Batch: B2 Roll No.: 16010124107
Experiment / assignment / tutorial No.08
Grade: AA / AB / BB / BC / CC / CD /DD
Signature of the Staff In-charge with date

TITLE : GCD and LCM

AIM: Write a recursive function 'gcd' to find the gcd of the given two numbers. Use this in main to find the gcd and LCM two given numbers.

Variations :
Implementation of Program with One class

Accessibility with static and non-static methods within class and outside class.

Expected OUTCOME of Experiment:
CO2: Explore arrays, vectors, classes and objects in C++ and Java

Books/ Journals/ Websites referred:

1. E. Balagurusamy , "Programming with Java" McGraw-Hill.

2. Sachin Malhotra, Saurabh Choudhary, "Programming in Java", Oxford Publications.

Pre Lab/ Prior Concepts:

The Scanner class is a class in java.util, which allows the user to read values of various types. There are far more methods in class Scanner than you will need in this course. We only cover a small useful subset, ones that allow us to read in numeric values from

either the keyboard or file without having to convert them from strings and determine if there are more values to be read.

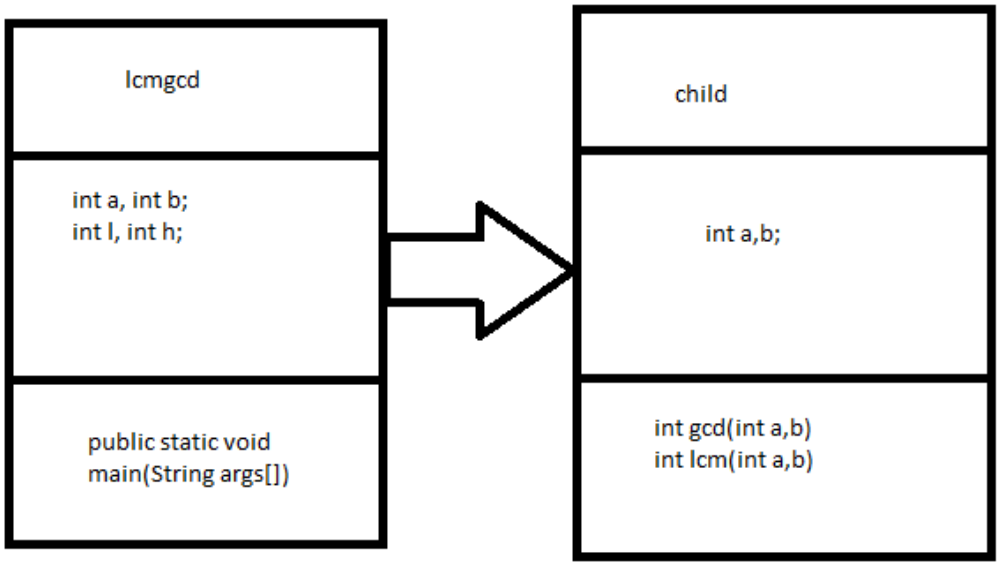
Scanner in = new Scanner(System.in); // System.in is an InputStream
Numeric and String Methods

Method	Returns
int nextInt()	Returns the next token as an int. If the next token is not an integer,InputMismatchException is thrown.
long nextLong()	Returns the next token as a long. If the next token is not an integer,InputMismatchException is thrown.
float nextFloat()	Returns the next token as a float. If the next token is not a float or is out of range, InputMismatchException is thrown.
double nextDouble()	Returns the next token as a long. If the next token is not a float or is out of range, InputMismatchException is thrown.
String next()	Finds and returns the next complete token from this scanner and returns it as a string; a token is usually ended by whitespace such as a blank or line break. If not token exists,NoSuchElementException is thrown.
String nextLine()	Returns the rest of the current line, excluding any line separator at the end.
void close()	Closes the scanner.

The Scanner looks for tokens in the input. A token is a series of characters that ends with what Java calls whitespace. A whitespace character can be a blank, a tab character, a carriage return. Thus, if we read a line that has a series of numbers separated by blanks, the scanner will take each number as a separate token. .
The numeric values may all be on one line with blanks between each value or may be on separate lines. Whitespace characters (blanks or carriage returns) act as separators. The next method returns the next input value as a string, regardless of what is keyed. For example, given the following code segment and data

- int number = in.nextInt();
- float real = in.nextFloat();
- long number2 = in.nextLong();
- double real2 = in.nextDouble();
- String string = in.next();

Class Diagram:



Algorithm:

1. Start
2. Import scanner class
3. Create a class
4. Declare main() method inside it.
5. Take input for variables inside main after prompting the user
6. Declare two more methods for gcd and lcm using recursive method.
7. Call the functions inside main using the three different methods: static, object creation same class, object creation different class.
8. Display the output.
9. Close scanner
10. Exit.

Implementation details:

1. Same class, without using object: declared method as static

```
import java.util.*;


class lcmgcd{
    static int gcd(int a,int b)
    {
        if(b==0) return a;
        else return gcd(b,a%b);
    }

    static int lcm(int a,int b)
    {
        return (a*b)/gcd(a,b);
    }

    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        int a,b;
        System.out.println("Enter two numbers for gcd.");
        a = sc.nextInt();
        b = sc.nextInt();
        System.out.println("Their gcd is: " + gcd(a,b));
        System.out.println("Their lcm is: "+ lcm(a,b));
```

```
    }
}
```

2. Same class, by making an object

```
import java.util.*;


class lcmgcd{
    int gcd(int a,int b)
    {
        if(b==0) return a;
        else return gcd(b,a%b);
    }

    int lcm(int a,int b)
    {
        return (a*b)/gcd(a,b);
    }

    public static void main(String args[])
    {
        Scanner sc = new Scanner(System.in);
        int a,b;
        System.out.println("Enter two numbers for gcd.");
        a = sc.nextInt();
```

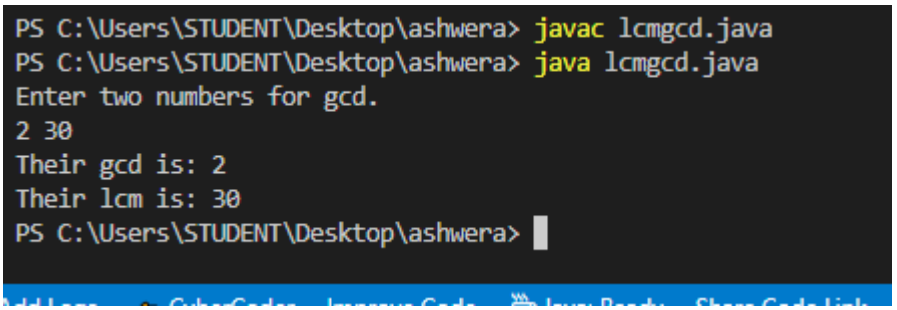
```
        b = sc.nextInt();

        lcmgcd ob = new lcmgcd();

        int h = ob.gcd(a,b);

        int l = ob.lcm(a,b);

        System.out.println("Their gcd is: " + h);

        System.out.println("Their lcm is: "+ l);

    }



}
```

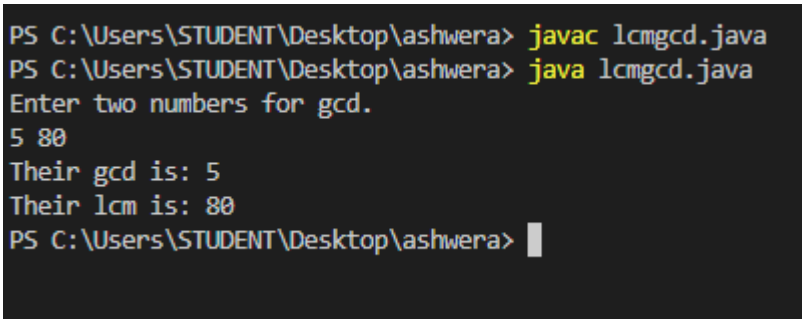
3. Different classes. By making an object

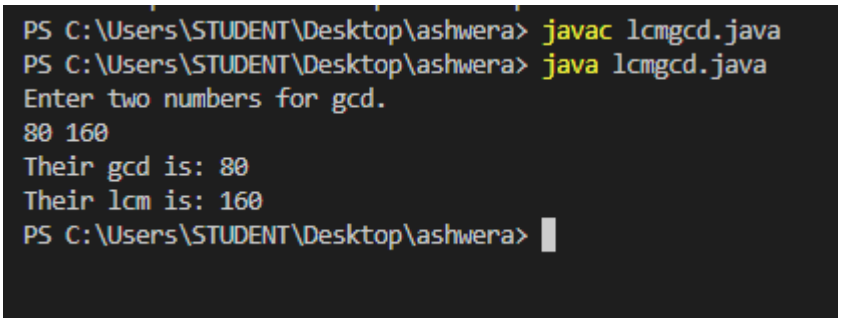
```
import java.util.*;


class lcmgcd{


    public static void main(String args[])

    {

        Scanner sc = new Scanner(System.in);

        int a,b;

        System.out.println("Enter two numbers for gcd.");

        a = sc.nextInt();

        b = sc.nextInt();

        child ob = new child();
```

```
        int h = ob.gcd(a,b);
        int l = ob.lcm(a,b);
        System.out.println("Their gcd is: " + h);
        System.out.println("Their lcm is: "+ l);
    }
}

class child
{
    int gcd(int a,int b)
    {
        if(b==0) return a;
        else return gcd(b,a%b);
    }

    int lcm(int a,int b)
    {
        return (a*b)/gcd(a,b);
    }
}
```

Output:

1. 

2. 

3. 

Conclusion:

In java programming, we can call functions in three different ways.

1. If the methods are in the same class, make an object with that class's name and call it inside main.
2. If the methods are in the same class but we do not want to make an object, declare the methods as static.
3. If the methods are in two different classes, make an object in the main() class using the other class name, then using this object, call the method.

Recursion allows us to call the same function within itself to perform some task iteratively. It is quite similar to a loop, but it is more efficient and implementable.

Date: 28/07/2025 **Signature of faculty in-charge**

Post Lab Descriptive Questions:

Q.1 Write a program to find the area and circumference of a circle using two classes.

```
import java.util.*;
class a
{
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter side of a circle: ");
        double r = sc.nextDouble();
        b obj = new b();
        double area = b.Area(r);
        System.out.println("Area of circle is: "+area);
        double circumference = b.Circumference(r);
        System.out.println("Circumference of circle is: "+circumference);
    }
}
class b{
    static double Circumference(double side)
    {
        return 3.14 * 2 * side;
    }
    static double Area (double side)
    {
        return 3.14 * Math.pow(side, 2);
    }
}
```

Output:

```
Enter side of a circle:
5
Area of circle is: 78.5
Circumference of circle is: 31.400000000000002
PS C:\Users\STUDENT\Desktop\ashwera>
```

Q.2 Write the output of following program

```
1.      public class BreakExample2 {
2.      public static void main(String[] args) {
3.              //outer loop
4.              for(int i=1;i<=3;i++){
5.                      //inner loop
6.                      for(int j=1;j<=3;j++){
7.                         if(i==2&&j==2){
8.                            //using break statement inside the inner loop
9.                            break;
10.                        }
11.                        System.out.println(i+" "+j);
12.                     }
13.             }
14.     }
15.     }
```

Output:

1 1

1 2

1 3

2 1

3 1

3 2

3 3

Q.3 Why is Java known as a platform independent language?

Java files are first compiled by the java compiler and then stored in bytecode in a .class file. The bytecode is platform independent, meaning it is not specific to a particular operating system. This is why Java is known as a platform independent language.

Q.4 Write a recursive static method for calculation of factorial of a number.

code:

```
import java.util.*;
class a
```

```
{
    static int factorial(int n)
    {
        if(n==0)
        {
            return 1;
        }
        else
        return n*factorial(n-1);
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter number: ");
        int n = sc.nextInt();
        int x = factorial(n);
        System.out.println("Factorial of " + n + " is " + x);
    }
}
```

output:

```
PS C:\Users\STUDENT\Desktop\ashwera> javac a.java
PS C:\Users\STUDENT\Desktop\ashwera> java a.java
Enter number:
5
Factorial of 5 is 120
PS C:\Users\STUDENT\Desktop\ashwera>
```