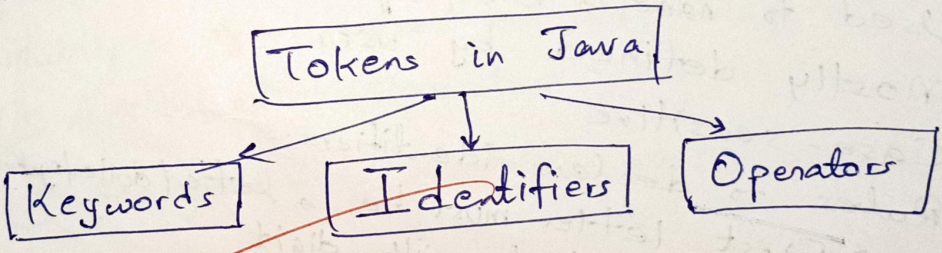
5/12 Object Oriented Programming in JAVA

OOP Principles

1. Abstraction → Shows the necessary data & hinds the unnecessary data
2. Encapsulation → Wrapping up of data and methods Also provides security
3. Inheritence → Inheritence of properties from one class to another
4. Polymorphism (poly = many, morphism = forms) → To be available/present in many forms

Tokens in Java

- Programs in Java, contain classes and methods.
- These methods contain statements and expressions.
- The statements and expressions are made up of tokens.



Types of Tokens in Java

- Keywords
- Identifiers
- Literals
- Operators
- Seperators
- Comments

Q1

Assignment -1

① Keywords

→ Pre-defined reserved words
→ Have a special meaning
→ Used for a specific purpose
→ Always written in lower case

eg: do
for
int
if
float
void
public
while

② Identifiers

→ Used to name a variable, function, constant, etc
→ Mostly defined by user.
→ Case-sensitive
→ Rules to declare identifiers:-
  → First letter must be a letter/ dollar/symbol.
  → Mustnot start with digit
  → May contain digits.
  → Whitespaces cannot be included.

eg: a
age
num
:x
Sum
count

③ Literals

→ Notation that represents a fixed value (const.)
→ Categorised as : integer literal,
string literal,
boolean literal.

→ Defined by ~~user~~ programmer

eg: integer
floating point
character
string
boolean

④ Operators

→ Special symbols that tells the compiler to perform a special operation.
→ Multiple types based on usage (or) purpose:

① Arithmetic Operators eg: =
② Assignment Operators eg: +, −, /, *, %
③ Relational Operators eg: ==, !=, >, <
④ Logical Operators eg: &&, ||
⑤ Unary Operators eg: ++, −−, !
⑥ ~~Binary~~ Bitwise operators eg: &, |, ^, ~
⑦ Ternary operators eg: (....)?(...):(...)
⑧ Shift operators eg: >>, <<, >>>

⑤ Seperators

→ Help defining structure of program

→ Known as punctuators

eg: Square brackets [ ]

Paranthesis: ( )

Curly braces: { }

Comma: ,

Period: .

Assignment operator: =

⑥ Comment

→ Help in user to understand the code

→ Doesn't be a part of program when run

→ Used in specify information inside the code

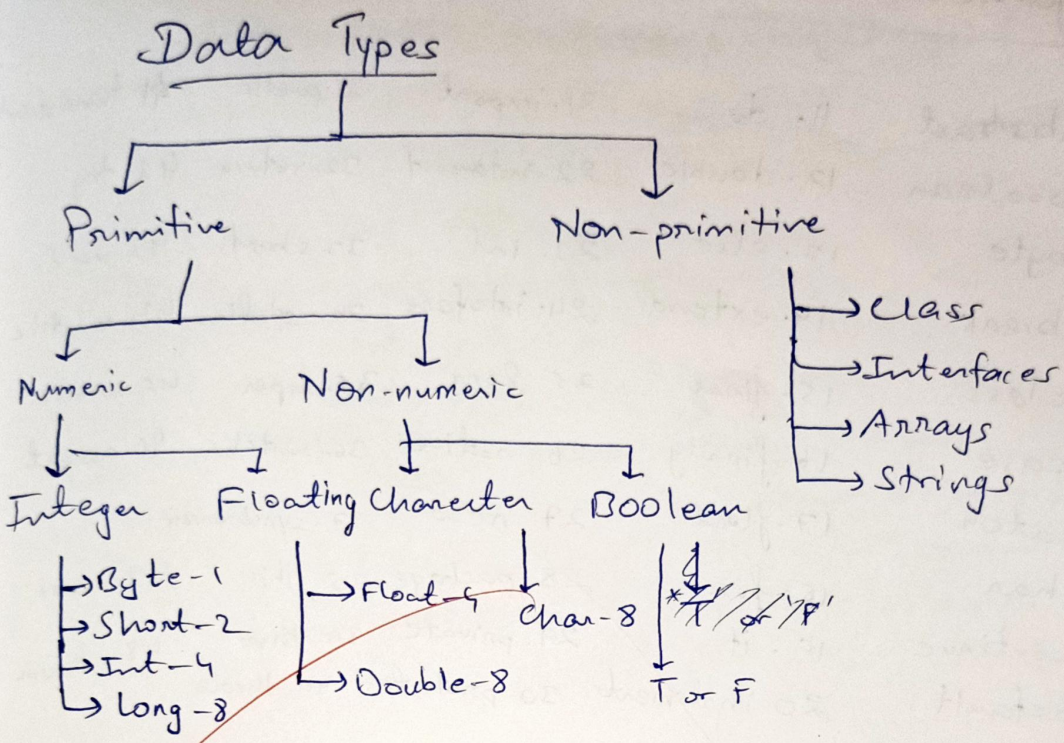
→ 2 types:

Single-line comments : //

Multi-line comments : /* ..... */

12/12

Data Types



PROGRAM TO ADD TWO NUMBERS

```
public class Add {
    public static void main(String[] args) {
        int a=10,b=5, c;
        c=a+b;
        System.out.println("Sum = "+c);
    }
}
```

Q1.

Keywords in Java

1. abstract
2. boolean
3. byte
4. break
5. class
6. case
7. catch
8. char
9. continue
10. default
11. do
12. double
13. else
14. extends
15. final
16. finally
17. float
18. for
19. if
20. implements
21. import
22. instanceof
23. int
24. interface
25. long
26. native
27. new
28. package
29. private
30. protected
31. public
32. return
33. short
34. static
35. super
36. switch
37. synchronized
38. this
39. thro
40. throws
41. transistent
42. try
43. void
44. volatile
45. while
46. assert
47. const
48. enum
49. goto
50. strictfp

Q2. List out all the operators & execute programs.

PROGRAMS

Operators

(1) Arithmetic Operators

1. Addition (+)
2. Subtraction (-)
3. Multiplication (*)
4. Divide (/)
5. Modulo (%)

```
public class Arith_Op {
    public static void main(String[] args) {
        int a = 15, b = 5;
        int sum = a+b;
        int diff = a-b;
        int product = a*b;
        int div = a/b;
        int rem = a%b;
        System.out.println("Sum is " + sum);
        System.out.println("Difference is " + diff);
        System.out.println("Product is " + product);
        System.out.println("Quotient is " + div);
        System.out.println("Remainder is " + rem);
    }
}
```

O/P

```
Sum is 20
Difference is 10
Product is 75
Quotient is 3
Remainder is 0
```

(2) Relational Operators

1. Equal to (==)
2. Not equal to (!=)
3. Less than (<)
4. Greater than (>)
5. Less than or equal to (<=)
6. Greater than or equal to (>=)

```
public class Relation_Op {

    public static void main(String[] args) {

        int a = 15;
        int b = 5;

        System.out.println("15 == 5 is " + (a == b));
        System.out.println("15 != 5 is " + (a != b));
        System.out.println("15 < 5 is " + (a < b));
        System.out.println("15 > 5 is " + (a > b));
        System.out.println("15 <= 5 is " + (a <= b));
        System.out.println("15 >= 5 is " + (a >= b));

    }

}
```

OP

15 == 5 is false
15 != 5 is true
15 < 5 is false
15 > 5 is true
15 <= 5 is false
15 >= 5 is true.

(3) Logical Operators

1. Logical AND (&&)
2. Logical OR (||)
3. Logical NOT (!)

```
public class Logical_Op {
    public static void main(String[] args) {
        int a = 15;
        int b = 5;
        System.out.println("(5 > 10) && (5 < 10) is
        int c = ((a > 10) && (b < 10));
        int d = ((a > 10) || (b < 10)));
        int e = !(a > b);
        System.out.println("(15 > 10) && (5 < 10) is " + c);
        System.out.println("(15 > 10) || (5 < 10) is " + d);
        System.out.println("!(15 > 5) is " + e);
    }
}
```

O/P

(15 > 10) && (5 < 10) is true

(15 > 10) || (5 < 10) is true

!(15 > 5) is false

(4) ~~Bitwise Operators~~

(4) Bitwise Operators

1. Bitwise AND
2. Bitwise OR
3. Bitwise XOR
4. Bitwise Complement.

```
public class Bitwise_Op {

    public static void main(String[] args) {

        int a=15;
        int b= 5;

        int AND = a & b;
        int OR = a | b;
        int XOR = a ^ b;
        int NOT = ~a ;

        System.out.println("a & b is "+AND);
        System.out.println("a | b is "+OR);
        System.out.println("a ^ b is "+XOR);
        System.out.println("~a is " + NOT);

    }

}
```

O/P

```
a & b  is  5
a | b  is  15
a ^ b  is  10
 ~a    is  -16
```

(5) Shift Operators

1. (<<) left shift
2. (>>) Right Shift

```
public class Shift_Op {
    public static void main(String [] args) {
        int a = 10;
        int b = a << 1;
        int c = a >> 1;
        System.out.println("a<<1 is "+ b);
        System.out.println("a>>1 is "+ c);
    }
}
```

Op

a<<1 is 20

a>>1 is 5

(6) Unary Operators

1. Unary plus (+a)
2. Unary minus (-a)
3. Pre-increment (++a)
4. Post-increment (a++)
5. Pre-decrement (--a)
6. (Post-decrement) (a--)

```
public class Unary_Op {

    public static void main(String[] args) {

        int a=15;
        int b=+a;  // Unary plus
        int c=-a;  // Unary minus
        int d=++a; // Pre-increment
        int e=a++; // Post-increment
        int f=--a; // Pre-decrement
        int g=a--; // Post-decrement.

        System.out.println("+a is "+ b);
        System.out.println("-a is "+ c);
        System.out.println("++a is "+ d);
        System.out.println("a++ is "+ e);
        System.out.println("--a is "+ f);
        System.out.println("a-- is "+ g);

    }

}
```

O/P

```
+a is 15
-a is -15
++a is 16
a++ is 16
--a is 15
a-- is 15
```

(7) Ternary operators

Ternary operator (_?_ : _)

```
public class ter_op {
    public static void main(String[] args) {
        int a = 15;
        int b = 5;
        int max = (a > b) ? a : b;
        System.out.println("Max of a and b is " + max);
    }
}
```

O/P

Max of a and b is 15.

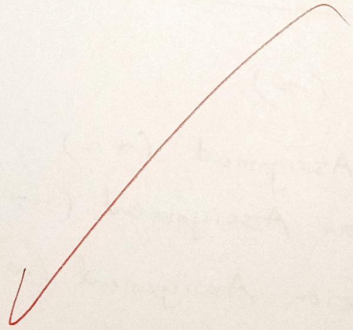
(8) Assignment Operators

1. Assignment (=)
2. Addition Assignment (+=)
3. Subtraction Assignment (-=)
4. Multiplication Assignment (*=)
5. Division Assignment (/=)
6. Modulus Assignment (%=)

```
public class Assign_Op {
    public static void main(String [] args) {
        int a=15;
        int b=5;
        int x = a;
        int A= a+=b;
        int S =a -= b;
        int M= a*=b;
        int D = a/=b;
        int MOD = a%=b;

System.out.println (x)
System.out.println (A)
System.out.println (S)
System.out.println (M)
System.out.println (D)
System.out.psrintln (MOD)

    }
}
```



O/P

15
20
15
75
3
0

n/n

Assignment-1

Q1: Explain type conversions (implicit and explicit).

Ans:

Type conversion, also known as type casting is the process of assigning a value of one primitive data type to another data type.

1. Implicit type conversion (widening)
   - This type conversion is used when smaller data type to a larger data type.
   - This is automatically done by the compiler without any specification from the user's end.

eg: int a = 200;
float b = a; [b = 200.0]

2. Explicit type conversion (Narrowing)
   - This type conversion is used when larger data types are converted to a smaller data types.
   - Needs to be done manually and may result in loss of data.
   - This requires specification from user's end

eg: float a = 10.5
int b = (int) a; [b = 10]

Q4. Explain operator precedence

Ans:- Operator precedence in Java defines the order in which operators are evaluated in an expression.

1. Highest (Top) Precedence
   - Operators like increment (++) & decrement (--)
   - Unary operators (+), (-), (!), (~), (type), etc.

2. Medium precedence
   - Multiplication (*), division (/), modulo (%)
   - Addition, subtraction (+), (-)

3. Lower precedence
   - Shift operators (<<, >>, >>>)
   - Relational operators (<, <=, >, >=)
   - equality (==, !=)
   - Bitwise AND (&)
   - Bitwise XOR (^)
   - Bitwise OR (|)

4. Logical operators
   - Logical AND (&&)
   - Logical OR (||)

5. Ternary operators (conditional)

. ___ ? ___ : ___

6. Assignment operators

. =, +=, -=, *=, /=, %= (R → L)

. '&=', ^=, |= (R → L)

. <<=, >>=, >>>= (R → L)

Q 5. Write the syntax of all the control statements:

Ans:- ① 2-way statements

① if statement

```
if (condition) {
    // condition when becomes true.
}
```

② ~~else~~ if else statement

```
if (condition) { // code to be executed when condition is true
}
else {
    // code to be executed when condition is false.
}
```

③ if else-if statement

```
if (cond 1) { // code to be executed when cond 1 is true
else if (cond 2) { // code to executed when cond 2 is true
    else if (cond 3) {
        // code to executed when cond 3 is T
    } else { // when cond 3 is False
    } when cond 2 is F
```

when all are false

② Multiway (switch)

```
switch (expression) {
    case value1;
        // code to be executed if expression equal to value1
        break;
    case value2;
        // code to be executed if expression equal to value2
        break;
    case value3;
        // code to be excated if expression is equal to value3
        break;
    default:
        // code to be excated when none of the values match.
}
```

③ Loops (for, while, do while)

① for loop.

```
for (initialisation, condition, updation) {
    // code
}
```

② while loop

```
while (condition) {
    // code
}
```

(3) do while loop

```
do {
  //
}while( condition);
```

(4) Nested loops

(1) for loops

```
for (int i=0 ; i<=5; i++) {
    for(int j=0 ; j<=4 ;j++) {
        //code
    }
}
```

(2) while (cond 1) { // code

```
    while(cond 2) {
      // code
    }
}
```

JAVA ASSIGNMENT - 1
EXERCISE PROBLEMS

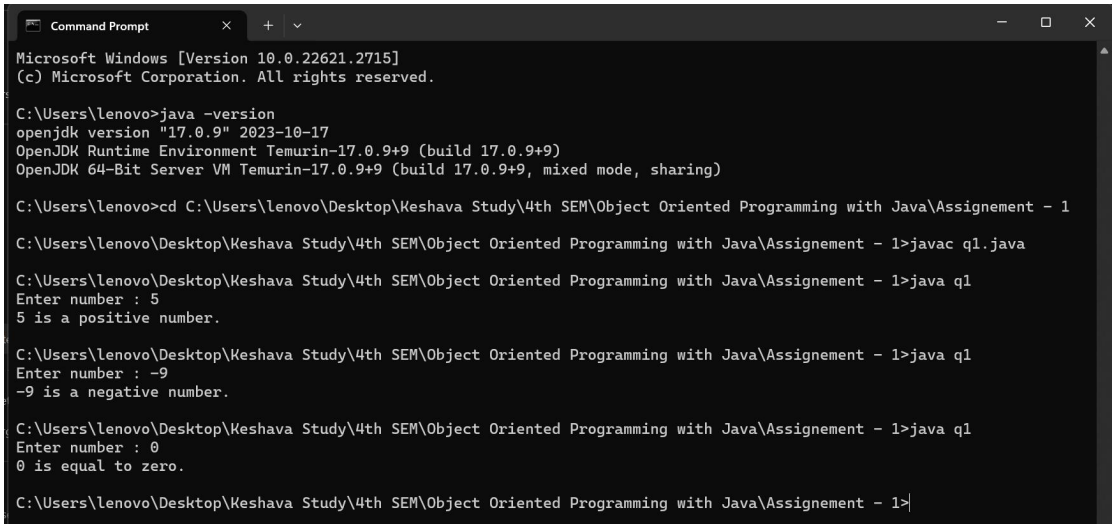
1. Program to read a number from the user and print whether it is positive or negative.

Program code:

```java
import java.util.*;
public class q1 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter number : ");
        int a = sc.nextInt();
        if(a>0) {
            System.out.println(a+" is a positive number.");
        }
        else if(a<0) {
            System.out.println(a+" is a negative number.");
        }
        else {
            System.out.println(a+" is equal to zero.");
        }

    }
}
```

Output:



2. Program to solve quadratic equations (use if, else if and else).

Program code:

```java
import java.util.Scanner;

public class q2 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter co-efficient a : ");
        float a = sc.nextFloat();
        System.out.print("Enter co-efficient b : ");
        float b = sc.nextFloat();
        System.out.print("Enter co-efficient c : ");
        float c = sc.nextFloat();
        float D = b*b-4*a*c;
        if (D > 0) {
            float x = (-b + (float) Math.sqrt(D)) / (2 * a);
            float y = (-b - (float) Math.sqrt(D)) / (2 * a);
            System.out.println("The equation has 2 real roots.");
            System.out.println("x = " + x + ", y = " + y);
        } else if (D == 0) {
            float x = -b / (2 * a);
            System.out.println("The equation has 1 real root.");
            System.out.println("x = " + x);
        } else {
            System.out.println("The equation has imaginary roots.");
        }
    }
}
```

Output :

```
C:\Users\lenovo>java -version
openjdk version "17.0.9" 2023-10-17
OpenJDK Runtime Environment Temurin-17.0.9+9 (build 17.0.9+9)
OpenJDK 64-Bit Server VM Temurin-17.0.9+9 (build 17.0.9+9, mixed mode, sharing)

C:\Users\lenovo>cd C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q2.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q2
Enter co-efficient a : 1
Enter co-efficient b : -5
Enter co-efficient c : 6
The equation has 2 real roots.
x = 3.0, y = 2.0

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q2
Enter co-efficient a : 1
Enter co-efficient b : 4
Enter co-efficient c : 4
The equation has 1 real root.
x = -2.0

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q2
Enter co-efficient a : 1
Enter co-efficient b : 1
Enter co-efficient c : 3
The equation has imaginary roots.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>
```

3. Take three numbers from the user and print the greatest number.

Program code :

```java
import java.util.Scanner;

public class q3 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int a,b,c;
        System.out.print("Enter first value : ");
        a=sc.nextInt();
        System.out.print("Enter second value : ");
        b=sc.nextInt();
        System.out.print("Enter third value : ");
        c=sc.nextInt();
        int top;
        if (a > b) {
            top = (a > c) ? a : c;
        } else {
            top = (b > c) ? b : c;
        }
        System.out.println("The greatest number is " + top);
    }
}
```

Output :

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo>java -version
openjdk version "17.0.9" 2023-10-17
OpenJDK Runtime Environment Temurin-17.0.9+9 (build 17.0.9+9)
OpenJDK 64-Bit Server VM Temurin-17.0.9+9 (build 17.0.9+9, mixed mode, sharing)

C:\Users\lenovo>cd C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q3.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q3
Enter first value : 4
Enter second value : 7
Enter third value : 1
The greatest number is 7

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q3
Enter first value : 3
Enter second value : -9
Enter third value : 15
The greatest number is 15

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q3
Enter first value : -5
Enter second value : -1
Enter third value : -3
The greatest number is -1

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>
```

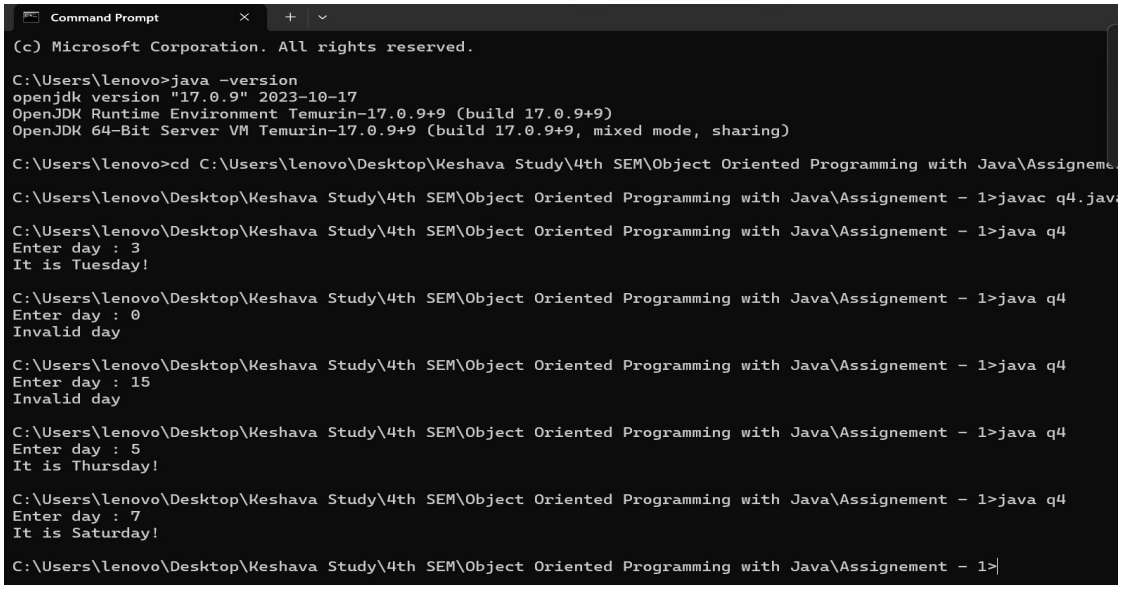
4. Program that keeps a number from the user and generates an integer between 1 and 7 and displays the name of the weekday.

Program code :

```java
import java.util.Scanner;

public class q4 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int day;
        System.out.print("Enter day : ");
        day = sc.nextInt();
        switch (day) {
            case 1:
                System.out.println("It is Sunday!");
                break;
            case 2:
                System.out.println("It is Monday!");
                break;
            case 3:
                System.out.println("It is Tuesday!");
                break;
            case 4:
                System.out.println("It is Wednesday!");
                break;
            case 5:
                System.out.println("It is Thursday!");
                break;
            case 6:
                System.out.println("It is Friday!");
                break;
            case 7:
                System.out.println("It is Saturday!");
                break;
            default:
                System.out.println("Invalid day");
        }
    }
}
```

Output :



5. Program that reads in two floating-point numbers and tests whether they are the same up to three decimal places.

Program Code :

```
iimport java.util.Scanner;

public class q5 {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.print("Enter first number: ");
        float x = in.nextFloat();
        System.out.print("Enter second number: ");
        float y = in.nextFloat();
        x = Math.round(x * 1000);
        x = x / 1000;
        y = Math.round(y * 1000);
        y = y / 1000;
        if (x == y) {
            System.out.println("Both are the same up to three decimal places");
        } else {
            System.out.println("Both are different");
        }
    }
}
```

Output :

```
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo>java -version
openjdk version "17.0.9" 2023-10-17
OpenJDK Runtime Environment Temurin-17.0.9+9 (build 17.0.9+9)
OpenJDK 64-Bit Server VM Temurin-17.0.9+9 (build 17.0.9+9, mixed mode, sharing)

C:\Users\lenovo>cd C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q5.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q5
Enter first number: 5.133
Enter second number: 2.342
Both are different

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q5
Enter first number: 1.389
Enter second number: 2.389
Both are different

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q5
Enter first number: 1.265
Enter second number: 1.265
Both are the same up to three decimal places

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>
```

6. Program that takes a year from user and print whether that year is a leap year or not.

Program Code :

```java
import java.util.Scanner;

public class q6 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int year;
        System.out.print("Enter a year: ");
        year = sc.nextInt();
        if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {
            System.out.println(year + " is a leap year.");
        } else {
            System.out.println(year + " is not a leap year.");
        }
    }
}
```

Output :

```
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q6.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q6
Enter a year: 2020
2020 is a leap year.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q6
Enter a year: 2018
2018 is not a leap year.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q6
Enter a year: 2016
2016 is a leap year.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q6
Enter a year: 2024
2024 is a leap year.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q6
Enter a year: 1990
1990 is not a leap year.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q6
Enter a year: 1900
1900 is not a leap year.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>
```

7. Program to display the first 10 natural numbers.

Program Code :

```java
public class q7 {
    public static void main(String[] args) {
        for(int i=1;i<=10;i++) {
            System.out.println(i);
        }
    }
}
```

Output :

```
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q7.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q7
1
2
3
4
5
6
7
8
9
10

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>
```

8. Program to input 5 numbers from keyboard and find their sum and average.

Program Code :

```java
import java.util.Scanner;

public class q8 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int sum=0;
        int a;
        for(int i=0;i<=4;i++) {
            System.out.print("Enter number "+(i+1)+" : ");
            a = sc.nextInt();
            sum+=a;
        }
        System.out.println("Sum of the 5 numbers : "+sum);
        int avg = sum/5;
        System.out.println("Average of the 5 numbers : "+avg);
    }
}
```

Output :

```
C:\Windows\System32\cmd.e
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q8.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q8
Enter number 1 : 4
Enter number 2 : 1
Enter number 3 : 6
Enter number 4 : 3
Enter number 5 : 8
Sum of the 5 numbers : 22
Average of the 5 numbers : 4

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q8
Enter number 1 : 1
Enter number 2 : 2
Enter number 3 : 3
Enter number 4 : 4
Enter number 5 : 5
Sum of the 5 numbers : 15
Average of the 5 numbers : 3

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q8
Enter number 1 : 1
Enter number 2 : 1
Enter number 3 : 1
Enter number 4 : 1
Enter number 5 : 1
Sum of the 5 numbers : 5
Average of the 5 numbers : 1

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>
```

9. Program in Java to display the multiplication table of a given integer.

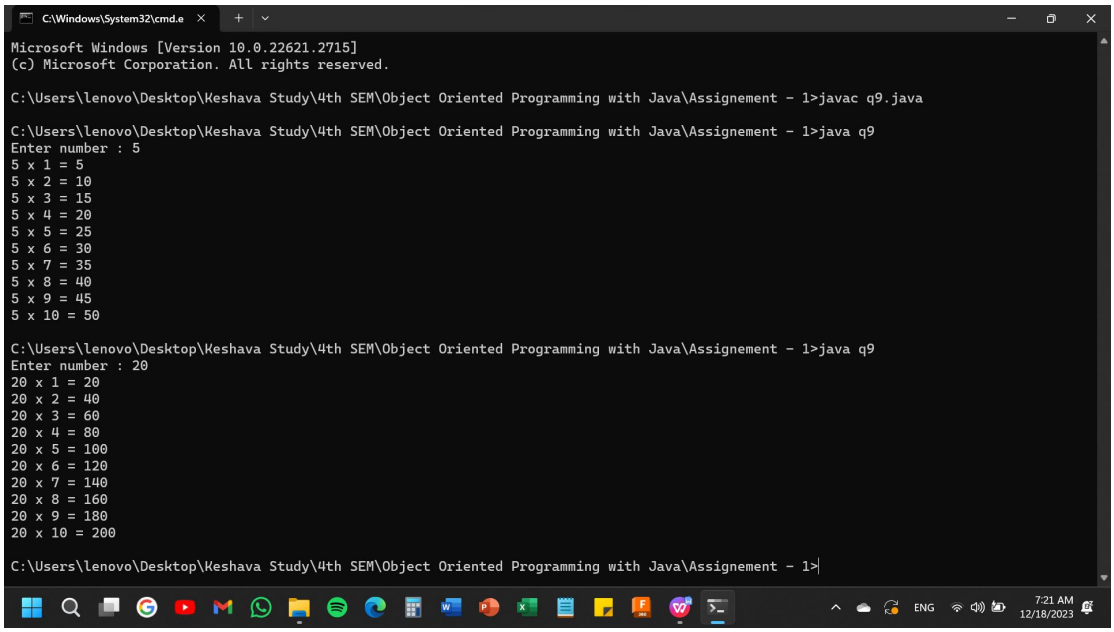
Program Code :

```java
import java.util.Scanner;

public class q9 {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int n;
        System.out.print("Enter number : ");
        n=sc.nextInt();
        for(int i=1;i<=10;i++) {
            System.out.println(n+" x "+i+" = "+n*i);
        }

    }
}
```

Output :



10. Program in Java to display the pattern like right angle triangle with a number.
Input number of rows : 5

Program Code :

```java
public class q10 {
    public static void main(String[] args) {
        for(int i=1;i<=5;i++) {
            for(int j=1;j<=i;j++) {
                System.out.print(j);
            }
            System.out.println();
        }
    }
}
```

Output :

```
Microsoft Windows [Version 10.0.22621.2715]
(c) Microsoft Corporation. All rights reserved.

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>javac q10.java

C:\Users\lenovo\Desktop\Keshava Study\4th SEM\Object Oriented Programming with Java\Assignement - 1>java q10
1
12
123
1234
12345
```