

PRACTICAL 4:

AIM: Exception handling

Program 1:

Write a following programmed in Java using exception handling. Consider a method to determine whether a student passes an internal exam or not. The method takes the number of correct answers and total number of question asked. Calculate percentage and if it is greater than 40,the student passes. Write appropriate main function also. Main must be able to catch any exception the method might throw.

Source code:

Refer main program as source code.

Program 2:

Create a class with a main() that throws an object of class Exception inside a try block. Give the constructor for Exception a string argument. Catch the exception inside a catch clause and print out the string argument. Add a finally clause and print a message.

Source code:

Refer main program as source code.

Program 3:

Interface named InfStk, having two methods push and pop

- classStack implements InfStk.
- Stack throws custom exceptions for underflow and overflow.
- Test the class with appropriate data.

Source code:

#interface infstack code:

```
package Lab4;

public interface infstack{

    void push(int i);

    int pop();

}
```

#class stack code:

```
package Lab4;

class OverflowException extends Exception{

    OverflowException(){

        super("Overflow occurred!!");

    }

}

class UnderflowException extends Exception{

    UnderflowException(){

        super("Underflow occurred!!");

    }

}

public class stack implements infstack{

    int top=-1;

    int ary[];

    public stack(){

        ary=new int[5];

    }

    public stack(int size){

        ary=new int[size];

    }

    public boolean isEmpty(){

        return top==-1;

    }

}
```

```
    }  
    public boolean isFull(){  
        return top==ary.length-1;  
    }  
    public void push(int i){  
        try{  
            if(!isFull())  
                ary[++top]=i;  
            else  
                throw new OverflowException();  
        }  
        catch(OverflowException e){  
            System.out.println(e);  
        }  
    }  
    public int pop(){  
        try{  
            if(!isEmpty())  
                return ary[top--];  
            else  
                throw new UnderflowException();  
        }  
        catch(UnderflowException e){  
            System.out.println(e);  
        }  
        return Integer.MIN_VALUE;  
    }  
}
```

Program 4:

Write a program for creating a **Bank** class, which is used to manage the bank account of customers. Class has two methods, Deposit () and withdraw (). Deposit method display old balance and new balance after depositing the specified amount. Withdrew method display old balance and new balance after withdrawing. If balance is not enough to withdraw the money, it throws **ArithmeticException** and if balance is less than 500rs after withdrawing then it throw custom exception, **NotEnoughMoneyException**.

Source code:

```
#class bank:
package Lab4;
public class Bank{
    private double amount;
    public Bank(){
        amount=500.0;
    }
    public void deposit(double add){
        display();
        amount=amount+add;
        System.out.println("Amount added successfully!");
        display();
    }
    public void withdraw(double sub) throws NotEnoughBalance,ArithmeticException{
        if(amount-sub<0)
            throw new NotEnoughBalance();
        else if (amount-sub<500)
            throw new ArithmeticException();
        else{
            amount=amount-sub;
            System.out.println("Withdrawal successful !");
        }
    }
}
```

```

    }
}

    public void display(){
        System.out.println("amount : "+amount);
    }
}

```

#class NotEnoughBalance as custom exception.

```

package Lab4;

public class NotEnoughBalance extends Exception{
    public NotEnoughBalance(){
        super("Enough Balance is not Available!");
    }
}

```

Main class to run all programs according to choice:

```

package main;

import Lab4.*;

import java.util.Scanner;

public class Main4 {

    public static void check(int total,int correct) throws ArithmeticException{
        if(correct/total == 1);

        double res= ((float) correct)/total;

        if(res>0.40)

            System.out.println("Pass!");

        else

            System.out.println("Failed!");

    }

    public static void exp3(String args[]){
        double d1,d2;

        try{
            d1=Double.parseDouble(args[0]);

```

```

        d2=Double.parseDouble(args[1]);
        if(d2==0.0)
            throw new ArithmeticException();
        double res=d1/d2;
        System.out.println("Result : "+res);
    }
    catch (ArrayIndexOutOfBoundsException e){
        System.out.println("Please enter the valid args!");
        System.out.println(e);
    }
    catch (ArithmeticException e){
        System.out.println("Second argument cannot be zero!");
        System.out.println(e);
    }
    catch (NumberFormatException e){
        System.out.println("Only Numerical values are allowed !");
        System.out.println(e);
    }
    catch (Exception e){
        System.out.println(e);
    }
}

public static void main(String[] args) {
    Scanner sc = new Scanner(System.in);
    int ch=1;
    while(ch>0){
        System.out.println("Enter the practical Number(0 for exit) : ");
        ch=sc.nextInt();
        switch(ch){
            case 1:

```

```
        try{
            check(0,5);
        }
        catch (ArithmeticException e) {
            System.out.println(e);
        }
        try{
            check(10,6);
        }
        catch (ArithmeticException e){
            System.out.println(e);
        }
        break;
    case 2:
        myclass obj=new myclass();
        obj.myMethod();
        break;

    case 3:
        exp3(args);
        break;

    case 4:
        stack mystack=new stack(4);
        int stkch=0;
        while(stkch!=3){
            System.out.println("1.push\n2.pop\n3.Exit");
            stkch=sc.nextInt();
            switch(stkch){
                case 1:
```

```

        System.out.println("Enter the number : ");
        mystack.push(sc.nextInt());
        break;
    case 2:
        System.out.println(mystack.pop());
        break;
    }
}

break;
case 5:
    Bank bobj = new Bank();
    int bch=0;
    while(bch!=4){
        System.out.println("1.Deposit\n2.Withdraw\n3.Display\n4.Exit");
        bch=sc.nextInt();
        switch(bch){
            case 1:
                System.out.println("Enter the amount : ");
                bobj.deposit(sc.nextDouble());
                break;
            case 2:
                try{
                    bobj.withdraw(sc.nextDouble());
                }
                catch(ArithmeticException e){
                    System.out.println("Balance cannot be less than 500");
                    System.out.println(e);
                }
                catch(NotEnoughBalance e){

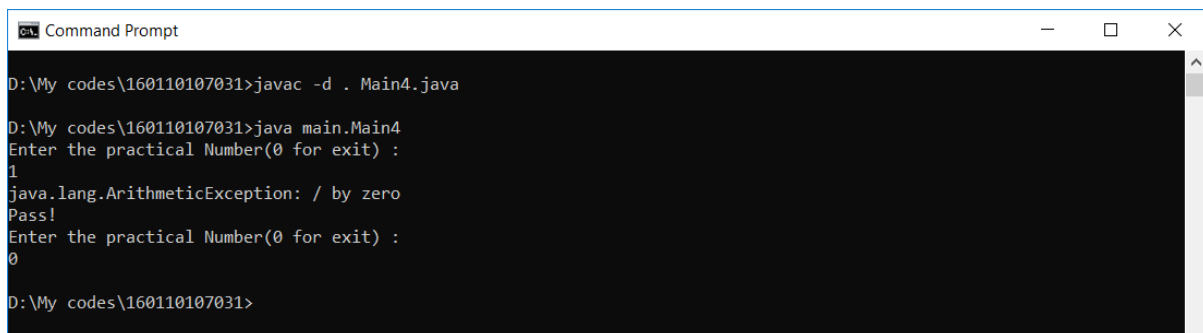
```


160110107031

```
                System.out.println(e.getMessage());
            }
            case 3:
                bobj.display();
                break;
        }
    }
}
}
```

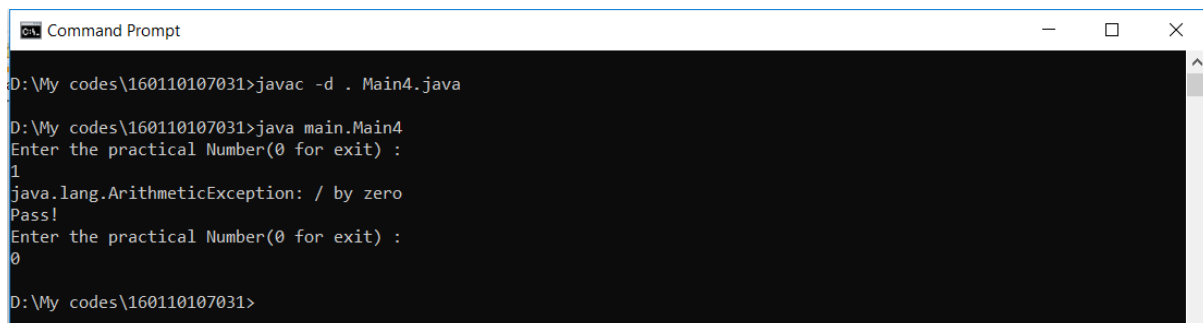
Output:

For program 1:



```
Command Prompt
D:\My codes\160110107031>javac -d . Main4.java
D:\My codes\160110107031>java main.Main4
Enter the practical Number(0 for exit) :
1
java.lang.ArithmeticException: / by zero
Pass!
Enter the practical Number(0 for exit) :
0
D:\My codes\160110107031>
```

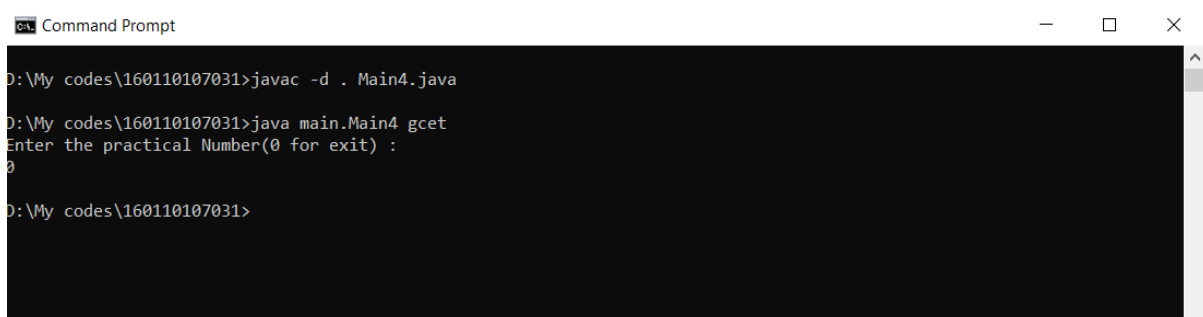
For program 2:



```
Command Prompt
D:\My codes\160110107031>javac -d . Main4.java
D:\My codes\160110107031>java main.Main4
Enter the practical Number(0 for exit) :
1
java.lang.ArithmeticException: / by zero
Pass!
Enter the practical Number(0 for exit) :
0
D:\My codes\160110107031>
```

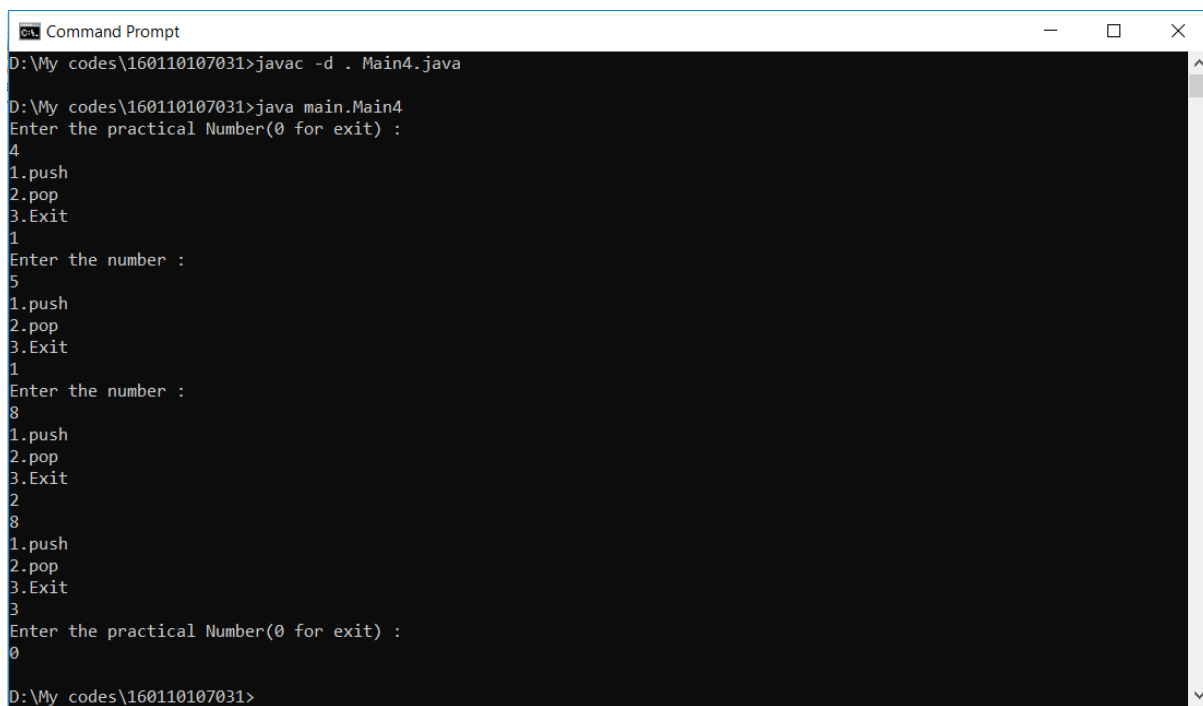
160110107031

For program 3:



```
Command Prompt
D:\My codes\160110107031>javac -d . Main4.java
D:\My codes\160110107031>java main.Main4 gcet
Enter the practical Number(0 for exit) :
0
D:\My codes\160110107031>
```

For program 4:



```
Command Prompt
D:\My codes\160110107031>javac -d . Main4.java
D:\My codes\160110107031>java main.Main4
Enter the practical Number(0 for exit) :
4
1.push
2.pop
3.Exit
1
Enter the number :
5
1.push
2.pop
3.Exit
1
Enter the number :
8
1.push
2.pop
3.Exit
2
8
1.push
2.pop
3.Exit
3
Enter the practical Number(0 for exit) :
0
D:\My codes\160110107031>
```