CS2304 JAVA PROGRAMMING

Week 8

NAME: B Satlas Rohit

REGISTER NUMBER: 2024503305

8.1 Code

```
public class Mathcalculator {
    public static double calculatePower(int base, int exponent) {
        if (base == 0 && exponent < 0) {
            throw new IllegalArgumentException("Base cannot be 0 when exponent is negative.");
        }
        if (base < 0 && exponent < 0) {
            throw new ArithmeticException("Both base and exponent cannot be negative.");
        }
        if (base == 0 && exponent == 0) {
            throw new UnsupportedOperationException("0^0 is undefined.");
        }
        return Math.pow(base, exponent);
    }
    public static long factorial(int n) {
        if (n < 0) {
            throw new IllegalArgumentException("Factorial is undefined for negative numbers.");
        }
        if (n > 20) {
            throw new ArithmeticException("Factorial overflow: n must be ≤ 20.");
```

```
        }
        long result = 1;
        for (int i = 2; i <= n; i++) {
            result *= i;
        }
        return result;
    }
    public static double safeDivide(double dividend, double divisor) {
        if (dividend == 0.0 && divisor == 0.0) {
            throw new IllegalArgumentException("Both dividend and divisor cannot be zero.");
        }
       if (divisor == 0.0) {
            throw new ArithmeticException("Division by zero is not allowed.");
        }
        return dividend / divisor;
    }
    public static void main(String[] args) {
        System.out.println("Name:"+Nullpointer(“Satlas Rohit.B)+"\nRegno.no:2024503305");
        System.out.println("Power: " + calculatePower(2, 3));
        System.out.println("Factorial: " + factorial(5));
        System.out.println("Safe Divide: " + safeDivide(10, 2));
    }
}
```

Output:

```
Name:B.Satlas Rohit
Regno.no:2024503305
Power: 8.0
Factorial: 120
Safe Divide: 5.0

Process finished with exit code 0
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Base cannot be 0 when exponent is negative.
    at Mathcalculator.calculatePower(Mathcalculator.java:4)
    at Mathcalculator.main(Mathcalculator.java:38)

Process finished with exit code 1
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Exception in thread "main" java.lang.ArithmeticException Create breakpoint : Both base and exponent cannot be negative.
    at Mathcalculator.calculatePower(Mathcalculator.java:7)
    at Mathcalculator.main(Mathcalculator.java:38)

Process finished with exit code 1
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Exception in thread "main" java.lang.UnsupportedOperationException Create breakpoint : 0^0 is undefined.
    at Mathcalculator.calculatePower(Mathcalculator.java:10)
    at Mathcalculator.main(Mathcalculator.java:38)

Process finished with exit code 1
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Power: 8.0
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Factorial is undefined for negative numbers.
    at Mathcalculator.factorial(Mathcalculator.java:16)
    at Mathcalculator.main(Mathcalculator.java:39)

Process finished with exit code 1
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Power: 8.0
Exception in thread "main" java.lang.ArithmeticException Create breakpoint : Factorial overflow: n must be ≤ 20.
    at Mathcalculator.factorial(Mathcalculator.java:19)
    at Mathcalculator.main(Mathcalculator.java:39)

Process finished with exit code 1
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Power: 8.0
Factorial: 120
Exception in thread "main" java.lang.ArithmeticException Create breakpoint : Division by zero is not allowed.
    at Mathcalculator.safeDivide(Mathcalculator.java:29)
    at Mathcalculator.main(Mathcalculator.java:40)

Process finished with exit code 1
```

```
Name:B.Satlas Rohit
Regno.no:2024503305
Power: 8.0
Factorial: 120
Exception in thread "main" java.lang.IllegalArgumentException Create breakpoint : Both dividend and divisor cannot be zero.
    at Mathcalculator.safeDivide(Mathcalculator.java:29)
    at Mathcalculator.main(Mathcalculator.java:40)

Process finished with exit code 1
```

```
Exception in thread "main" java.lang.NullPointerException Create breakpoint : The String is NUll
    at Mathcalculator.Nullpointer(Mathcalculator.java:31)
    at Mathcalculator.main(Mathcalculator.java:45)

Process finished with exit code 1
```

8.2 Code

```
class InsufficientFundsException extends Exception {
    private double balance;
    private double shortfall;
    public InsufficientFundsException(double balance, double shortfall) {
        super("Insufficient funds: Balance = " + balance + ", Shortfall = " + shortfall);
        this.balance = balance;
```

```
        this.shortfall = shortfall;
    }
    public double getBalance() {
        return balance;
    }
    public double getShortfall() {
        return shortfall;
    }
}
class InvalidAmountException extends Exception {
    public InvalidAmountException(double amount) {
        super("Invalid amount: " + amount + ". Amount must be greater than zero.");
    }
}
class BankAccount {
    private String accountId;
    private double balance;
    public BankAccount(String accountId, double initialBalance) throws
InvalidAmountException {
        if (accountId == null || accountId.trim().isEmpty()) {
            throw new IllegalArgumentException("Account ID cannot be null or empty.");
        }
        if (initialBalance < 0) {
            throw new InvalidAmountException(initialBalance);
```

```
        }
        this.accountId = accountId;
        this.balance = initialBalance;
    }
    public void deposit(double amount) throws InvalidAmountException {
        if (amount <= 0) {
            throw new InvalidAmountException(amount);
        }
        balance += amount;
        System.out.println("Deposited " + amount + " to " + accountId + ". New balance: " +
balance);
    }
    public void withdraw(double amount) throws InvalidAmountException,
InsufficientFundsException {
        if (amount <= 0) {
            throw new InvalidAmountException(amount);
        }
        if (amount > balance) {
            throw new InsufficientFundsException(balance, amount - balance);
        }
        balance -= amount;
        System.out.println("Withdrew " + amount + " from " + accountId + ". New balance: " +
balance);
    }
    public void transfer(BankAccount target, double amount) throws InvalidAmountException,
InsufficientFundsException {
```

```
        if (target == null) {
            throw new IllegalArgumentException("Target account cannot be null.");
        }
        this.withdraw(amount);
        target.deposit(amount);
        System.out.println("Transferred " + amount + " from " + accountId + " to " +
target.accountId);
    }
    public double getBalance() {
        return balance;
    }
    public String getAccountId() {
        return accountId;
    }
}
public class Bankingsystem {
    public static void main(String[] args) {
        System.out.println("Name:B.Satlas Rohit\nRegno:2024503305");
        try {
            BankAccount acc1 = new BankAccount("ACC001", 1000);
            BankAccount acc2 = new BankAccount("ACC002", 500);
            acc1.deposit(200);
            acc1.withdraw(150);
            acc1.transfer(acc2, 300);
```

```
            System.out.println("Final Balance of " + acc1.getAccountId() + ": " + acc1.getBalance());
            System.out.println("Final Balance of " + acc2.getAccountId() + ": " + acc2.getBalance());
        } catch (InvalidAmountException | InsufficientFundsException |
IllegalArgumentException e) {
            System.err.println("Error: " + e.getMessage());
        }
    }
}
```

Output:

```
Name:B.Satlas Rohit
Regno:2024503305
Deposited 200.0 to ACC001. New balance: 1200.0
Withdrew 150.0 from ACC001. New balance: 1050.0
Withdrew 300.0 from ACC001. New balance: 750.0
Deposited 300.0 to ACC002. New balance: 800.0
Transferred 300.0 from ACC001 to ACC002
Final Balance of ACC001: 750.0
Final Balance of ACC002: 800.0

Process finished with exit code 0
```

```
Deposited 450.0 to ACC001. New balance: 1450.0
Withdrew 150.0 from ACC001. New balance: 1300.0
Withdrew 300.0 from ACC001. New balance: 1000.0
Deposited 300.0 to ACC002. New balance: 800.0
Transferred 300.0 from ACC001 to ACC002
Final Balance of ACC001: 1000.0
Final Balance of ACC002: 800.0
Error: Insufficient funds: Balance = 1000.0, Shortfall = 4000.0

Process finished with exit code 0
```

8.3 Code

```
import java.io.*;
public class ex {
   static void openFile(String filename) throws IOException {
      FileReader fr = new FileReader(filename);
      BufferedReader br = new BufferedReader(fr);
      System.out.println("File opened successfully!");
      br.close();
      fr.close();
   }
   static void readFile(String filename) throws IOException {
      System.out.println("Reading file...");
      openFile(filename);
   }
   static void processFile(String filename) throws IOException {
      System.out.println("Processing file...");
      readFile(filename);
   }
   public static void main(String[] args) {
      try {
         processFile("data.txt");
      } catch (IOException e) {
         System.out.println("Exception caught in main: " + e);
      }
```

```
        System.out.println("Program continues after handling the exception.");
    }
}
```

Output:

```
Name:B.Satlas Rohit
Regno:2024503305
Processing file...
Reading file...
Exception caught in main: java.io.FileNotFoundException: data.txt (The system cannot find the file specified)
Program continues after handling the exception.
```

8.4 Code

```
import java.util.Scanner;
public class Unchecked{
    static int parseNumber(String input) {
        return Integer.parseInt(input);
    }
    static int divide(int a, int b) {
        return a / b;
    }
    static void performCalculation() {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter first number: ");
        String num1 = sc.nextLine();
        System.out.print("Enter second number: ");
        String num2 = sc.nextLine();
        int a = parseNumber(num1);
        int b = parseNumber(num2);
```

```
        int result = divide(a, b);
        System.out.println("Result = " + result);
        sc.close();
    }
    public static void main(String[] args) {
        System.out.println("Name:B.Satlas Rohit\nRegno:2024503305");
        try {
            performCalculation();
        } catch (NumberFormatException e) {
            System.out.println("Invalid input! Please enter numbers only.");
        } catch (ArithmeticException e) {
            System.out.println("Cannot divide by zero!");
        }
        System.out.println("Program continues after handling exception.");
    }
}
```

Output:

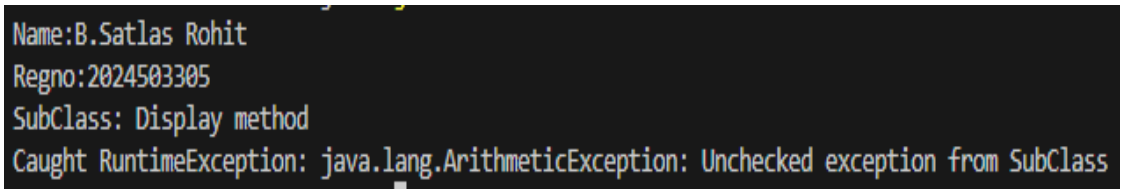
```
Name:B.Satlas Rohit
Regno:2024503305
Enter first number: 2.4
Enter second number: 2
Invalid input! Please enter numbers only.
Program continues after handling exception.
```

8.5 Code

```
import java.io.IOException;
class SuperClass{
    void display() throws IOException{
```

```
        System.out.println("SuperClass: Display method");
        throw new IOException("IOException from SuperClass");
    }
}
class SubClass extends SuperClass{
    @Override
    void display() throws RuntimeException {
        System.out.println("SubClass: Display method");
        throw new ArithmeticException("Unchecked exception from SubClass");
    }
}
public class override{
    public static void main(String[] args) {
        SuperClass obj = new SubClass();
        try {
            obj.display();
        } catch (IOException e) {
            System.out.println("Caught IOException: " + e);
        } catch (RuntimeException e) {
            System.out.println("Caught RuntimeException: " + e);
        }
    }
}
```

Output:


Name:B.Satlas Rohit
Regno:2024503305
SubClass: Display method
Caught RuntimeException: java.lang.ArithmeticException: Unchecked exception from SubClass