```

```

/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java
enter word
helloh
first and last character are equal
enter other word
hey
first and last character are not equal
enter other word
lyruh1
first and last character are equal
enter other word
done

Process finished with exit code 0

```

**9. Design classes having attributes for furniture where there are wooden chairs and tables, metal chairs and tables. There are stress and fire tests for each products.**

Ques13.java × Linked.java × Ques11.java × Ques9.java × Ques8.ja

```
import java.util.Scanner;

interface Furniture {
    public void stressTest();
    public void fireTest();
}

abstract class Chair implements Furniture {
    public abstract String chairType();
}

abstract class Table implements Furniture {
    public abstract String tableType();
}

class MetalChair extends Chair {...}

class MetalTable extends Table {...}

class WoodenTable extends Table {...}

class WoodenChair extends Chair {...}
```

```

public class Ques9 {
    public static void main(String[] args){
        Table table = null;
        Chair chair=null;
        System.out.println("enter either metallic or wooden");
        Scanner input = new Scanner(System.in);
        String str = input.next();
        if(str.equals("wooden")){
            System.out.println("Enter chair or table:- ");
            String entry= input.next();
            if(entry.equals("chair")) {
                chair = new WoodenChair();
                System.out.println(chair.chairType());
                chair.stressTest();
                chair.fireTest();
            }
            else if(entry.equals("table")){
                table = new WoodenTable();
                System.out.println(table.tableType());
                table.stressTest();
                table.fireTest();
            }
            else {
                System.out.println("enter either chair or table");
            }
        }
    }
}

```

```

    }
    if(str.equals("metallic")){
        System.out.println("Enter chair or table:- ");
        String entry= input.next();
        if(entry.equals("chair")) {
            chair = new MetalChair();
            System.out.println(chair.chairType());
            chair.stressTest();
            chair.fireTest();
        }
        else if(entry.equals("table")){
            table = new MetalTable();
            System.out.println(table.tableType());
            table.stressTest();
            table.fireTest();
        }
        else {
            System.out.println("enter either chair or table");
        }
    }
}

```

```
Ques9 x Ques9 x
/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/java .
enter either metallic or wooden
wooden
Enter chair or table:-
table
This is a wooden Table
Failed Stress Test
Failed Fire Test

Process finished with exit code 0
|
```

**10. Design classes having attributes and method(only skeleton) for a coffee shop. There are three different actors in our scenario and i have listed the different actions they do also below**

**\* Customer**

- Pays the cash to the cashier and places his order, get a token number back
- Waits for the intimation that order for his token is ready
- Upon intimation/notification he collects the coffee and enjoys his drink

( Assumption: Customer waits till the coffee is done, he wont timeout and cancel the order. Customer always likes the drink served. Exceptions like he not liking his coffee, he getting wrong coffee are not considered to keep the design simple.)

**\* Cashier**

- Takes an order and payment from the customer
- Upon payment, creates an order and places it into the order queue
- Intimates the customer that he has to wait for his token and gives him his token

( Assumption: Token returned to the customer is the order id. Order queue is unlimited. With a simple modification, we can design for a limited queue size)

**\* Barista**

- Gets the next order from the queue



- Prepares the coffee
- Places the coffee in the completed order queue
- Places a notification that order for token is ready

```
import java.util.Scanner;

public class Ques10 {

    private static int token_number=1001;
    private double order_amount;
    private boolean token_status;
    private boolean payment_status;
    private int[] order_queue;
    private int[] ready_queue;
    private final int MAX_ORDER;
    private Scanner In;
    private static int o_count;

    public Ques10() {
        MAX_ORDER=10;
        token_status=false;
        payment_status=false;
        order_queue=new int[MAX_ORDER];
        ready_queue=new int[MAX_ORDER];
        In=new Scanner(System.in);
    }
}
```

```
public int customer_requestService() {
    System.out.println("[Pay Bill] Enter amount: ");
    order_amount=In.nextLong();
    if(order_amount>=0) {
        order_queue[o_count]=token_number;
        o_count++;
        token_number++;
        System.out.println("Your Order id is "+token_number+" , please wait.");

        return token_number;
    } else {
        System.out.println("Amount not paid, try again.");
        return 0;
    }
}
```

```
public void cashier_processOrder() {
    if(order_queue.length>0) {
        for(int i=0;i<order_queue.length;i++) {
            if(order_queue[i]==0) {
                break;
            } else {
                System.out.println("\n*****\nOrder id " +
                    order_queue[i] + " is being prepared.");
                ready_queue[i] = order_queue[i];
            }
        }
        order_queue=null;
    }
}
```

```
public void barista_deliverService() {
    if(ready_queue.length>0) {
        for(int i=0;i<ready_queue.length;i++) {
            if(ready_queue[i]==0) {
                break;
            } else {
                System.out.println("\n*****\nOrder id " + ready_queue[i] + " is completed.");
            }
        }
        ready_queue=null;
    }
}
```

```
public static void main(String[] args) {
    Ques10 obj= new Ques10();
    obj.customer_requestService();
    obj.customer_requestService();
    obj.cashier_processOrder();
    obj.barista_deliverService();
}
```

```
Ques10 x Ques10 x
/home/yukti/.sdkman/candidates/java/8.0.202-amzn/bin/j
[Pay Bill] Enter amount:
234
Your Order id is 1002 , please wait.
[Pay Bill] Enter amount:
787
Your Order id is 1003 , please wait.

*****
Order id 1001 is being prepared.

*****
Order id 1002 is being prepared.

*****
Order id 1001 is completed.

*****
Order id 1002 is completed.

Process finished with exit code 0
|
```

11. Convert the following code so that it uses nested while statements instead of for statements:

```
int s = 0;
int t = 1;
for (int i = 0; i < 10; i++)
{
    s = s + i;
    for (int j = i; j > 0; j--)
    {
        t = t * (j - i);
    }
    s = s * t;
    System.out.println("T is " + t);
}
System.out.println("S is " + s);
```

```

public class Ques11 {
    public static void main(String[] args) {
        int s = 0;
        int t = 1;
        int i=0;
        while (i<10)
        {
            s = s + i;
            int j=i;
            while (j>0)
            {
                t = t * (j - i);
                j--;
            }
            s = s * t;
            System.out.println("T is " + t);
            i++;
        }
        System.out.println("S is " + s);
    }
}

```

12.What will be the output on new Child(); ?

class Parent extends Grandparent {

```

{
    System.out.println("instance - parent");
}
public Parent() {
    System.out.println("constructor - parent");
}
static {
    System.out.println("static - parent");
}
}

```

class Grandparent {

```

static {
    System.out.println("static - grandparent");
}
}

```

```

    {
        System.out.println("instance - grandparent");
    }
    public Grandparent() {
        System.out.println("constructor - grandparent");
    }
}
class Child extends Parent {
    public Child() {
        System.out.println("constructor - child");
    }
    static {
        System.out.println("static - child");
    }
    {
        System.out.println("instance - child");
    }
}

```

### **Output-**

```

static - grandparent
static - parent
static - child
instance - grandparent
constructor - grandparent
instance - parent
constructor - parent
instance - child
constructor - child

```

**Q13. Create a custom exception that do not have any stack trace.**

```

public class Ques13 {
    public static void main(String[] args) {
        try {
            Name1("YUKTI");
        }
        catch (MyException e)
        {
            System.out.println("Exception occurred :"+e);
        }
    }

    public static void Name1(String name) throws MyException
    {
        if(name.equals("YUKTI"))
        {
            throw new MyException(name);
        }
        else System.out.println("no exception");
    }
}

class MyException extends Exception
{
    public MyException(String s) { super(s); }
}

```

output-

```

/home/yukti/.sdkman/candidates/java/8.0.202-amz
Exception occurred :MyException: YUKTI

Process finished with exit code 0

```