1.Write a Java program that reads data from a file named "data.txt". Implement error handling using try-catch blocks to handle FileNotFoundException. If the file is not found, print an error message indicating the issue.

```
import java.io.*;
import java.util.*;

public class question_1 {
    public static void main(String[] args) {
        String myFile = "text.txt";

        try{
            File file = new File(myFile);
            Scanner scan = new Scanner(file);

            while(scan.hasNextLine()){
                System.out.println(scan.nextLine());
            }

            scan.close();
        }catch(FileNotFoundException e){
            System.out.println("File not found: " + myFile);
        }
    }
}
```

```
sers\rifat\AppData\Roaming\Code\User\workspaceStorage\7
File not found: text.txt
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

2.Write a Java program that initializes an array of integers and attempts to access an element at an index beyond the array's length. Implement try-catch blocks to handle the ArrayIndexOutOfBoundsException that may occur. If the exception occurs, print a message indicating the invalid index.

```
public class question_2 {
    public static void main(String[] args) {
        int[] arr = {1,2,3,4,5};

        try{
            System.out.println(arr[6]);
        }catch(ArrayIndexOutOfBoundsException e){
            System.out.println(e);
        }
    }
}
```

```
java.lang.ArrayIndexOutOfBoundsException: Index 6 out of bounds for length 5
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

3.Write a Java program to simulate bank account transactions. Implement trycatch blocks to handle exceptions that may occur during withdrawal or deposit operations, such as InsufficientFundsException for insufficient balance and NegativeAmountException for negative amounts. Use a finally block to ensure that resources are properly released after each transaction.

```
class InsufficientFundsException extends Exception{
   InsufficientFundsException(String message){
      super(message);
   }
}

class NegativeAmountException extends Exception{
   NegativeAmountException(String message){
      super(message);
   }
}

class Bank{
   double balance = 0;

   void deposit(double amount) throws NegativeAmountException{
      if(amount < 0){
         throw new NegativeAmountException("Negative amount cannot be deposited.");
      }else{
         balance += amount;
         System.out.println("Deposit successfully. New Balance: $" + balance);
      }
   }

   void withdrow(double amount) throws InsufficientFundsException, NegativeAmountException{
      if(amount < 0){
         throw new NegativeAmountException("Negative amount cannot be withdrow.");
      }
      else if(amount > balance){
         throw new InsufficientFundsException("Insufficient funds. Right now balanace: " + balance);
      }else{
         balance -= amount;
         System.out.println("Withdrowal successful. New balanace: " + balance);
      }
   }
}

public class question_3 {
   public static void main(String[] args) {
      Bank b = new Bank();

      try{
```

```
            b.deposit(-100);
            b.withdrow(100);
        }catch (InsufficientFundsException | NegativeAmountException e){
            System.out.println(e.getMessage());
        }finally{
            System.out.println("Transaction completed. Thank your!!!!!");
        }
    }
}
```

```
Negative amount cannot be deposited.
Transaction completed. Thank your!!!!!
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

4.Imagine you have a bank account. You can deposit and withdraw money from your account. You should keep in mind that the total amount of money withdrawn from your account must not exceed the total balance present in your account. If such a scenario happens, you need to safely execute from the banking system. Implement the above case in Java with the proper utilization of user-defined exception mechanism.

```
class InsufficientFundsException extends Exception{
    InsufficientFundsException(String message){
        super(message);
    }
}

class Bank{
    double balance = 0;

    void deposit(double amount){
        balance += amount;
        System.out.println("Deposit successfully. New Balance: $" + balance);
    }

    void withdrow(double amount) throws InsufficientFundsException{
        if(amount > balance){
            throw new InsufficientFundsException("Insufficient funds. Right now balanace: " +
balance);
        }else{
            balance -= amount;
            System.out.println("Withdrowal successful. New balanace: " + balance);
        }
    }
}

public class question_4 {
    public static void main(String[] args) {
        Bank b = new Bank();
```

```
        try{
            b.deposit(100);
            b.withdrow(50);
            b.withdrow(100);
        }catch (InsufficientFundsException e){
            System.out.println(e.getMessage());
        }
    }
}
```

```
Deposit successfully. New Balance: $100.0
Withdrowal successful. New balanace: 50.0
Insufficient funds. Right now balanace: 50.0
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

5.Write a Java program to validate an email address entered by the user. Implement multiple catch blocks to handle different types of exceptions that may occur during validation, such as IllegalArgumentException for invalid format and NullPointerException for null input. Use a finally block to close any resources opened during validation.

```
import java.util.Scanner;
import java.util.regex.Pattern;

public class question_5 {
    static boolean isValidEmail(String email) {
        String emailRegex = "^[a-zA-Z0-9_+&*-]+(?:\\." +
                "[a-zA-Z0-9_+&*-]+)*@" +
                "(?:[a-zA-Z0-9-]+\\.)+[a-z" +
                "A-Z]{2,7}$";

        Pattern pat = Pattern.compile(emailRegex);
        if (email == null)
            return false;
        return pat.matcher(email).matches();
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        try{
            System.out.println("Enter your email address: ");
            String email = sc.nextLine();

            if(email == null || email.isEmpty()){
                throw new NullPointerException("Email cannot be null or empty.");
            }

            if(!isValidEmail(email)){
                throw new IllegalArgumentException("Invalid email format.");
```

```
            }

            System.out.println("Email is valid!!");
        }catch(NullPointerException | IllegalArgumentException e){
            System.out.println(e.getMessage());
        }finally{
            sc.close();
            System.out.println("Validation completed!!");
        }
    }
}
```

```
Enter your email address:
rifatuul@gmail.com
Email is valid!!
Validation completed!!
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

6.Write a program to create four threads. Inside the first thread print your Dept. 10 times but wait for 2 second before printing each time. Inside the second thread print your Name 20 times. Inside the third thread print your ID 30 times. Make sure second thread gets more OS access than the first thread and the third thread starts after finishing the second thread.

```
class myThreads extends Thread{
    String message;
    int count;
    int delayTime = 2000;
    myThreads(String message, int count){
        this.message = message;
        this.count = count;
    }

    @Override
    public void run() {
        for(int i = 1; i <= count; i++){
            System.out.println(message);

            try{
                Thread.sleep(delayTime);
            }catch(InterruptedException e){
                e.printStackTrace();
            }
        }
    }
}

public class question_6 {
    public static void main(String[] args) throws InterruptedException {
        myThreads t1 = new myThreads("Dept: CSE", 10);
```

```
myThreads t2 = new myThreads("Name: Rifat", 20);
myThreads t3 = new myThreads("ID: 101", 30);

t1.setPriority(Thread.MAX_PRIORITY);
t2.setPriority(Thread.MAX_PRIORITY);
t3.setPriority(Thread.NORM_PRIORITY);


t1.start();
t2.start();
t2.join();
t3.start();
    }
}
```

```
Dept: CSE
Name: Rifat
Dept: CSE
Name: Rifat
```

7.Write a Java program to perform matrix multiplication using multithreading for parallel computation. Implement a method that takes two matrices as input and computes their product using multiple threads, each responsible for computing a portion of the result matrix. Ensure efficient utilization of resources and minimize thread synchronization overhead.

```
class MatrixMultiplier extends Thread {
    private final int[][] matrix1;
    private final int[][] matrix2;
    private final int[][] resultMatrix;
    private final int startRow;
    private final int endRow;

    public MatrixMultiplier(int[][] matrix1, int[][] matrix2, int[][] resultMatrix, int startRow, int
endRow) {
        this.matrix1 = matrix1;
        this.matrix2 = matrix2;
        this.resultMatrix = resultMatrix;
        this.startRow = startRow;
        this.endRow = endRow;
    }

@Override
public void run() {
    multiplyMatrices(matrix1, matrix2, resultMatrix, startRow, endRow);
}
```

```
    private void multiplyMatrices(int[][] matrix1, int[][] matrix2, int[][] resultMatrix, int
startRow, int endRow) {
        int colsMatrix1 = matrix1[0].length;
        int colsMatrix2 = matrix2[0].length;
        for (int i = startRow; i < endRow; i++) {
            for (int j = 0; j < colsMatrix2; j++) {
                int sum = 0;
                for (int k = 0; k < colsMatrix1; k++) {
                    sum += matrix1[i][k] * matrix2[k][j];
                }
                resultMatrix[i][j] = sum;
            }
        }
    }
}

public class question_7 {

    public static void main(String[] args) {
        int[][] matrix1 = {
                {1, 2, 3},
                {4, 5, 6},
                {7, 8, 9}
        };

        int[][] matrix2 = {
                {9, 8, 7},
                {6, 5, 4},
                {3, 2, 1}
        };

        int rowsMatrix1 = matrix1.length;
        int colsMatrix2 = matrix2[0].length;
        int[][] resultMatrix = new int[rowsMatrix1][colsMatrix2];

        int numThreads = rowsMatrix1;
        MatrixMultiplier[] multipliers = new MatrixMultiplier[numThreads];
        int blockSize = rowsMatrix1 / numThreads;
        int startRow = 0;
        int endRow = blockSize;

        for (int i = 0; i < numThreads; i++) {
            multipliers[i] = new MatrixMultiplier(matrix1, matrix2, resultMatrix, startRow,
endRow);
            startRow = endRow;
            endRow += blockSize;
            if (i == numThreads - 2) {
                endRow = rowsMatrix1;
            }
            multipliers[i].start();
```

```
}

try {
    for (MatrixMultiplier multiplier : multipliers) {
        multiplier.join();
    }
} catch (InterruptedException e) {
    e.printStackTrace();
}

System.out.println("Result Matrix:");
        for (int[] row : resultMatrix) {
            for (int element : row) {
                System.out.print(element + " ");
            }
            System.out.println();
        }
    }
}
```

```
PS C:\Users\rifat\OneDrive\Deskto
sers\rifat\AppData\Roaming\Code\U
Result Matrix:
30 24 18
84 69 54
138 114 90
```

8.Write a Java program to compute the factorial of a given number using multithreading. Create two threads, one for computing the factorial of even numbers and the other for computing the factorial of odd numbers. Combine the results to get the final factorial.

```
import java.math.BigInteger;

class FactorialCalculator extends Thread {
    private final int inputNumber;
    private BigInteger factorialResult;

    public FactorialCalculator(int inputNumber) {
        this.inputNumber = inputNumber;
    }

public BigInteger getResult() {
    return factorialResult;
}

@Override
public void run() {
```

```
        factorialResult = calculateFactorial(inputNumber);
    }

    private BigInteger calculateFactorial(int n) {
        BigInteger factorial = BigInteger.ONE;
        for (int i = 2; i <= n; i++) {
            factorial = factorial.multiply(BigInteger.valueOf(i));
        }
        return factorial;
    }
}

public class question_8 {

    public static void main(String[] args) {
        int number = 10; // Example number

        FactorialCalculator evenFactorialCalculator = new FactorialCalculator(number % 2 == 0
? number : number - 1);
        FactorialCalculator oddFactorialCalculator = new FactorialCalculator(number % 2 == 0
? number - 1 : number);

        evenFactorialCalculator.start();
        oddFactorialCalculator.start();

        try {
            evenFactorialCalculator.join();
            oddFactorialCalculator.join();
        } catch (InterruptedException e) {
            e.printStackTrace();
        }

                                                    BigInteger          result          =
evenFactorialCalculator.getResult().multiply(oddFactorialCalculator.getResult());
        System.out.println("Factorial of " + number + " is: " + result);
    }
}
```

```
PS C:\Users\rifat\OneDrive\Desktop\java
sers\rifat\AppData\Roaming\Code\User\wo
Factorial of 10 is: 13168189440000
PS C:\Users\rifat\OneDrive\Desktop\java
```

10. Write a Java program that calculates the sum of all numbers from 1 to 100 using multiple threads. Divide the range of numbers into equal segments and assign each thread to compute the sum of a segment. Then, combine the results from all threads to get the final sum.

```
import java.util.concurrent.Callable;
import java.util.concurrent.ExecutionException;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.Future;
import java.util.ArrayList;
import java.util.List;

public class question_10 {

    static class SumRangeTask implements Callable<Integer> {
        private final int rangeStart;
        private final int rangeEnd;

        public SumRangeTask(int rangeStart, int rangeEnd) {
            this.rangeStart = rangeStart;
            this.rangeEnd = rangeEnd;
        }

        @Override
        public Integer call() {
            int rangeSum = 0;
            for (int i = rangeStart; i <= rangeEnd; i++) {
                rangeSum += i;
            }
            return rangeSum;
        }
    }

    public static void main(String[] args) {
        final int threadCount = 10;
        final int maxNumber = 100;
        final int segmentSize = maxNumber / threadCount;

        ExecutorService executor = Executors.newFixedThreadPool(threadCount);
        List<Future<Integer>> results = new ArrayList<>();

        for (int i = 0; i < threadCount; i++) {
            int rangeStart = i * segmentSize + 1;
            int rangeEnd = (i == threadCount - 1) ? maxNumber : (i + 1) * segmentSize;
            results.add(executor.submit(new SumRangeTask(rangeStart, rangeEnd)));
        }

        int totalSum = 0;
        for (Future<Integer> result : results) {
```

```
        try {
            totalSum += result.get();
        } catch (InterruptedException | ExecutionException e) {
            e.printStackTrace();
        }
    }

    executor.shutdown();

    System.out.println("The sum of numbers from 1 to 100 is: " + totalSum);
  }
}
```

```
PS C:\Users\rifat\OneDrive\Desktop\java
sers\rifat\AppData\Roaming\Code\User\wo
Factorial of 10 is: 13168189440000
PS C:\Users\rifat\OneDrive\Desktop\java
```

11.Write a program that takes a paragraph of text as input and counts the occurrences of each word. Additionally, identify the five most common words and display them along with their frequencies.

```
import java.util.*;
import java.util.stream.Collectors;

public class question_11 {

    public static void main(String[] args) {
        Scanner inputScanner = new Scanner(System.in);
        System.out.println("Enter a paragraph of text:");
        String inputText = inputScanner.nextLine();
        inputScanner.close();

        inputText = inputText.toLowerCase().replaceAll("[^a-zA-Z\\s]", "");

        String[] wordArray = inputText.split("\\s+");

        Map<String, Integer> frequencyMap = new HashMap<>();
        for (String word : wordArray) {
            frequencyMap.put(word, frequencyMap.getOrDefault(word, 0) + 1);
        }

        List<Map.Entry<String, Integer>> topWords = frequencyMap.entrySet().stream()
                .sorted((entry1, entry2) -> entry2.getValue().compareTo(entry1.getValue()))
                .limit(5)
                .collect(Collectors.toList());

        System.out.println("The five most common words are:");
        for (Map.Entry<String, Integer> entry : topWords) {
```

```
            System.out.println(entry.getKey() + ": " + entry.getValue());
        }
    }
}
```

```
Enter a paragraph of text:
I am Riatul Islam
The five most common words are:
islam: 1
i: 1
am: 1
riatul: 1
```

12.Write a program that takes a sentence and a word as input and finds whether the word is present as a substring in the sentence.

```
import java.util.Scanner;

public class question_12 {

    public static void main(String[] args) {
        Scanner userInputScanner = new Scanner(System.in);

        System.out.println("Enter a sentence:");
        String userSentence = userInputScanner.nextLine();

        System.out.println("Enter a word to search:");
        String searchWord = userInputScanner.nextLine();

        userInputScanner.close();

        if (userSentence.contains(searchWord)) {
            System.out.println("The word \"" + searchWord + "\" is present in the sentence.");
        } else {
            System.out.println("The word \"" + searchWord + "\" is not present in the sentence.");
        }
    }
}
```

```
Enter a sentence:
I love my country
Enter a word to search:
love
The word "love" is present in the sentence.
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

13.Write a program that takes a sentence as input and capitalizes the first letter of each word. For example, "hello world" should become "Hello World".

```
import java.util.Scanner;

public class CapitalizeWords {

    public static void main(String[] args) {
        Scanner inputScanner = new Scanner(System.in);

        System.out.println("Enter a sentence:");
        String inputSentence = inputScanner.nextLine();

        inputScanner.close();

        String capitalizedResult = capitalizeWords(inputSentence);

        System.out.println("Capitalized sentence: " + capitalizedResult);
    }

    private static String capitalizeWords(String sentence) {
        String[] wordArray = sentence.split("\\s+");
        StringBuilder capitalizedSentenceBuilder = new StringBuilder();

        for (String word : wordArray) {
            if (word.length() > 0) {
                capitalizedSentenceBuilder.append(Character.toUpperCase(word.charAt(0)))
                                          .append(word.substring(1).toLowerCase())
                                          .append(" ");
            }
        }

        return capitalizedSentenceBuilder.toString().trim();
    }
}
```

```
Enter a sentence:
hello world
Capitalized sentence: Hello World
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

14.Create a function that takes a sentence as input and reverses the order of words in it. For example, "Hello world" should become "world Hello".

```
import java.util.Scanner;

public class question_14 {
```

```
public static void main(String[] args) {
    Scanner inputScanner = new Scanner(System.in);

    System.out.println("Enter a sentence:");
    String userSentence = inputScanner.nextLine();

    inputScanner.close();

    String reversedResult = reverseWordOrder(userSentence);

    System.out.println("Reversed sentence: " + reversedResult);
  }

  private static String reverseWordOrder(String sentence) {
    String[] wordsArray = sentence.split("\\s+");
    StringBuilder reversedSentenceBuilder = new StringBuilder();

    for (int i = wordsArray.length - 1; i >= 0; i--) {
      reversedSentenceBuilder.append(wordsArray[i]).append(" ");
    }

    return reversedSentenceBuilder.toString().trim();
  }
}
```

```
Enter a sentence:
Hello world
Reversed sentence: world Hello
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

15.Write a program that counts the occurrences of each character in a given string and displays the count for each character.

```
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class question_15 {

    public static void main(String[] args) {
        Scanner inputScanner = new Scanner(System.in);

        System.out.println("Enter a string:");
        String userString = inputScanner.nextLine();

        inputScanner.close();

        Map<Character, Integer> charCountMap = getCharacterCount(userString);
```

```
    System.out.println("Character occurrences:");
    for (Map.Entry<Character, Integer> entry : charCountMap.entrySet()) {
        System.out.println(entry.getKey() + ": " + entry.getValue());
    }
}

private static Map<Character, Integer> getCharacterCount(String inputString) {
    Map<Character, Integer> charCountMap = new HashMap<>();

    for (char character : inputString.toCharArray()) {
        charCountMap.put(character, charCountMap.getOrDefault(character, 0) + 1);
    }

    return charCountMap;
  }
}
```

```
Enter a string:
I am Rifatul Islam
Character occurrences:
 : 3
a: 3
R: 1
s: 1
t: 1
u: 1
f: 1
I: 2
i: 1
l: 2
m: 2
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

16.Write a program that takes the first name and last name of a person as input and concatenates them to form a full name.

```
import java.util.Scanner;

public class question_16 {

    public static void main(String[] args) {
        Scanner userInputScanner = new Scanner(System.in);

        System.out.println("Enter your first name:");
        String inputFirstName = userInputScanner.nextLine();

        System.out.println("Enter your last name:");
        String inputLastName = userInputScanner.nextLine();

        userInputScanner.close();
```

```
    String completeName = generateFullName(inputFirstName, inputLastName);

    System.out.println("Full name: " + completeName);
  }

  private static String generateFullName(String firstName, String lastName) {
    return firstName + " " + lastName;
  }
}
```

```
Enter your first name:
Rifatul
Enter your last name:
Islam
Full name: Rifatul Islam
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

17.Given the following strings: A = "The early bird catches the worm" B = "Patience is a virtue" Your task is to extract the word "early" from A and "virtue" from B. Then, concatenate these two words to form a sentence. After that, capitalize the sentence and find the last occurrence of the letter 'V' from the capitalized sentence. Perform all of these tasks using proper String class functions.

```
Public class question_17 {

  public static void main(String[] args) {
    String stringA = "The early bird catches the worm";
    String   oolea = "Patience is a virtue";

    String wordFromA = extractWord(stringA, "early");
    String wordFromB = extractWord(   oolea, "virtue");

    String concatenatedSentence = wordFromA + " " + wordFromB;

    String capitalizedSentence = concatenatedSentence.toUpperCase();

    int lastIndexOfV = capitalizedSentence.lastIndexOf('V');

    System.out.println("Concatenated Sentence: " + concatenatedSentence);
    System.out.println("Capitalized Sentence: " + capitalizedSentence);
    System.out.println("Last occurrence of 'V': " + lastIndexOfV);
  }

  private static String extractWord(String sentence, String targetWord) {
    int startIndex = sentence.indexOf(targetWord);
    if (startIndex != -1) {
      return sentence.substring(startIndex, startIndex + targetWord.length());
```

```
    }
    return "";
  }
}
```

```
Concatenated Sentence: early virtue
Capitalized Sentence: EARLY VIRTUE
Last occurrence of 'V': 6
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

18.You are developing a ticket booking system for a movie theater. Design a Java program that uses a Queue to manage ticket requests, where each request represents a customer wanting to book a ticket. Implement methods to add new booking requests, process bookings in the order they were received, and display the status of ticket bookings.

```
Import java.util.LinkedList;
import java.util.Queue;

class BookingSystem {
   private Queue<String> bookingRequests;

   public BookingSystem() {
      bookingRequests = new LinkedList<>();
   }

public void addRequest(String customerName) {
   bookingRequests.offer(customerName);
   System.out.println(customerName + " added to the booking queue.");
}

public void processRequests() {
   while (!bookingRequests.isEmpty()) {
      String customer = bookingRequests.poll();
      System.out.println("Booking ticket for " + customer);
   }
   System.out.println("All ticket bookings processed.");
}
public void displayStatus() {
   if (bookingRequests.isEmpty()) {
      System.out.println("No pending booking requests.");
   } else {
      System.out.println("Pending booking requests:");
      for (String customer : bookingRequests) {
         System.out.println(customer);
      }
   }
  }
}
```

```
public class question_18 {
    public static void main(String[] args) {
        BookingSystem bookingSystem = new BookingSystem();

        bookingSystem.addRequest("Alice");
        bookingSystem.addRequest("Bob");
        bookingSystem.addRequest("Charlie");

        bookingSystem.displayStatus();

        bookingSystem.processRequests();
    }
}
```

```
Alice added to the booking queue.
Bob added to the booking queue.
Charlie added to the booking queue.
Pending booking requests:
Alice
Bob
Charlie
Booking ticket for Alice
Booking ticket for Bob
Booking ticket for Charlie
All ticket bookings processed.
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

19.Create a class named Car with properties such as price (double), brand (String), and speed (double). These properties will be initialized when an object of the class is created. Create five objects of the Car class and add them to an ArrayList. Display the cars whose price is over 2000000 takas. Complete the program.

```
Import java.util.ArrayList;
import java.util.List;

class Car {
    private double carPrice;
    private String carBrand;
    private double carSpeed;

    public Car(double price, String brand, double speed) {
        this.carPrice = price;
        this.carBrand = brand;
        this.carSpeed = speed;
    }

public double getPrice() {
    return carPrice;
}

public String getBrand() {
    return carBrand;
```

```
    }

    public double getSpeed() {
        return carSpeed;
    }
}

public class question_19 {
    public static void main(String[] args) {
        List<Car> carList = new ArrayList<>();

        carList.add(new Car(2500000, "Toyota", 180));
        carList.add(new Car(1800000, "Honda", 200));
        carList.add(new Car(3000000, "BMW", 250));
        carList.add(new Car(1500000, "Ford", 170));
        carList.add(new Car(2200000, "Mercedes", 210));

        System.out.println("Cars with price over 2000000 takas:");
        for (Car car : carList) {
            if (car.getPrice() > 2000000) {
                System.out.println("Brand: " + car.getBrand() + ", Price: " + car.getPrice() + " takas,
Speed: " + car.getSpeed() + " km/h");
            }
        }
    }
}
```

```
Cars with price over 2000000 takas:
Brand: Toyota, Price: 2500000.0 takas, Speed: 180.0 km/h
Brand: BMW, Price: 3000000.0 takas, Speed: 250.0 km/h
Brand: Mercedes, Price: 2200000.0 takas, Speed: 210.0 km/h
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

20.Write a basic Java program for managing student IDs and their grades in a gradebook system. Implement methods to add new student IDs, remove existing student IDs, display the list of student IDs, and store/display grades for each student. Utilize simple data structures like arrays for storing student IDs and grades.

```
Import java.util.Arrays;
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class question_20 {
    private String[] studentIDs;
    private Map<String, Double> studentGrades;

    public question_20() {
```

```
    studentIDs = new String[0];
    studentGrades = new HashMap<>();
}

public void addStudent(String id) {
    studentIDs = Arrays.copyOf(studentIDs, studentIDs.length + 1);
    studentIDs[studentIDs.length – 1] = id;
    System.out.println(“Student ID “ + id + “ added successfully.”);
}

public void removeStudent(String id) {
    int indexToRemove = -1;
    for (int I = 0; I < studentIDs.length; i++) {
        if (studentIDs[i].equals(id)) {
            indexToRemove = I;
            break;
        }
    }
    if (indexToRemove != -1) {
        studentIDs[indexToRemove] = studentIDs[studentIDs.length – 1];
        studentIDs = Arrays.copyOf(studentIDs, studentIDs.length – 1);
        studentGrades.remove(id);
        System.out.println(“Student ID “ + id + “ removed successfully.”);
    } else {
        System.out.println(“Student ID “ + id + “ not found.”);
    }
}

public void displayStudentIDs() {
    System.out.println(“List of Student IDs:”);
    for (String id : studentIDs) {
        System.out.println(id);
    }
}

public void addGrade(String id, double grade) {
    if (Arrays.asList(studentIDs).contains(id)) {
        studentGrades.put(id, grade);
        System.out.println(“Grade “ + grade + “ added for student ID “ + id + “.”);
    } else {
        System.out.println(“Student ID “ + id + “ not found.”);
    }
}

public void displayGrades() {
    System.out.println(“Grades for Student IDs:”);
    for (Map.Entry<String, Double> entry : studentGrades.entrySet()) {
        System.out.println(“Student ID: “ + entry.getKey() + “, Grade: “ + entry.getValue());
    }
}
```

```
public static void main(String[] args) {
    question_20 gradebook = new question_20();
    Scanner scanner = new Scanner(System.in);

      oolean exit = false;
    while (!exit) {
        System.out.println(“\nGradebook System Menu:”);
        System.out.println(“1. Add Student”);
        System.out.println(“2. Remove Student”);
        System.out.println(“3. Display Student IDs”);
        System.out.println(“4. Add Grade”);
        System.out.println(“5. Display Grades”);
        System.out.println(“6. Exit”);
        System.out.print(“Enter your choice: “);
        int choice = scanner.nextInt();
        scanner.nextLine(); // Consume newline

        switch (choice) {
            case 1:
                System.out.print(“Enter student ID to add: “);
                String addID = scanner.nextLine();
                gradebook.addStudent(addID);
                break;
            case 2:
                System.out.print(“Enter student ID to remove: “);
                String removeID = scanner.nextLine();
                gradebook.removeStudent(removeID);
                break;
            case 3:
                gradebook.displayStudentIDs();
                break;
            case 4:
                System.out.print(“Enter student ID: “);
                String id = scanner.nextLine();
                System.out.print(“Enter grade: “);
                double grade = scanner.nextDouble();
                gradebook.addGrade(id, grade);
                break;
            case 5:
                gradebook.displayGrades();
                break;
            case 6:
                exit = true;
                System.out.println(“Exiting Gradebook System. Goodbye!”);
                break;
            default:
                System.out.println(“Invalid choice. Please enter a number between 1 and 6.”);
        }
    }
```

```
        scanner.close();
    }
}
```

```
Gradebook System Menu:
1. Add Student
2. Remove Student
3. Display Student IDs
4. Add Grade
5. Display Grades
6. Exit
Enter your choice: 1
Enter student ID to add: 46
Student ID 46 added successfully.

Gradebook System Menu:
1. Add Student
2. Remove Student
3. Display Student IDs
4. Add Grade
5. Display Grades
6. Exit
Enter your choice:
```

21.Create a class named Student. Write a program to insert 10 Student objects in a Stack list. Now take user input for a variable named "menu". If menu is 1 then insert another Student object. If menu is 2, delete the top Student object from the stack list. If menu is 3, just output the top Student object. Use proper stack list methods.

```
import java.util.Scanner;
import java.util.Stack;

class Student {
    private String studentName;
    private int studentRollNumber;

    public Student(String name, int rollNumber) {
        this.studentName = name;
        this.studentRollNumber = rollNumber;
    }

    public String getName() {
        return studentName;
    }

    public int getRollNumber() {
```

```
        return studentRollNumber;
    }
}

public class question_21 {
    public static void main(String[] args) {
        Stack<Student> studentStack = new Stack<>();
        Scanner scanner = new Scanner(System.in);

        for (int i = 1; i <= 10; i++) {
            studentStack.push(new Student("Student" + i, i));
        }

        int menuChoice = 0;
        while (menuChoice != 4) {
            System.out.println("\nMenu:");
            System.out.println("1. Insert a Student");
            System.out.println("2. Delete the top Student");
            System.out.println("3. Output the top Student");
            System.out.println("4. Exit");
            System.out.print("Enter your choice: ");
            menuChoice = scanner.nextInt();
            scanner.nextLine();

            switch (menuChoice) {
                case 1:
                    System.out.print("Enter student name: ");
                    String name = scanner.nextLine();
                    System.out.print("Enter student roll number: ");
                    int rollNumber = scanner.nextInt();
                    scanner.nextLine();
                    studentStack.push(new Student(name, rollNumber));
                    System.out.println("Student inserted successfully.");
                    break;
                case 2:
                    if (!studentStack.isEmpty()) {
                        Student deletedStudent = studentStack.pop();
                       System.out.println("Top student deleted: " + deletedStudent.getName() + ", Roll
Number: " + deletedStudent.getRollNumber());
                    } else {
                        System.out.println("Stack is empty. No student to delete.");
                    }
                    break;
                case 3:
                    if (!studentStack.isEmpty()) {
                        Student topStudent = studentStack.peek();
                         System.out.println("Top student: " + topStudent.getName() + ", Roll Number:
" + topStudent.getRollNumber());
                    } else {
                        System.out.println("Stack is empty. No student to display.");
```

```
            }
            break;
        case 4:
            System.out.println("Exiting the program.");
            break;
        default:
            System.out.println("Invalid choice. Please enter a number between 1 and 4.");
    }
}

    scanner.close();
  }
}
```

```
Menu:
1. Insert a Student
2. Delete the top Student
3. Output the top Student
4. Exit
Enter your choice: 1
Enter student name: Riatul Islam
Enter student roll number: 46
Student inserted successfully.

Menu:
1. Insert a Student
2. Delete the top Student
3. Output the top Student
4. Exit
Enter your choice: 4
Exiting the program.
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```

22.Write a Java program to remove duplicates from a list of strings. Implement a method remove duplicates that takes a List of strings as input and removes any duplicate elements, keeping only the first occurrence of each element.

```
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;

public class question_22 {

    public static void main(String[] args) {
        List<String> originalList = new ArrayList<>();
```

```
originalList.add("apple");
originalList.add("banana");
originalList.add("apple");
originalList.add("orange");
originalList.add("banana");

System.out.println("Original list: " + originalList);

List<String> listWithoutDuplicates = removeDuplicates(originalList);

System.out.println("List without duplicates: " + listWithoutDuplicates);
    }
    public static List<String> removeDuplicates(List<String> inputList) {
        Set<String> uniqueSet = new HashSet<>(inputList);
        return new ArrayList<>(uniqueSet);
    }
}
```

```
Original list: [apple, banana, apple, orange, banana]
List without duplicates: [banana, orange, apple]
PS C:\Users\rifat\OneDrive\Desktop\java assignment>
```