1. Write a program to print n natural number in descending order using a while loop

```java
import java.util.Scanner;

class Main {
  public static void main(String[] args) {
    // Create a Scanner object for user input
    Scanner sc = new Scanner(System.in);

    // Prompt the user to enter a number
    System.out.print("Enter a number:");

    // Read an integer from the user's input and store it in the
variable 'n'
    int n = sc.nextInt();

    // Use a while loop to print numbers from 'n' down to 1
    while (n > 0) {
      System.out.println(n);  // Print the current value of 'n'
      n--;                    // Decrement the value of 'n' by 1
    }

    // Close the Scanner to release system resources
    sc.close();
  }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/1. Write a p
rogram to print n natural number in descending order using a while lo
op
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/1. Write a p
rogram to print n natural number in descending order using a while lo
op
$ java Main
Enter a number:6
6
5
4
3
2
1
```

2. Write a program which counts vowels and consonant in a word

```java
import java.util.Scanner;


class Main {
  public static void main(String[] args) {
    // Create a Scanner object for user input
    Scanner sc = new Scanner(System.in);

    // Prompt the user to enter a number
    System.out.print("Enter a number:");

    // Read an integer from the user's input and store it in the
variable 'n'
    int n = sc.nextInt();

    // Use a while loop to print numbers from 'n' down to 1
    while (n > 0) {
      System.out.println(n);  // Print the current value of 'n'
      n--;                    // Decrement the value of 'n' by 1
    }

    // Close the Scanner to release system resources
    sc.close();
  }
}
```

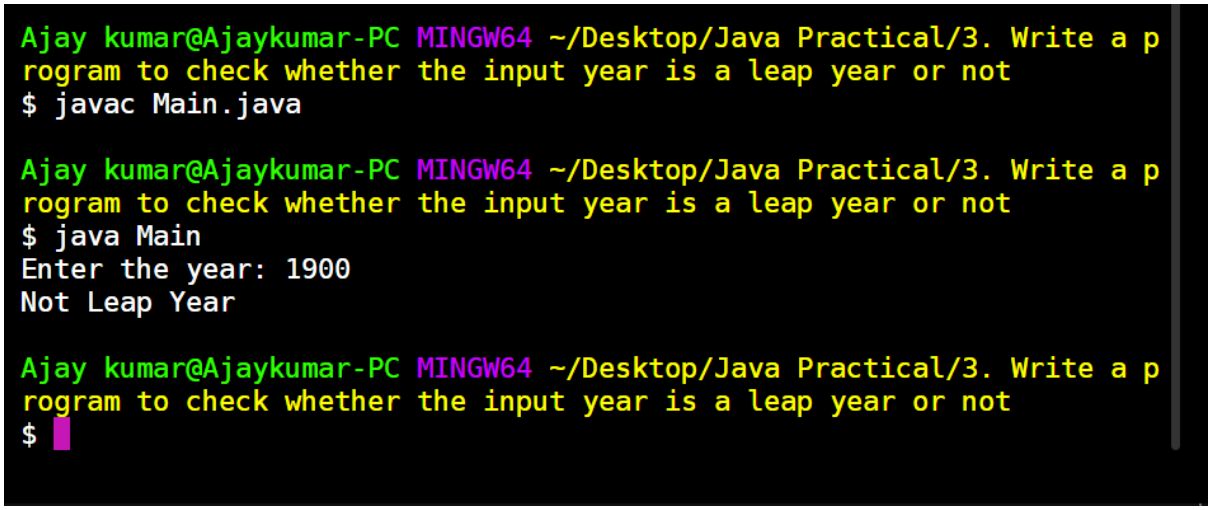
Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/2. Write a p
rogram which counts vowels and consonant in a word
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/2. Write a p
rogram which counts vowels and consonant in a word
$ java Main
Enter a word: Ajay kumar
The no. of vowels: 4
The no. of consonants: 6

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/2. Write a p
rogram which counts vowels and consonant in a word
$
```

3. Write a program to check whether the input year is a leap year or not

```
import java.util.Scanner;


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

    // Prompt the user to enter a year
    System.out.print("Enter the year: ");

    // Read the user's input as an integer and store it in the 'year' variable
    int year = sc.nextInt();

    // Check if the entered year is a leap year
    if (year % 4 == 0) {
      if (year % 100 == 0) {
        if (year % 400 == 0) {
          // If the year is divisible by 400, it's a leap year
          System.out.println("Leap Year");
        } else {
          // If the year is divisible by 4, divisible by 100 and not divisible by 100, it's not a leap year
          System.out.println("Not Leap Year");
        }
      } else {
        // If the year is divisible by 4 but not divisible by 100, it's a leap year
        System.out.println("Leap Year");
      }
    } else {
      // If the year is not divisible by 4, it's not a leap year
      System.out.println("Not Leap Year");
    }

    // Close the Scanner to release system resources
    sc.close();
  }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/3. Write a p
rogram to check whether the input year is a leap year or not
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/3. Write a p
rogram to check whether the input year is a leap year or not
$ java Main
Enter the year: 1900
Not Leap Year
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/3. Write a p
rogram to check whether the input year is a leap year or not
$

4. Write a program to print largest of three numbers with the help of the if-else

```java
import java.util.Scanner;


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

        // Prompt the user to enter the first number
        System.out.print("Enter 1st number: ");
        int a = sc.nextInt();

        // Prompt the user to enter the second number
        System.out.print("Enter 2nd number: ");
        int b = sc.nextInt();

        // Prompt the user to enter the third number
        System.out.print("Enter 3rd number: ");
        int c = sc.nextInt();

        // Check which number is the largest using if-else statements
        if (a > b && a > c) {
            System.out.println("The largest number is: " + a);
        } else if (b > a && b > c) {
            System.out.println("The largest number is: " + b);
        } else {
            System.out.println("The largest number is: " + c);
        }

        // Close the scanner to release resources
        sc.close();
    }
}
```

Output:

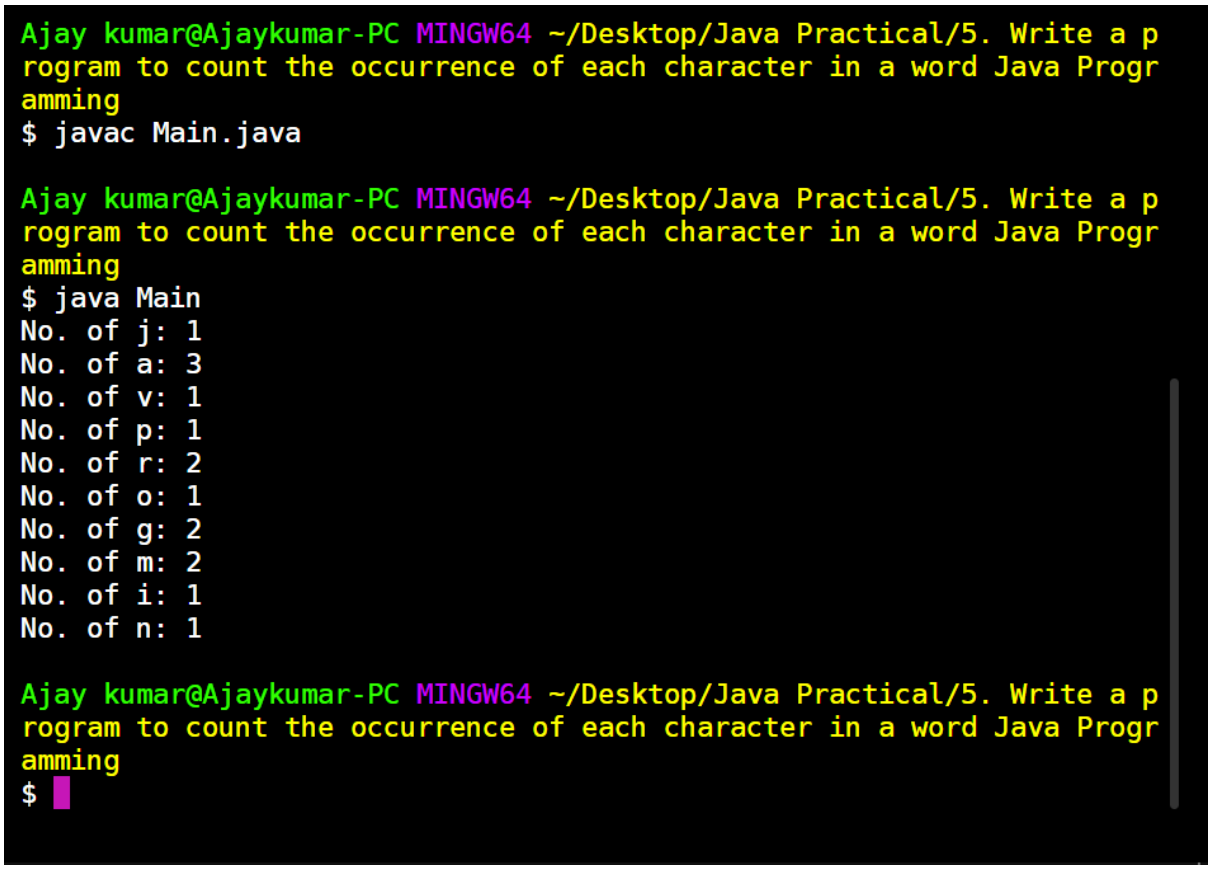
```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/4. Write a p
rogram to print largest of three numbers with the help of the if-else
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/4. Write a p
rogram to print largest of three numbers with the help of the if-else
$ java Main
Enter 1st number: 5
Enter 2nd number: 3
Enter 3rd number: 4
The largest number is: 5

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/4. Write a p
rogram to print largest of three numbers with the help of the if-else
$
```

5. Write a program to count the occurrence of each character in a word Java Programming

```
import java.util.Scanner;


class Main {
    public static void main(String[] args) {
        // The input word to analyze
        String word = "Java Programming";
        // Convert the word to lowercase to treat uppercase and
lowercase characters the same
        word = word.toLowerCase();
        int counter; // Variable to count the occurrences of each
character

        // Loop through each character in the word
        for (int i = 0; i < word.length(); i++) {
            if (word.charAt(i) == ' ') {
                // Skip spaces and continue to the next character
            } else {
                counter = 1; // Initialize the counter to 1 for the
current character

                // Check if the character has already been counted
                for (int a = 0; a < i; a++) {
                    if (word.charAt(i) == word.charAt(a)) {
                        counter = 0; // If the character has been
encountered before, set the counter to 0
                    }
                }

                // If the character is unique, count its occurrences in
the rest of the word
                if (counter == 1) {
                    for (int j = i + 1; j < word.length(); j++) {
                        if (word.charAt(i) == word.charAt(j)) {
                            counter++; // Increment the counter for
each occurrence of the character
                        }
                    }
                    // Print the character and its count
                    System.out.println("No. of " + word.charAt(i) + ":
" + counter);
                }
            }
        }
```

```
    }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/5. Write a program to count the occurrence of each character in a word Java Programming
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/5. Write a program to count the occurrence of each character in a word Java Programming
$ java Main
No. of j: 1
No. of a: 3
No. of v: 1
No. of p: 1
No. of r: 2
No. of o: 1
No. of g: 2
No. of m: 2
No. of i: 1
No. of n: 1
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/5. Write a program to count the occurrence of each character in a word Java Programming
$

6. Write a program that accepts radius and returns the area of a circle using function call

```java
import java.util.Scanner;

class Main {
    // A function to calculate the area of a circle based on its radius
    static double areaOfCircle(double radius) {
        double area = 3.14 * radius * radius; // Calculate the area using the formula πr^2
        return area; // Return the calculated area
    }

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

        // Prompt the user to enter the radius of the circle
        System.out.print("Enter the radius: ");

        // Read the user's input as a double and store it in the 'r' variable
        double r = sc.nextDouble();

        // Call the 'areaOfCircle' function to calculate the area
        double a = areaOfCircle(r);

        // Display the calculated area
        System.out.println("Area = " + a);

        // Close the Scanner to release system resources
        sc.close();
    }
}
```

Output:

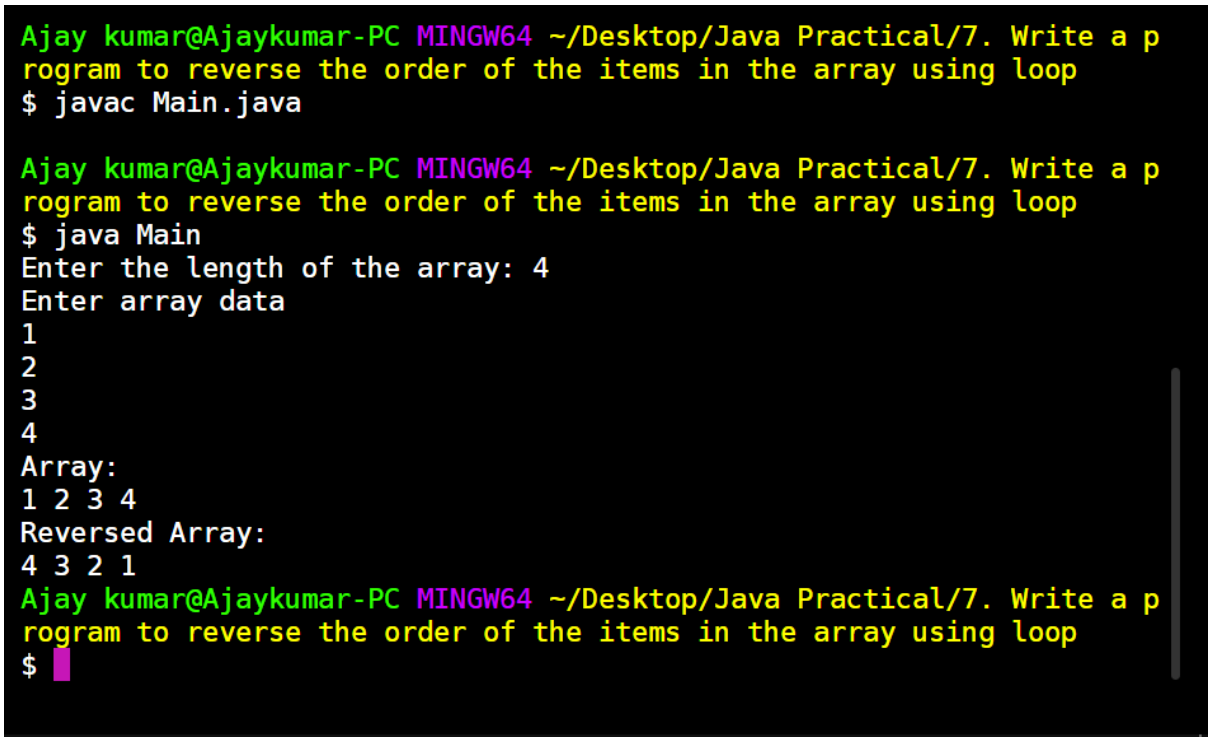
```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/6. Write a p
rogram that accepts radius and returns the area of a circle using fun
ction call
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/6. Write a p
rogram that accepts radius and returns the area of a circle using fun
ction call
$ java Main
Enter the radius: 5
Area = 78.5

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/6. Write a p
rogram that accepts radius and returns the area of a circle using fun
ction call
$
```

7. Write a program to reverse the order of the items in the array using loop

```
import java.util.Scanner;


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

        // Prompt the user to enter the length of the array
        System.out.print("Enter the length of the array: ");
        int len = sc.nextInt();

        int[] arr = new int[len];
        System.out.println("Enter array data");

        // Read input data for the array
        for (int i = 0; i < len; i++) {
            arr[i] = sc.nextInt();
        }

        System.out.println("Array: ");

        // Display the original array
        for (int j = 0; j < len; j++) {
            System.out.print(arr[j] + " ");
        }

        int temp;

        // Reverse the array by swapping elements from the beginning
and end
        for (int k = 0; k <= (len / 2) - 1; k++) {
            temp = arr[k];
            arr[k] = arr[len - 1 - k];
            arr[len - 1 - k] = temp;
        }

        System.out.println("\nReversed Array: ");

        // Display the reversed array
        for (int l = 0; l < len; l++) {
            System.out.print(arr[l] + " ");
        }

        sc.close(); // Close the scanner to release resources
```

```
    }
}
```

Output:

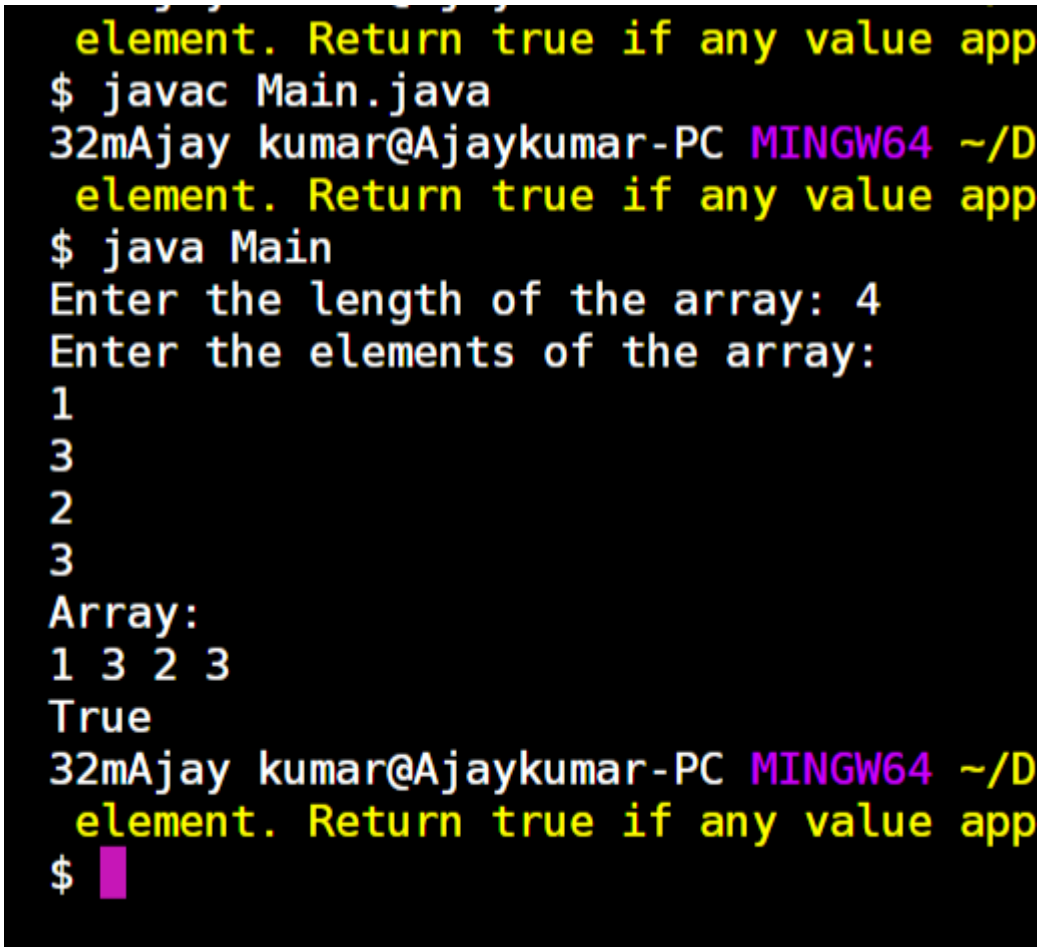

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/7. Write a program to reverse the order of the items in the array using loop
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/7. Write a program to reverse the order of the items in the array using loop
$ java Main
Enter the length of the array: 4
Enter array data
1
2
3
4
Array:
1 2 3 4
Reversed Array:
4 3 2 1
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/7. Write a program to reverse the order of the items in the array using loop
$

8. Write a program to find whether a given array of integers contains any duplicate element. Return true if any value appears at least twice in the said array and return false if every element is distinct

```
import java.util.Scanner;


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

        System.out.print("Enter the length of the array: ");
        int len = sc.nextInt();

        int[] arr = new int[len];

        System.out.println("Enter the elements of the array: ");

        // Read input values and populate the array
        for (int i = 0; i < len; i++) {
            arr[i] = sc.nextInt();
        }

        System.out.println("Array: ");

        // Display the original array
        for (int j = 0; j < len; j++) {
            System.out.print(arr[j] + " ");
        }

        // Check for duplicate elements in the array using nested loops
        for (int a = 0; a < len; a++) {
            for (int b = a + 1; b < len; b++) {
                if (arr[a] == arr[b]) {
                    // If a duplicate is found, print "True," close the
scanner, and return
                    System.out.println("\nTrue");
                    sc.close();
                    return;
                }
            }
        }

        // If no duplicates are found, print "False" and close the
scanner
```

```
            System.out.println("\nFalse");
            sc.close();
            return;
        }
}
```

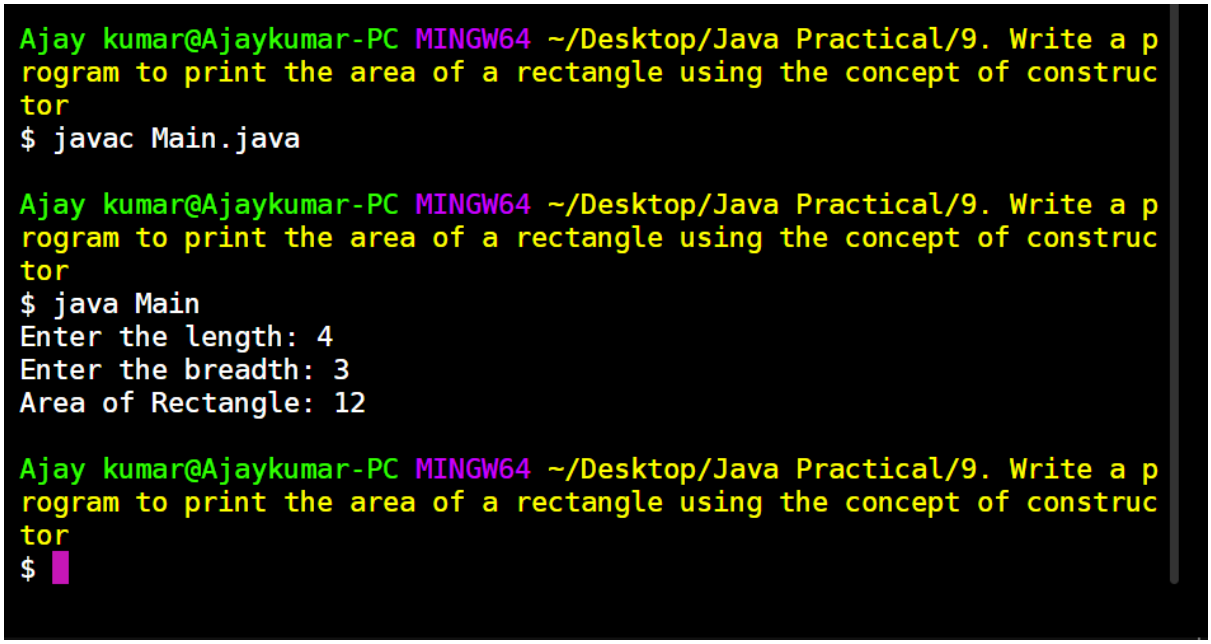
Output:



9. Write a program to print the area of a rectangle using the concept of constructor

```
import java.util.Scanner;

// Define a class called Rectangle
class Rectangle {
    // Constructor for the Rectangle class that calculates and prints
the area
    public Rectangle(int length, int breadth) {
        // Calculate the area of the rectangle
        int area = length * breadth;

        // Print the area of the rectangle
        System.out.println("Area of Rectangle: " + area);
    }
}

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

        // Prompt the user to enter the length of the rectangle
        System.out.print("Enter the length: ");
        int length = sc.nextInt();

        // Prompt the user to enter the breadth of the rectangle
        System.out.print("Enter the breadth: ");
        int breadth = sc.nextInt();

        // Create an instance of the Rectangle class, passing the
length and breadth as parameters
        Rectangle rect = new Rectangle(length, breadth);

        // Close the scanner to release resources
        sc.close();
    }
}
```

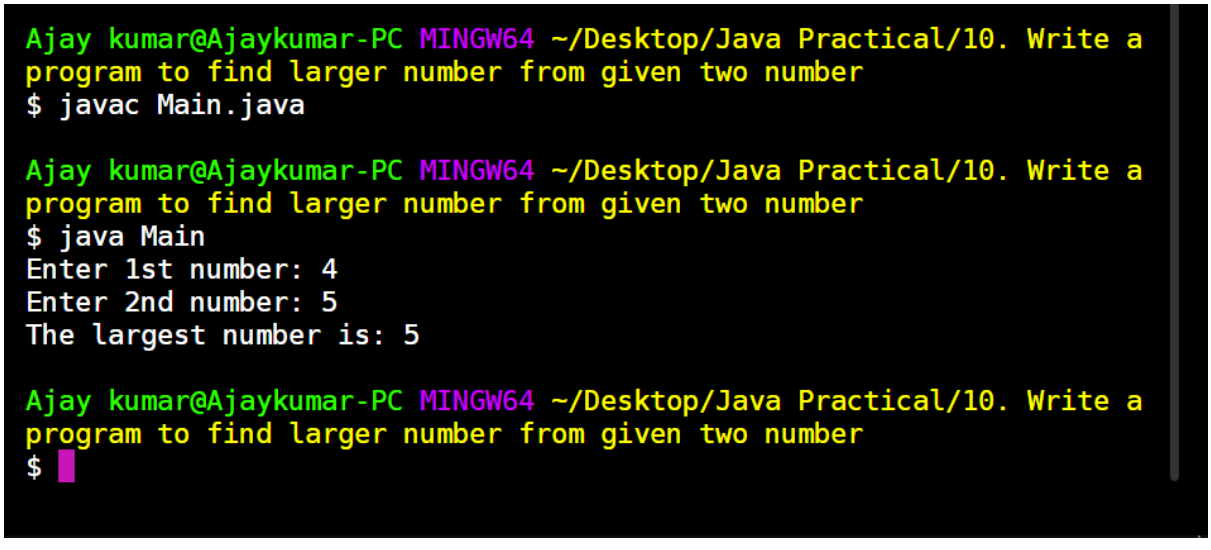
Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/9. Write a program to print the area of a rectangle using the concept of constructor
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/9. Write a program to print the area of a rectangle using the concept of constructor
$ java Main
Enter the length: 4
Enter the breadth: 3
Area of Rectangle: 12
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/9. Write a program to print the area of a rectangle using the concept of constructor
$

10. Write a program to find larger number from given two number

```java
import java.util.Scanner;

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

        // Prompt the user to enter the first number
        System.out.print("Enter 1st number: ");
        int a = sc.nextInt();

        // Prompt the user to enter the second number
        System.out.print("Enter 2nd number: ");
        int b = sc.nextInt();

        // Check which number is the largest using if-else statements
        if (a > b) {
            // If 'a' is greater, print it as the largest number
            System.out.println("The largest number is: " + a);
        } else {
            // If 'b' is greater or equal to 'a', print 'b' as the
largest number
            System.out.println("The largest number is: " + b);
        }

        // Close the scanner to release resources
        sc.close();
    }
}
```

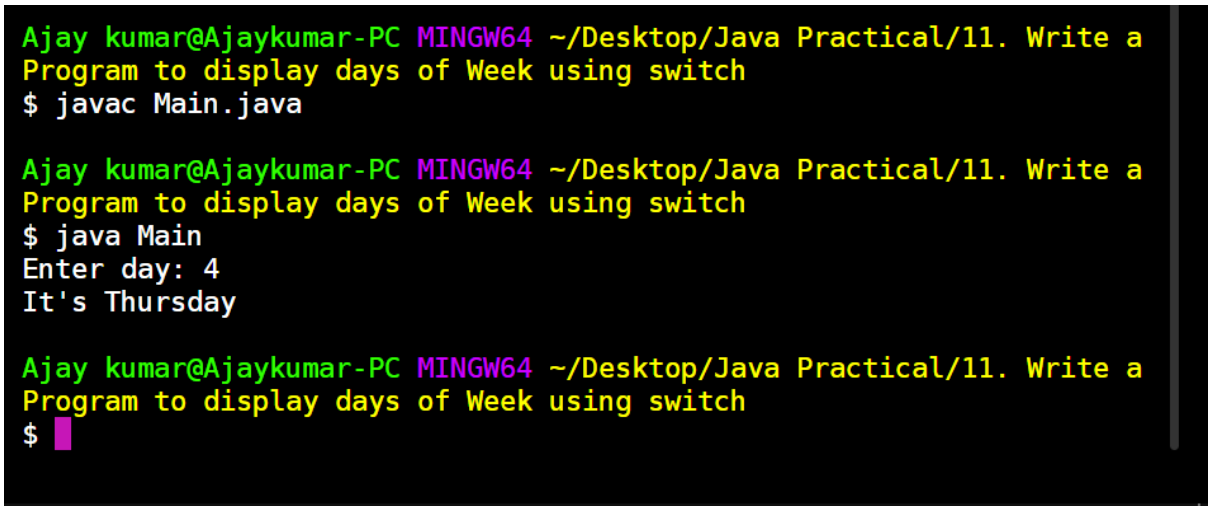
Output:



11. Write a Program to display days of Week using switch

```
import java.util.Scanner;


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

        // Prompt the user to enter a day represented as an integer
        System.out.print("Enter day: ");
        int day = sc.nextInt();

        // Use a switch statement to determine the corresponding day of
the week based on the user's input
        switch (day) {
            case 1:
                System.out.println("It's Monday");
                break;
            case 2:
                System.out.println("It's Tuesday");
                break;
            case 3:
                System.out.println("It's Wednesday");
                break;
            case 4:
                System.out.println("It's Thursday");
                break;
            case 5:
                System.out.println("It's Friday");
                break;
            case 6:
                System.out.println("It's Saturday");
                break;
            case 7:
                System.out.println("It's Sunday");
                break;
            default:
                System.out.println("Invalid number"); // Print this
message for any number other than 1-7
                break;
        }

        // Close the scanner to release resources
        sc.close();
    }
}
```

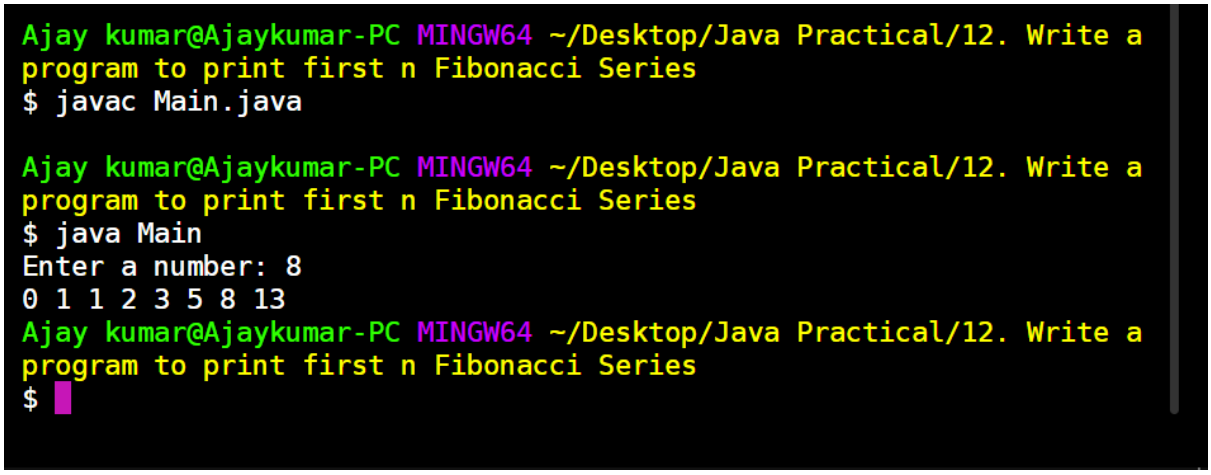
Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/11. Write a
Program to display days of Week using switch
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/11. Write a
Program to display days of Week using switch
$ java Main
Enter day: 4
It's Thursday
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/11. Write a
Program to display days of Week using switch
$

12. Write a program to print first n Fibonacci Series

```
import java.util.Scanner;


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

        // Prompt the user to enter a number
        System.out.print("Enter a number: ");
        int num = sc.nextInt();

        if (num <= 0) {
            // If the number is less than or equal to 0, do nothing (no
output)
        } else if (num == 1) {
            // If the number is 1, print "0"
            System.out.print("0");
        } else if (num == 2) {
            // If the number is 2, print "0 1"
            System.out.print("0 1");
        } else {
            // For numbers greater than 2, generate a Fibonacci
sequence
            int a = 0, b = 1, temp;
            System.out.print("0 1 "); // Initial values of the
Fibonacci sequence
            for (int i = 0; i < num - 2; i++) {
                temp = b;
                b = a + b;
                a = temp;
                System.out.print(b + " "); // Print the next value in
the sequence
            }
        }

        // Close the scanner to release resources
        sc.close();
    }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/12. Write a program to print first n Fibonacci Series
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/12. Write a program to print first n Fibonacci Series
$ java Main
Enter a number: 8
0 1 1 2 3 5 8 13
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/12. Write a program to print first n Fibonacci Series
$

13. Write a program to check number is Armstrong Number

```java
import java.util.Scanner;

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

        // Prompt the user to enter a number
        System.out.print("Enter the number: ");
        int num = sc.nextInt();

        int temp = num;
        int save = 0;
        int a;

        // Calculate the sum of cubes of individual digits in the
number
        while (temp != 0) {
            a = temp % 10;     // Get the last digit
            save += a * a * a; // Add the cube of the digit to the sum
            temp /= 10;        // Remove the last digit
        }

        // Check if the sum of cubes is equal to the original number
        if (save == num) {
            System.out.println("Armstrong");
        } else {
            System.out.println("Not Armstrong");
        }
    }
}
```

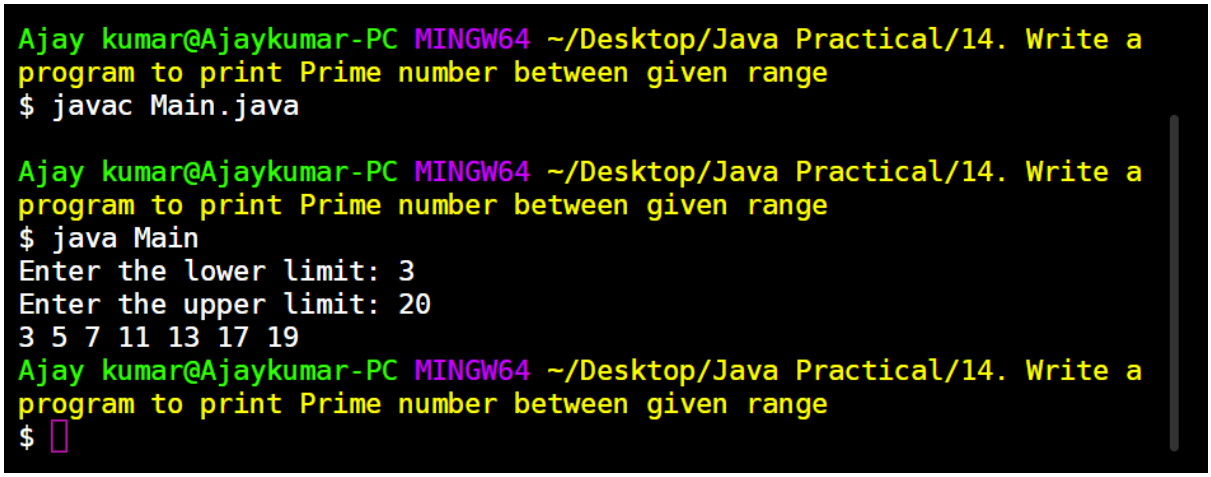
Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/13. Write a
program to check number is Armstrong Number
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/13. Write a
program to check number is Armstrong Number
$ java Main
Enter the number: 153
Armstrong

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/13. Write a
program to check number is Armstrong Number
$
```

14. Write a program to print Prime number between given range

```
import java.util.Scanner;

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

        // Prompt the user to enter the lower limit
        System.out.print("Enter the lower limit: ");
        int l = sc.nextInt();

        // Prompt the user to enter the upper limit
        System.out.print("Enter the upper limit: ");
        int u = sc.nextInt();

        int flag;

        // Iterate through the numbers from the lower limit to the
upper limit
        for (int i = l; i <= u; i++) {
            flag = 0;

            // Check for prime numbers using a nested loop
            for (int j = 2; j <= i / 2 && flag == 0; j++) {
                if (i % j == 0) {
                    flag = 1;
                    // If the number is divisible by any number from 2
to half of itself, it's not prime, set flag to 1
                }
            }

            // If flag is 0, the number is prime, so print it
            if (flag == 0) {
                System.out.print(i + " ");
            }
        }
    }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/14. Write a program to print Prime number between given range
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/14. Write a program to print Prime number between given range
$ java Main
Enter the lower limit: 3
Enter the upper limit: 20
3 5 7 11 13 17 19
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/14. Write a program to print Prime number between given range
$

15. Write a program to calculate area of given object(use method overloading)

```
import java.util.Scanner;

// Define a class called Area
class Area {
    // Method to calculate and return the area of a circle
    double calculateArea(double radius) {
        double area = 3.14 * radius * radius;
        return area;
    }

    // Method to calculate and return the area of a rectangle
    int calculateArea(int length, int breadth) {
        int area = length * breadth;
        return area;
    }

    // Method to calculate and return the area of a triangle
    double calculateArea(double height, double base) {
        double area = (base * height) / 2;
        return area;
    }
}

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

        // Create an instance of the Area class
        Area a = new Area();

        // Prompt the user to choose a shape to calculate the area for
        System.out.println("1. Circle\n2. Rectangle\n3.
Triangle\nCalculate area of: ");
        int choice = sc.nextInt();

        switch (choice) {
            case 1:
                // For a circle, input the radius and calculate the
area
                System.out.print("Enter the radius: ");
                double r = sc.nextDouble();

                double circleArea = a.calculateArea(r);
```

```
                System.out.println("Area = " + circleArea);
                break;

            case 2:
                // For a rectangle, input the length and breadth and
calculate the area
                System.out.print("Enter the length: ");
                int l = sc.nextInt();
                System.out.print("Enter the breadth: ");
                int b = sc.nextInt();

                int rectArea = a.calculateArea(l, b);
                System.out.println("Area = " + rectArea);
                break;

            case 3:
                // For a triangle, input the height and base and
calculate the area
                System.out.print("Enter the height: ");
                double h = sc.nextDouble();
                System.out.print("Enter the base: ");
                double base = sc.nextDouble();

                double triArea = a.calculateArea(h, base);
                System.out.println("Area = " + triArea);
                break;

            default:
                System.out.println("Invalid Operation"); // Print this
message for an invalid choice
                break;
        }

        // Close the scanner to release resources
        sc.close();
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/15. Write a
program to calculate area of given object(use method overloading)
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/15. Write a
program to calculate area of given object(use method overloading)
$ java Main
1. Circle
2. Rectangle
3. Triangle
Calculate area of:
3
Enter the height: 2
Enter the base: 4
Area = 4.0

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/15. Write a
program to calculate area of given object(use method overloading)
$
```

16. Write a program to demonstrate concept of constructor chaining

```
// Define a Student class
class Student {
    int pid, age;
    String name;

    // Constructor with three parameters to initialize PID, name, and
age
    public Student(int sPid, String sName, int sAge) {
        pid = sPid;
        name = sName;
        age = sAge;
    }

    // Constructor with only PID and uses default values for name and
age
    public Student(int sPid) {
        this(sPid, "noName", 19);
    }

    // Constructor with PID and name, and uses a default age
    public Student(int sPid, String sName) {
        this(sPid, sName, 19);
    }

    // Constructor with PID and age, and uses a default name
    public Student(int sPid, int sAge) {
        this(sPid, "noName", sAge);
    }

    // Method to print the details of a student
    void getDetails() {
        System.out.println("PID: " + pid);
        System.out.println("Name: " + name);
        System.out.println("Age: " + age);
        System.out.println();
    }
}

class Main {
    public static void main(String[] args) {
        // Create instances of the Student class with different
constructors
        Student stu1 = new Student(221077, "Ajaykumar", 20);
        stu1.getDetails();
```

```
        Student stu2 = new Student(221078, "Kevin");
        stu2.getDetails();

        Student stu3 = new Student(221080, 18);
        stu3.getDetails();

        Student stu4 = new Student(201010);
        stu4.getDetails();
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/16. Write a
program to demonstrate concept of constructor chaining
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/16. Write a
program to demonstrate concept of constructor chaining
$ java Main
PID: 221077
Name: Ajaykumar
Age: 20

PID: 221078
Name: Kevin
Age: 19

PID: 221080
Name: noName
Age: 18

PID: 201010
Name: noName
Age: 19
```

17. Write a program to print the names of students by creating a Student class

```
class Student {
    String name;

    // Constructor with a parameter to set the name
    public Student(String n) {
        name = n;
    }

    // Default constructor with no parameters to set the name to
"Unknown"
    public Student() {
        name = "Unknown";
    }

    // Method to print the name of the student
    void getName() {
        System.out.println("Name: " + name);
    }
}

class Main {
    public static void main(String[] args) {
        // Create instances of the Student class with and without
passing a name
        Student stu1 = new Student("Ajaykumar");
        stu1.getName();

        Student stu2 = new Student();
        stu2.getName();
    }
}
```

Output:

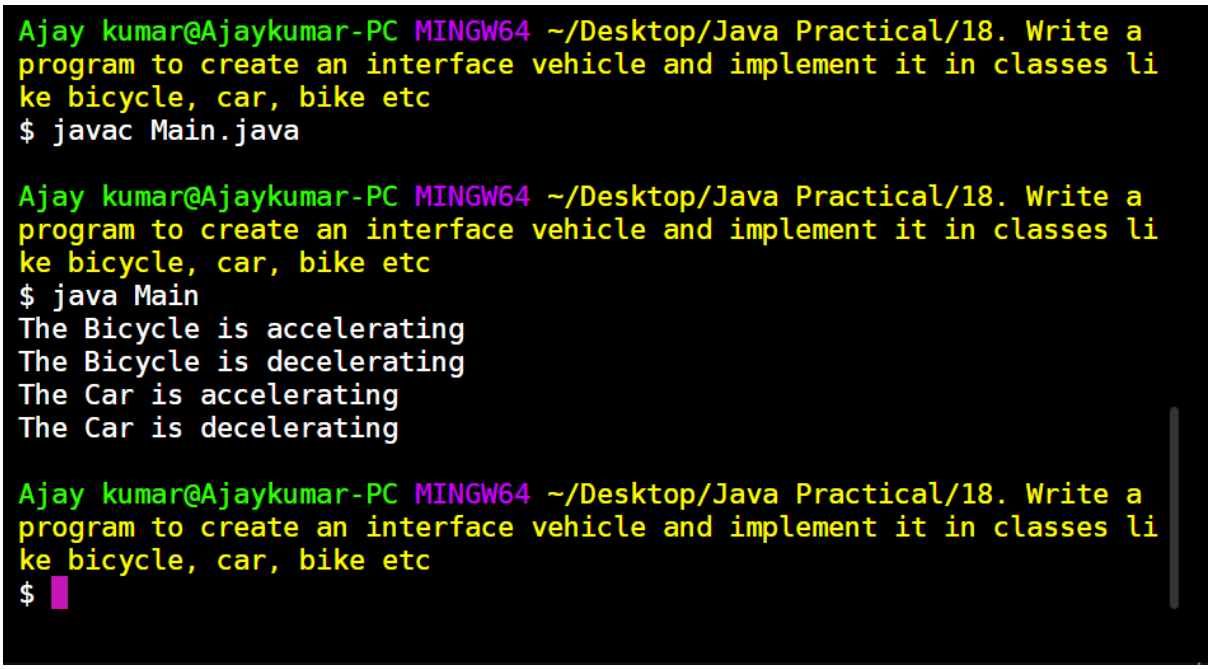
```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/17. Write a
program to print the names of students by creating a Student class
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/17. Write a
program to print the names of students by creating a Student class
$ java Main
Name: Ajaykumar
Name: Unknown

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/17. Write a
program to print the names of students by creating a Student class
$
```

18. Write a program to create an interface vehicle and implement it in classes like bicycle, car, bike etc

```
// Define an interface called Vehicle with accelerate and decelerate
methods
interface Vehicle {
    void accelerate();
    void decelerate();
}

// Define a class Bicycle that implements the Vehicle interface
class Bicycle implements Vehicle {
    int noOfTyres = 2;
    int noOfPassengers = 1;
    String color = "red";

    // Implement the accelerate method for the Bicycle
    public void accelerate() {
        System.out.println("The Bicycle is accelerating");
    }

    // Implement the decelerate method for the Bicycle
    public void decelerate() {
        System.out.println("The Bicycle is decelerating");
    }
}

// Define a class Car that implements the Vehicle interface
class Car implements Vehicle {
    int noOfTyres = 4;
    int noOfPassengers = 5;
    String color = "black";

    // Implement the accelerate method for the Car
    public void accelerate() {
        System.out.println("The Car is accelerating");
    }

    // Implement the decelerate method for the Car
    public void decelerate() {
        System.out.println("The Car is decelerating");
    }
}

class Main {
    public static void main(String args[]) {
```

```
        // Create an instance of Bicycle
        Bicycle b = new Bicycle();

        // Call accelerate and decelerate methods for Bicycle
        b.accelerate();
        b.decelerate();

        // Create an instance of Car
        Car c = new Car();

        // Call accelerate and decelerate methods for Car
        c.accelerate();
        c.decelerate();
    }
}
```

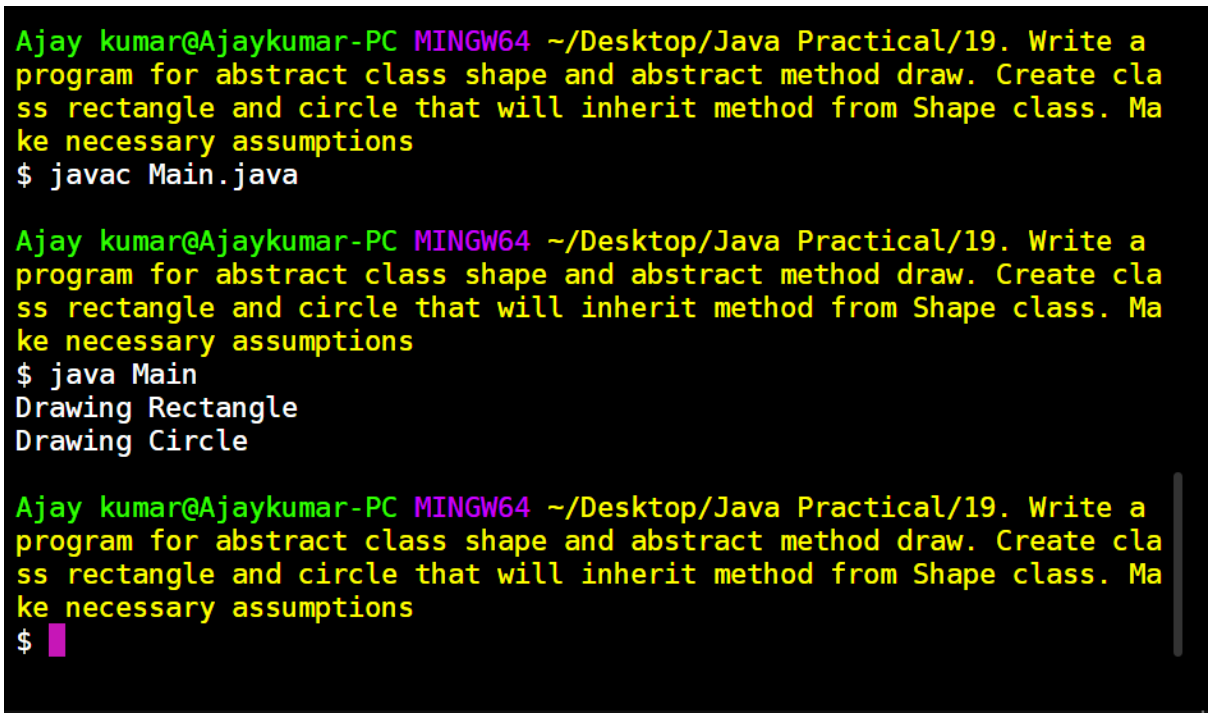
Output:



19. Write a program for abstract class shape and abstract method draw. Create class rectangle and circle that will inherit method from Shape class. Make necessary assumptions

```
// Define an abstract class called Shape
abstract class Shape {
    // Declare an abstract method called draw (to be implemented by
subclasses)
    abstract void draw();
}

// Define a subclass Rectangle that extends the Shape class
class Rectangle extends Shape {
    // Implement the draw method for Rectangle
    void draw() {
        System.out.println("Drawing Rectangle");
    }
}

// Define a subclass Circle that extends the Shape class
class Circle extends Shape {
    // Implement the draw method for Circle
    void draw() {
        System.out.println("Drawing Circle");
    }
}

class Main {
    public static void main(String[] args) {
        // Create an instance of Rectangle
        Rectangle r = new Rectangle();

        // Call the draw method for Rectangle
        r.draw();

        // Create an instance of Circle
        Circle c = new Circle();

        // Call the draw method for Circle
        c.draw();
    }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/19. Write a program for abstract class shape and abstract method draw. Create class rectangle and circle that will inherit method from Shape class. Make necessary assumptions
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/19. Write a program for abstract class shape and abstract method draw. Create class rectangle and circle that will inherit method from Shape class. Make necessary assumptions
$ java Main
Drawing Rectangle
Drawing Circle
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/19. Write a program for abstract class shape and abstract method draw. Create class rectangle and circle that will inherit method from Shape class. Make necessary assumptions
$

20. Write a program that creates a user defined package and import that package in another package

```java
// Define a class named "Main" in the "myPackage" package
package myPackage;

public class Main {
    public void print() {
        System.out.println("This is myPackage");
    }
}
```

```java
// Import the "Main" class from the "myPackage" package
package mainPackage;
import myPackage.Main;

// Define another class named "Main2" in the "mainPackage" package
class Main2 {
    public static void main(String[] args) {
        // Create an instance of the "Main" class from the "myPackage"
package
        Main m = new Main();

        // Call the "print" method of the "Main" class
        m.print();
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/20. Write a
program that creates a user defined package and import that package i
n another package
$ javac -d . Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/20. Write a
program that creates a user defined package and import that package i
n another package
$ javac -d . Main2.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/20. Write a
program that creates a user defined package and import that package i
n another package
$ java mainPackage.Main2
This is myPackage

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/20. Write a
program that creates a user defined package and import that package i
n another package
$
```

21.Write a program to implement multiple inheritance

```java
// Define an interface named "Mother" with a constant "colorOfEyes"
interface Mother {
    String colorOfEyes = "black";
}

// Define an interface named "Father" with a constant "money"
interface Father {
    String money = "10000";
}

// Define a class "Son" that implements both "Mother" and "Father"
interfaces
class Son implements Mother, Father {
    void print() {
        // Access and print the constants from the interfaces
        System.out.println("Color of Eyes: " + colorOfEyes);
        System.out.println("Money: " + money);
    }
}

// Define the main class
class Main {
    public static void main(String[] args) {
        // Create an instance of the "Son" class
        Son s = new Son();

        // Call the "print" method to access and display the constants
        s.print();
    }
}
```

Output:

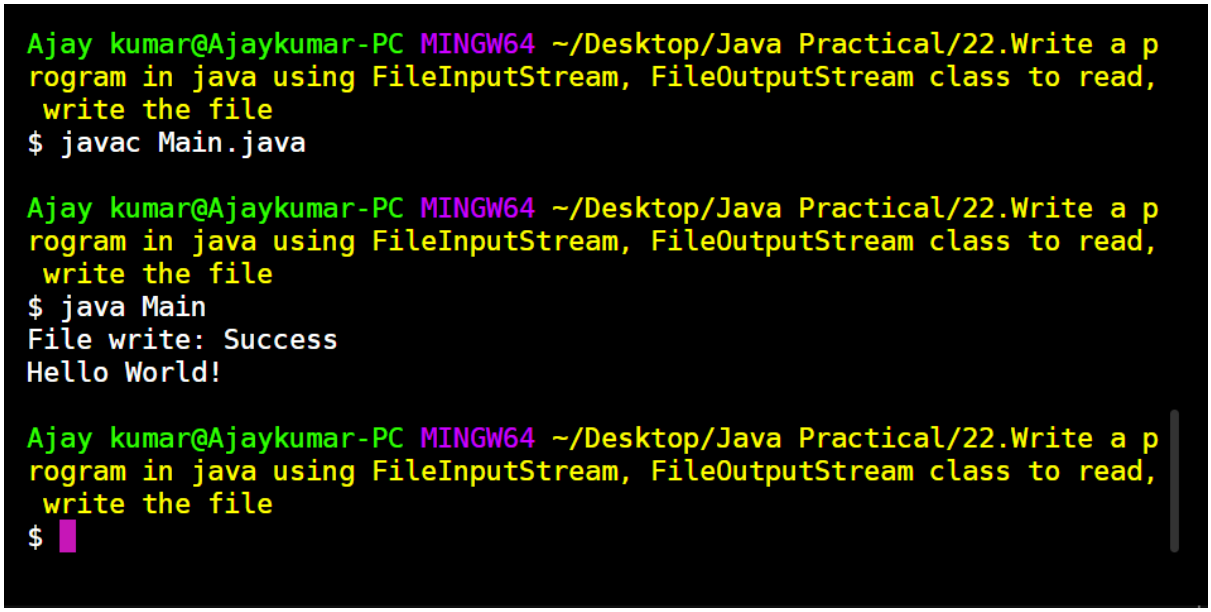
```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/21.Write a p
rogram to implement multiple inheritance
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/21.Write a p
rogram to implement multiple inheritance
$ java Main
Color of Eyes: black
Money: 10000

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/21.Write a p
rogram to implement multiple inheritance
$
```

22.Write a program in java using FileInputStream, FileOutputStream class to read, write the file

```
import java.io.FileOutputStream;
import java.io.FileInputStream;

public class Main {
  public static void main(String args[]) {
    try {
      // Create a string to write to the file
      String s = "Hello World!\n";
      byte b[] = s.getBytes();

      // Create a FileOutputStream to write to a file named
"testout2.txt"
      FileOutputStream fout = new FileOutputStream("testout2.txt");

      // Write the byte array containing the string to the file
      fout.write(b);

      // Close the FileOutputStream
      fout.close();

      System.out.println("File write: Success");
    } catch (Exception e) {
      // Handle exceptions, if any
      System.out.println("Error during file write: " + e);
    }

    try {
      // Create a FileInputStream to read from the file "testout2.txt"
      FileInputStream fin = new FileInputStream("testout2.txt");
      int i = 0;

      // Read and print each character from the file until the end of
file is reached
      while ((i = fin.read()) != -1) {
        System.out.print((char) i);
      }

      // Close the FileInputStream
      fin.close();
    } catch (Exception e) {
      // Handle exceptions, if any
      System.out.println("Error during file read: " + e);
    }
```

```
  }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/22.Write a program in java using FileInputStream, FileOutputStream class to read, write the file
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/22.Write a program in java using FileInputStream, FileOutputStream class to read, write the file
$ java Main
File write: Success
Hello World!
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/22.Write a program in java using FileInputStream, FileOutputStream class to read, write the file
$

25. Write a program to demonstrate checked Exception Handling using nested try, catch statements

```
import java.lang.*;

// Define the main class
class Main {
    public static void main(String args[]) {
        try {
            // Attempt a division by zero, which will result in an
ArithmeticException
            int i = 10/0;

        } catch (ArithmeticException e) {
            // Catch the ArithmeticException and print the exception
details
            System.out.println("Caught ArithmeticException: " + e);

            // Attempt to access an element at an out-of-bounds index,
which will result in an IndexOutOfBoundsException
            try {
                int[] num = {0, 1, 2};
                int b = num[4];
            } catch (IndexOutOfBoundsException f) {
                // Catch the IndexOutOfBoundsException and print the
exception details
                System.out.println("Caught IndexOutOfBoundsException: "
+ f);
            }
        }
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/25. Write a
program to demonstrate checked Exception Handling using nested try, c
atch statements
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/25. Write a
program to demonstrate checked Exception Handling using nested try, c
atch statements
$ java Main
Caught ArithmeticException: java.lang.ArithmeticException: / by zero
Caught IndexOutOfBoundsException: java.lang.ArrayIndexOutOfBoundsExce
ption: Index 4 out of bounds for length 3

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/25. Write a
program to demonstrate checked Exception Handling using nested try, c
atch statements
$
```

26. Write a program that validate username and password using user defined exception

```
import java.lang.Exception;
import java.util.Scanner;

// Define a custom exception for a username not found situation
class UsernameNotFoundException extends Exception {
    UsernameNotFoundException(String message) {
        super(message);
    }
}

// Define a custom exception for an incorrect password situation
class IncorrectPasswordException extends Exception {
    IncorrectPasswordException(String message) {
        super(message);
    }
}

// Define a User class to represent a user with a username and password
class User {
    protected String username;
    protected String password;

    // Constructor to initialize the username and password
    User(String user, String pass) {
        username = user;
        password = pass;
    }

    // Method to authorize a user by comparing the provided username
and password
    void authorize(String u, String p) {
        try {
            if (u.equals(username)) { // Check if the provided username
matches
                try {
                    if (p.equals(password)) { // Check if the provided
password matches
                        System.out.println("Authorization Successful");
                    } else {
                        throw new IncorrectPasswordException("Password
doesn't match");
                    }
                } catch (IncorrectPasswordException f) {
```

```
                    System.out.println(f);
                }
            } else {
                throw new UsernameNotFoundException("Username not
found");
            }
        } catch (UsernameNotFoundException e) {
            System.out.println(e);
        }
    }
}

// Define the main class
class Main {
    public static void main(String args[]) {
        Scanner sc = new Scanner(System.in);

        // Create a User instance with a predefined username and
password
        User u1 = new User("ajaykumar", "1234567");

        System.out.print("Enter username: ");
        String name = sc.nextLine();

        System.out.print("Enter password: ");
        String key = sc.nextLine();

        // Authorize the user by checking the provided username and
password
        u1.authorize(name, key);
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/26. Write a
program that validate username and password using user defined except
ion
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/26. Write a
program that validate username and password using user defined except
ion
$ java Main
Enter username: ajaykumar
Enter password: 1234
IncorrectPasswordException: Password doesn't match

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/26. Write a
program that validate username and password using user defined except
ion
$
```

27. Write a program to create a user defined exception class MarksOutOfBoundsException to validate marks entry

```
import java.util.Scanner;

// Custom exception class for handling marks out of bounds
class MarksOutOfBoundsException extends Exception {
    public MarksOutOfBoundsException(String message) {
        super(message);
    }
}

// Class to store and display Semester 2 results
class Sem2Result {
    String name;
    int seatNo;
    String date;
    int centerNo;
    int marks;

    // Constructor to initialize result details
    Sem2Result(String name, int seatNo, String date, int centerNo, int
marks) {
        this.name = name;
        this.seatNo = seatNo;
        this.date = date;
        this.centerNo = centerNo;
        this.marks = marks;
    }

    // Method to display result details
    void display() {
        System.out.println("Result Details:");
        System.out.println("Name: " + name);
        System.out.println("Seat No: " + seatNo);
        System.out.println("Date: " + date);
        System.out.println("Center Number: " + centerNo);
        System.out.println("Semester 2 Marks: " + marks);
    }
}

public class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the Marks: ");
        int marks = sc.nextInt();
```

```
        try {
            // Create a Sem2Result object with sample details
            Sem2Result studentResult = new Sem2Result("Ajaykumar
Nadar", 123456, "2023-09-22", 001, marks);

            // Check if the marks are within the valid range (0-100)
            if (studentResult.marks < 0 || studentResult.marks > 100) {
                throw new MarksOutOfBoundsException("Marks out of
bounds (0-100): " + studentResult.marks);
            }

            // Display the result
            studentResult.display();
        } catch (MarksOutOfBoundsException e) {
            // Handle the custom exception for out-of-bounds marks
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/27. Write a
program to create a user defined exception class MarksOutOfBoundsExce
ption to validate marks entry
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/27. Write a
program to create a user defined exception class MarksOutOfBoundsExce
ption to validate marks entry
$ java Main
Enter the Marks: 42
Result Details:
Name: Ajaykumar Nadar
Seat No: 123456
Date: 2023-09-22
Center Number: 1
Semester 2 Marks: 42

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/27. Write a
program to create a user defined exception class MarksOutOfBoundsExce
ption to validate marks entry
$
```

30. Create two threads such that one thread will print even number and another will print odd number in an ordered fashion

```
import java.lang.Thread;

// A Counter class to manage counting and printing even or odd numbers
class Counter {
    // Synchronized method to ensure thread safety
    synchronized void count(String type) {
        for (int i = 0; i < 10; i++) {
            if (i % 2 == 0 && type.equals("even")) {
                System.out.println("Even: " + i);
            }
            if (i % 2 == 1 && type.equals("odd")) {
                System.out.println("Odd: " + i);
            }
        }
    }
}

// Define a thread class to print odd numbers
class Odd extends Thread {
    Counter c;

    Odd(Counter c) {
        this.c = c;
    }

    public void run() {
        c.count("odd");
    }
}

// Define a thread class to print even numbers
class Even extends Thread {
    Counter c;

    Even(Counter c) {
        this.c = c;
    }

    public void run() {
        c.count("even");
    }
}
```

```java
// Main class
class Main {
    public static void main(String args[]) {
        Counter c1 = new Counter();
        Odd o = new Odd(c1);
        Even e = new Even(c1);

        o.start(); // Start the Odd thread
        e.start(); // Start the Even thread
    }
}
```

Output:

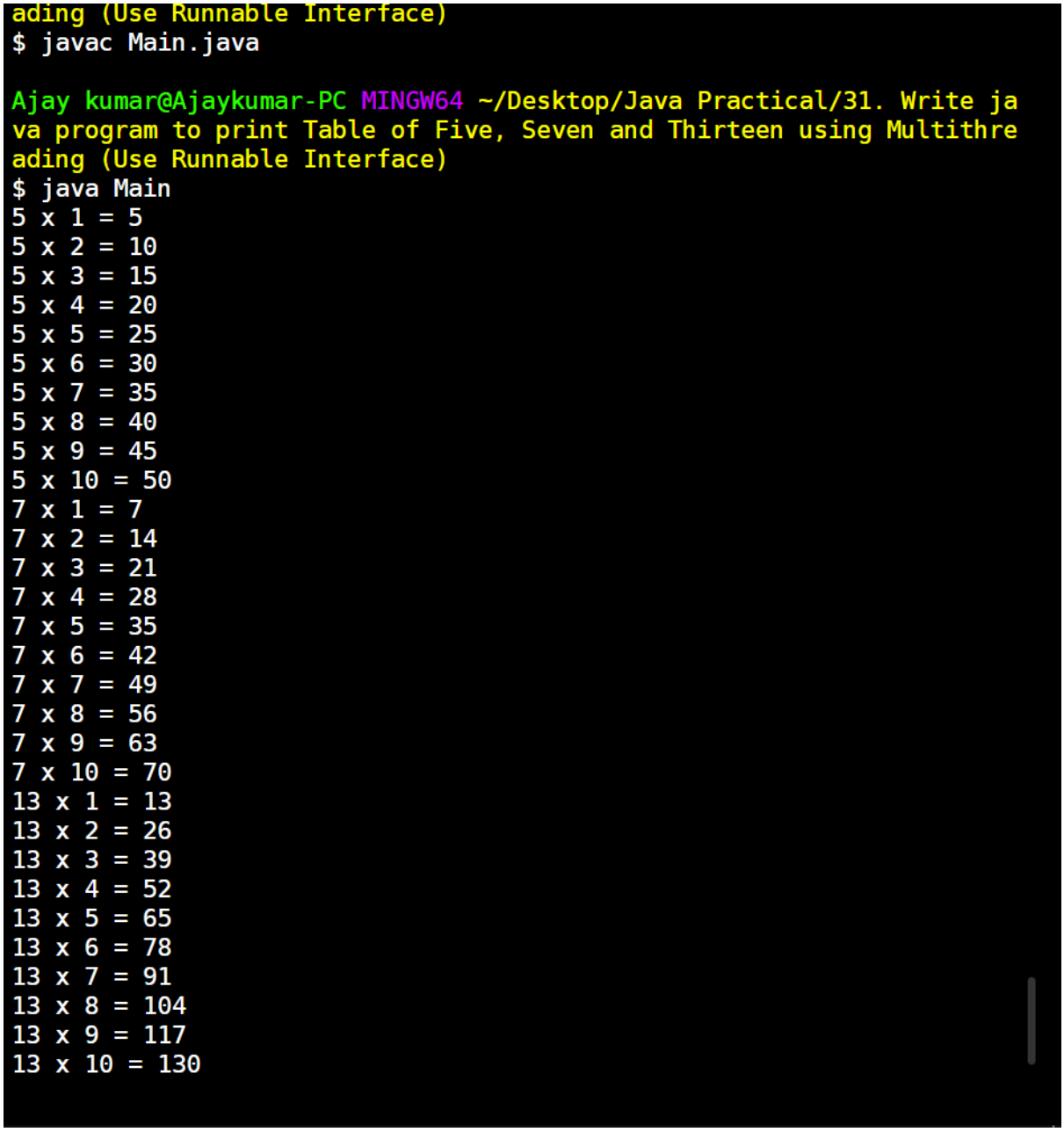
```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/30. Create t
wo threads such that one thread will print even number and another wi
ll print odd number in an ordered fashion
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/30. Create t
wo threads such that one thread will print even number and another wi
ll print odd number in an ordered fashion
$ java Main
Odd: 1
Odd: 3
Odd: 5
Odd: 7
Odd: 9
Even: 0
Even: 2
Even: 4
Even: 6
Even: 8

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/30. Create t
wo threads such that one thread will print even number and another wi
ll print odd number in an ordered fashion
$
```

31. Write java program to print Table of Five, Seven and Thirteen using Multithreading (Use Runnable Interface)

```
// Define a class "Five" that implements the Runnable interface
class Five implements Runnable {
    public void run() {
        // Loop to print the multiplication table of 5
        for (int i = 1, j = 5; i <= 10; i++, j += 5) {
            System.out.println("5 x " + i + " = " + j);
        }
    }
}

// Define a class "Seven" that implements the Runnable interface
class Seven implements Runnable {
    public void run() {
        // Loop to print the multiplication table of 7
        for (int i = 1, j = 7; i <= 10; i++, j += 7) {
            System.out.println("7 x " + i + " = " + j);
        }
    }
}

// Define a class "Thirteen" that implements the Runnable interface
class Thirteen implements Runnable {
    public void run() {
        // Loop to print the multiplication table of 13
        for (int i = 1, j = 13; i <= 10; i++, j += 13) {
            System.out.println("13 x " + i + " = " + j);
        }
    }
}

// Define the main class
class Main {
    public static void main(String args[]) {
        // Create instances of the "Five," "Seven," and "Thirteen"
classes
        Five five = new Five();
        Seven seven = new Seven();
        Thirteen thirteen = new Thirteen();

        // Create threads for each multiplication table
        Thread t1 = new Thread(five);
        Thread t2 = new Thread(seven);
        Thread t3 = new Thread(thirteen);
```

```
        // Start the threads using the run method (should use start
instead)
        t1.run();
        t2.run();
        t3.run();
    }
}
```

Output:



33.Write a program calling any 5 methods of vector class

```java
import java.util.Vector;


class Main {
    public static void main(String[] args) {
        // Create a Vector of strings
        Vector<String> vec = new Vector<>();

        // 1. Add elements
        vec.add("Ajaykumar");
        vec.add("Bianca");
        vec.add("Kevin");

        // 2. Accessing elements
        System.out.println("Element at index 2: " + vec.get(2));

        // 3. Changing elements
        vec.set(2, "Subhodeep");
        System.out.println("Element at index 2: " + vec.get(2));

        // 4. Remove element at index 1
        vec.remove(1);
        System.out.println("Element at index 1: " + vec.get(1));

        // 5. Check size
        System.out.println("Vector size: " + vec.size());

        // 6. Check if empty
        System.out.println("Is vector Empty?: " + vec.isEmpty());
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/33.Write a p
rogram calling any 5 methods of vector class
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/33.Write a p
rogram calling any 5 methods of vector class
$ java Main
Element at index 2: Kevin
Element at index 2: Subhodeep
Element at index 1: Subhodeep
Vector size: 2
Is vector Empty?: false

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/33.Write a p
rogram calling any 5 methods of vector class
$
```

34.Write a program where thread sleeps for 500 ms

```
import java.lang.Thread;

// Define a class A
class A {
    // Constructor for class A
    A() {
        // Loop from 0 to 9
        for (int i = 0; i < 10; i++) {
            System.out.println(i); // Print the current value of i
            try {
                Thread.sleep(500); // Pause execution for 500
milliseconds (0.5 seconds)
            } catch (Exception e) {
                System.out.println(e); // Handle any exceptions that
may occur during sleep
            }
        }
    }
}

// Define the main class
class Main {
    public static void main(String args[]) {
        A a = new A(); // Create an instance of class A
    }
}
```

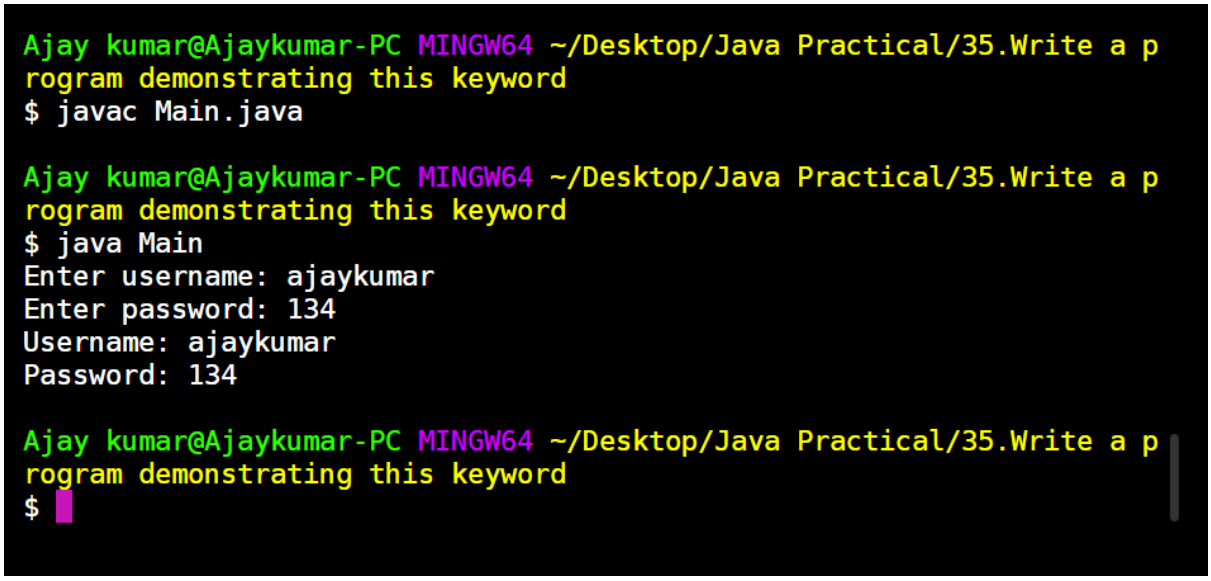
Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/34.Write a p
rogram where thread sleeps for 500 ms
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/34.Write a p
rogram where thread sleeps for 500 ms
$ java Main
0
1
2
3
4
5
6
7
8
9

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/34.Write a p
rogram where thread sleeps for 500 ms
$
```

35.Write a program demonstrating this keyword

```java
import java.util.Scanner;

// Define a User class
class User {
    protected String name, password;

    // Constructor to initialize user details
    User(String name, String password) {
        this.name = name;
        this.password = password;
    }

    // Method to get and print user details
    void getDetails() {
        System.out.println("Username: " + name);
        System.out.println("Password: " + password);
    }
}

// Main class
class Main {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        // Prompt the user to enter a username
        System.out.print("Enter username: ");
        String name = sc.nextLine();

        // Prompt the user to enter a password
        System.out.print("Enter password: ");
        String password = sc.nextLine();

        // Create a User object with the provided username and password
        User u1 = new User(name, password);

        // Display the user details
        u1.getDetails();
    }
}
```

Output:


Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/35.Write a p
rogram demonstrating this keyword
$ javac Main.java
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/35.Write a p
rogram demonstrating this keyword
$ java Main
Enter username: ajaykumar
Enter password: 134
Username: ajaykumar
Password: 134
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/35.Write a p
rogram demonstrating this keyword
$

36.Write a program demonstrating super keyword

```java
// Define a User class with username and password
class User {
    protected String name, password;

    // Constructor to initialize user details
    User(String name, String password) {
        this.name = name;
        this.password = password;
    }

    // Method to get and print user details
    void getData() {
        System.out.println("Username: " + name);
        System.out.println("Password: " + password);
    }
}

// Define an Admin class that extends User
class Admin extends User {
    // Constructor to initialize admin details
    Admin(String name, String password) {
        super(name, password);
    }
}

// Main class
class Main {
    public static void main(String[] args) {
        // Create an Admin object with a username and password
        Admin admin1 = new Admin("ajaykumar", "1234");

        // Display the admin's user details using the inherited method
        admin1.getData();
    }
}
```

Output:

```
Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/36.Write a p
rogram demonstrating super keyword
$ javac Main.java

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/36.Write a p
rogram demonstrating super keyword
$ java Main
Username: ajaykumar
Password: 1234

Ajay kumar@Ajaykumar-PC MINGW64 ~/Desktop/Java Practical/36.Write a p
rogram demonstrating super keyword
$
```