METHODS

What are methods:

- Methods are members of classes which provide functionality for classes.
- We can write our own methods in the classes.
- The functions performing on the data are known as methods.
- When a method returns a value then the method itself takes the value.
- A method will have its own copy of variable.
- Skeleton of method:

  returnType methodName(parameter list) →signature/header

  {

  }

  Example program:

```
class test
{
   static int max(int x, int y)
   {
      x++;
      if(x>y)
        return x;
      else
        return y;
   }
   public static void main(String args[])
   {
     int a=10,b=15,c;
     c=max(a,b);
     System.out.println(c);
   }
}
```

Passing object as parameters:

- To call a method from main method it is needed to be maid static.
- When the method is called the value of actual parameters are copied in formal parameters which is the only parameter passing method in java.
- The address of object in formal and actual parameter is Same.
- String cannot be modified as it is immutable.
- A method can also return an object.

  Example program:

```
class test

{
   static void update(int A[])
   {
     A[0]=25;
   }
   public static void main(String args[])
   {
      int A[]={2,3,4,5,6};
      update(A);
      System.out.println(A[0]);
   }
}
```

- A method can have its object as the return type.

Parameter passing in java.

- Whoever is calling a method is called as a caller or a method call.
- The method which is called by a caller is known as called method.
- The parameters/arguments passed in calling method are called as actual parameters.

- And the parameters of a called method are called as formal parameters.
- Formal parameters are nothing but input into a method where the return type is known as output to a method.
- The contents of actual parameters are copied in formal parameters is the only method of parameter passing in java.
- Passing of objects also follow the same method.
- Parameter passing for primitive datatypes the values are copied in formal parameters, whereas in parameter passing of objects the reference of the object id is copied in formal parameters.
- In short the primitive datatypes are passed by value and the objects are passed by reference.

Example program:

```
class test
{
   int add(int x, int y)
   {
     int z;
     z=x+y;
     return z;
   }
   public static void main(String args[])
   {
      int a=10,b=5,c;
      c=add(a,b);
      System.out.prinlt(c);
   }
}
```

Method overloading:

- Method overloading means writing more than one method having same names but different parameter list or data types.

➢ **Compiler will call the corresponding method depending upon the parameter list.**

Java Tutorial: Variable Arguments (VarArgs) in Java

- In the previous tutorial, we discussed how we can overload the methods in Java.
- Now, let's suppose you want to overload an "add" method. The "add" method will accept one argument for the first time and every time the number of arguments passed will be incremented by 1 till the number of arguments is equaled to 10.
- One approach to solve this problem is to overload the "add" method 10 times. But is it the optimal approach? What if I say that the number of arguments passed will be incremented by 1 till the number of arguments is equaled to 1000. Do you think that it is good practice to overload a method 1000 times?
- To solve this problem of method overloading, Variable Arguments(Varargs) were introduced with the release of JDK 5.
- With the help of Varargs, we do not need to overload the methods.

Syntax :

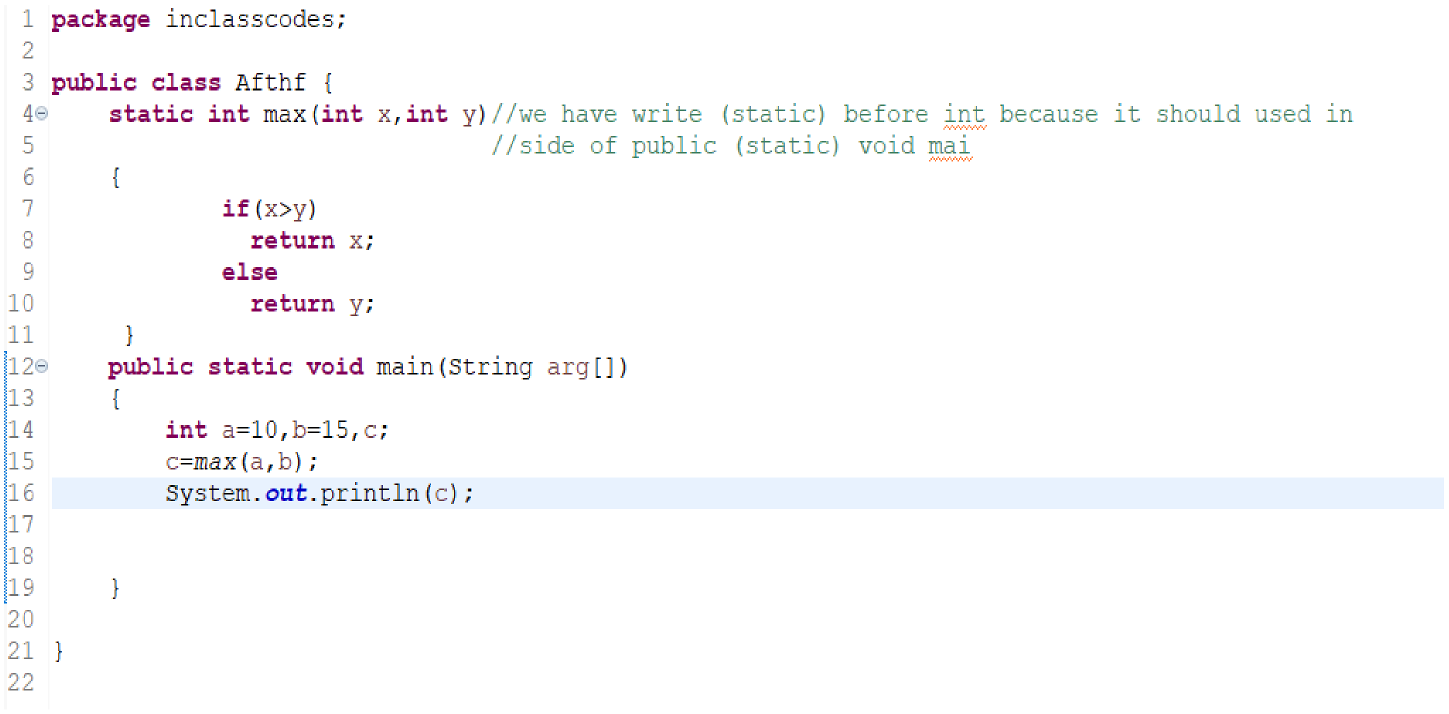
```
/*
public static void foo(int … arr)
{
// arr is available here as int[] arr
}
*/
```

- foo can be called with zero or more arguments like this:
  - foo(7)
  - foo(7,8,9)

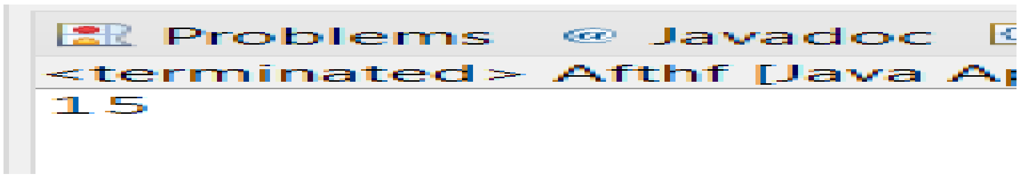
RECURSION

- A recursive method is the one which will call itself.
- When the recursive function can not call itself further because of the base condition it will return back along the same path.
- Not to make the program lengthy loops are used instead of recursive functions.
- The recursions are used in problem solving.

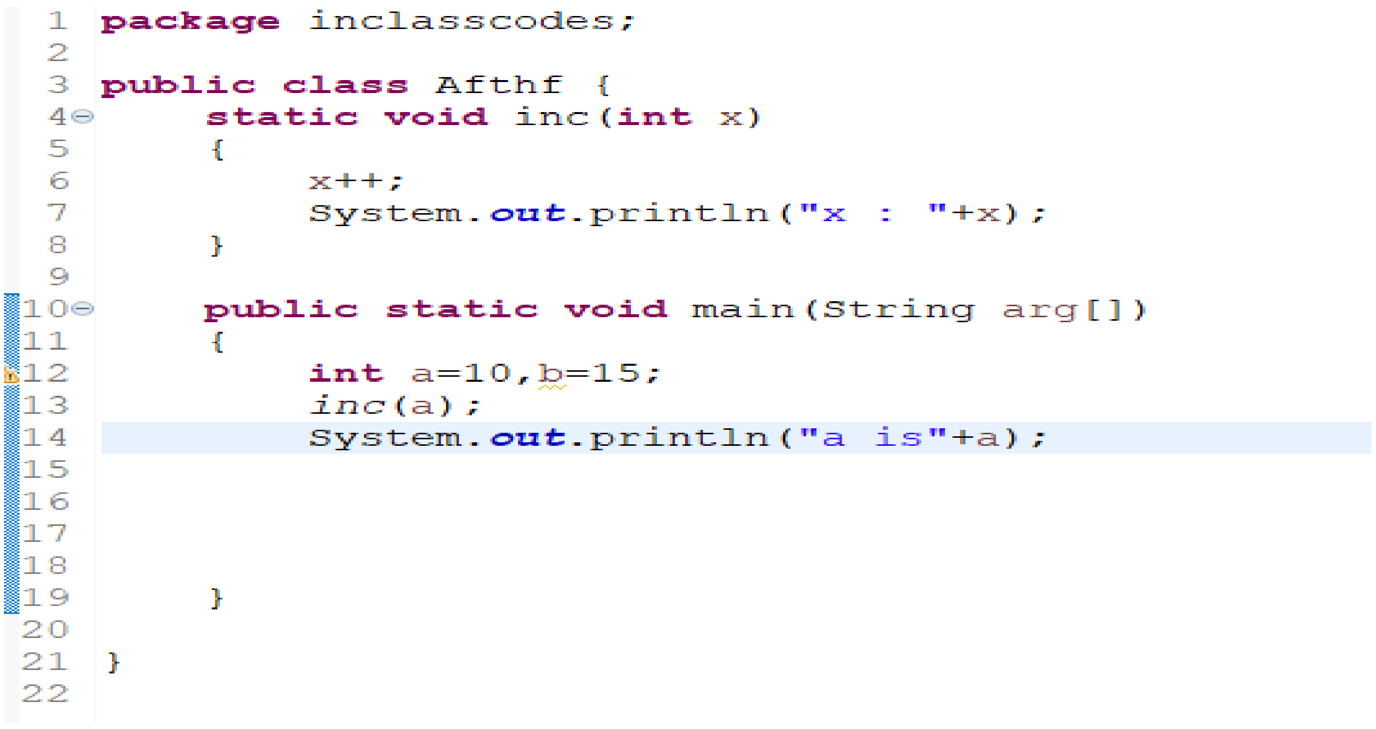
90. WRITING METHODS -- --



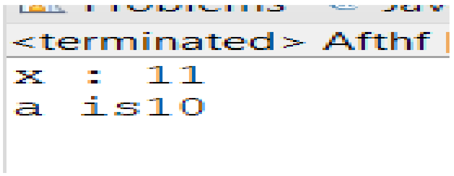
```
package inclasscodes;

public class Afthf {
    static int max(int x,int y)//we have write (static) before int because it should used in
                               //side of public (static) void mai
    {
            if(x>y)
              return x;
            else
              return y;
     }
    public static void main(String arg[])
    {
        int a=10,b=15,c;
        c=max(a,b);
        System.out.println(c);


    }

}

```



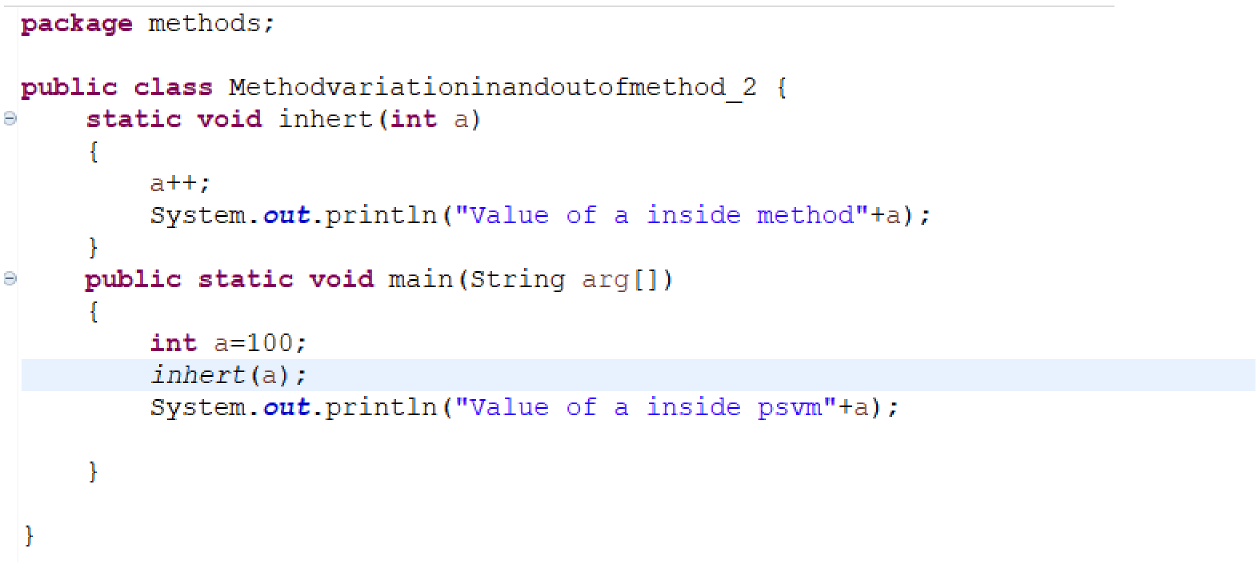
```
Problems   Javadoc
<terminated> Afthf [Java Ap
15
```



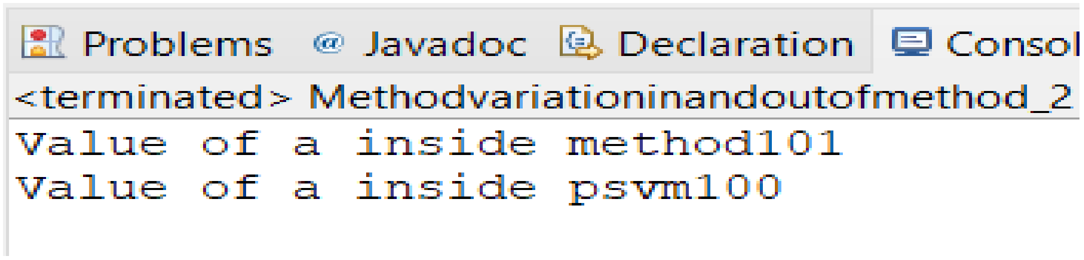
```
package inclasscodes;

public class Afthf {
    static void inc(int x)
    {
        x++;
        System.out.println("x : "+x);
    }

    public static void main(String arg[])
    {
        int a=10,b=15;
        inc(a);
        System.out.println("a is"+a);




    }

}

```



```
<terminated> Afthf
x  :  11
a  is10
```



```
package methods;

public class Methodvariationinandoutofmethod_2 {
    static void inhert(int a)
    {
        a++;
        System.out.println("Value of a inside method"+a);
    }
    public static void main(String arg[])
    {
        int a=100;
        inhert(a);
        System.out.println("Value of a inside psvm"+a);

    }

}
```

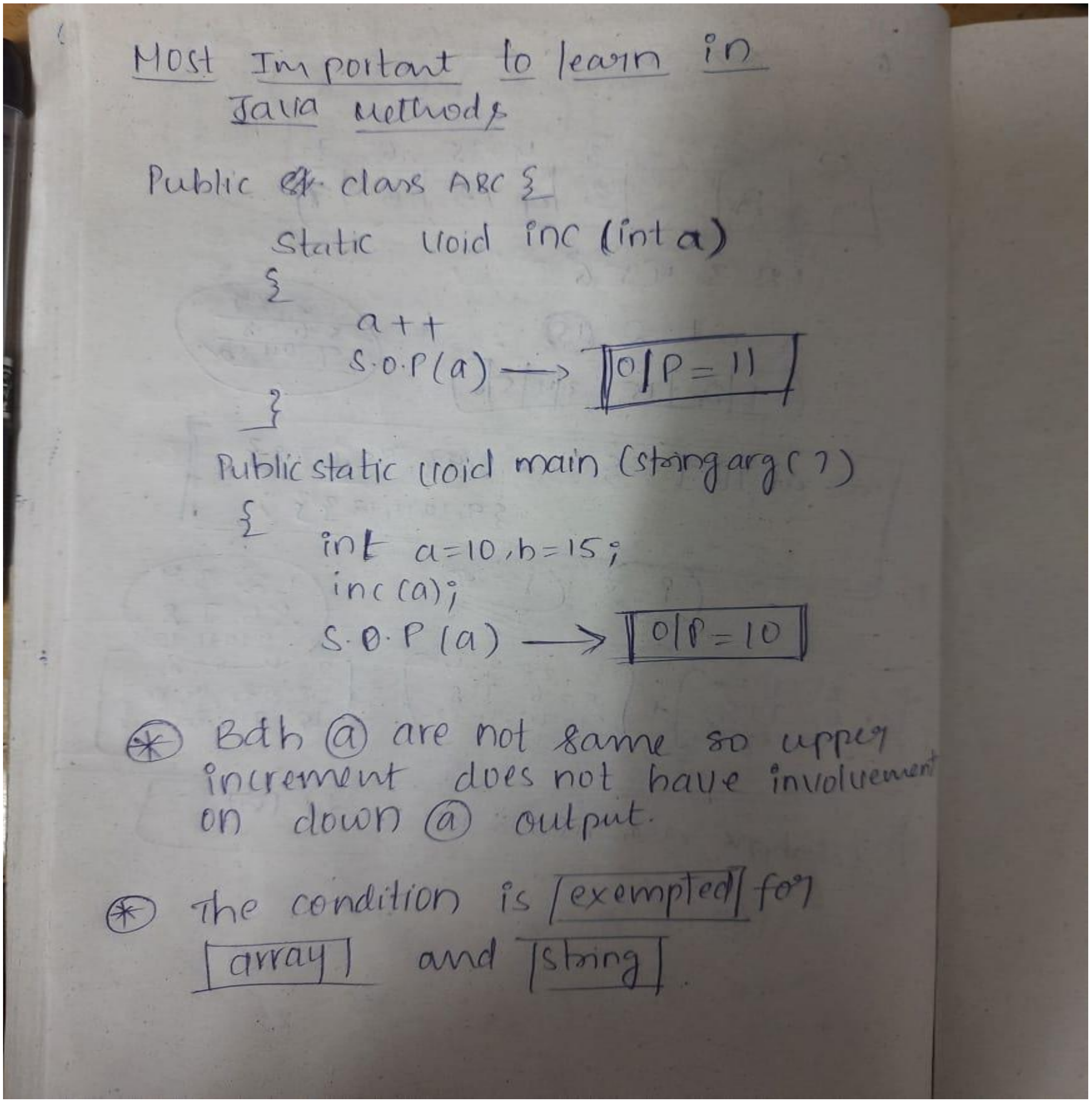


```
<terminated> Methodvariationinandoutofmethod_2
Value of a inside method101
Value of a inside psvm100
```

HERE SEE THE BOTH A ARE DIFFERENT BECAUSE ***ONE A IS IN METHOD AND ***ONE A IS PSVM;;;

******HERE BOTH WILL HAVE DIFFERENT STORAGE*****

Most important to note--- ---



Most Important to learn in Java Methods

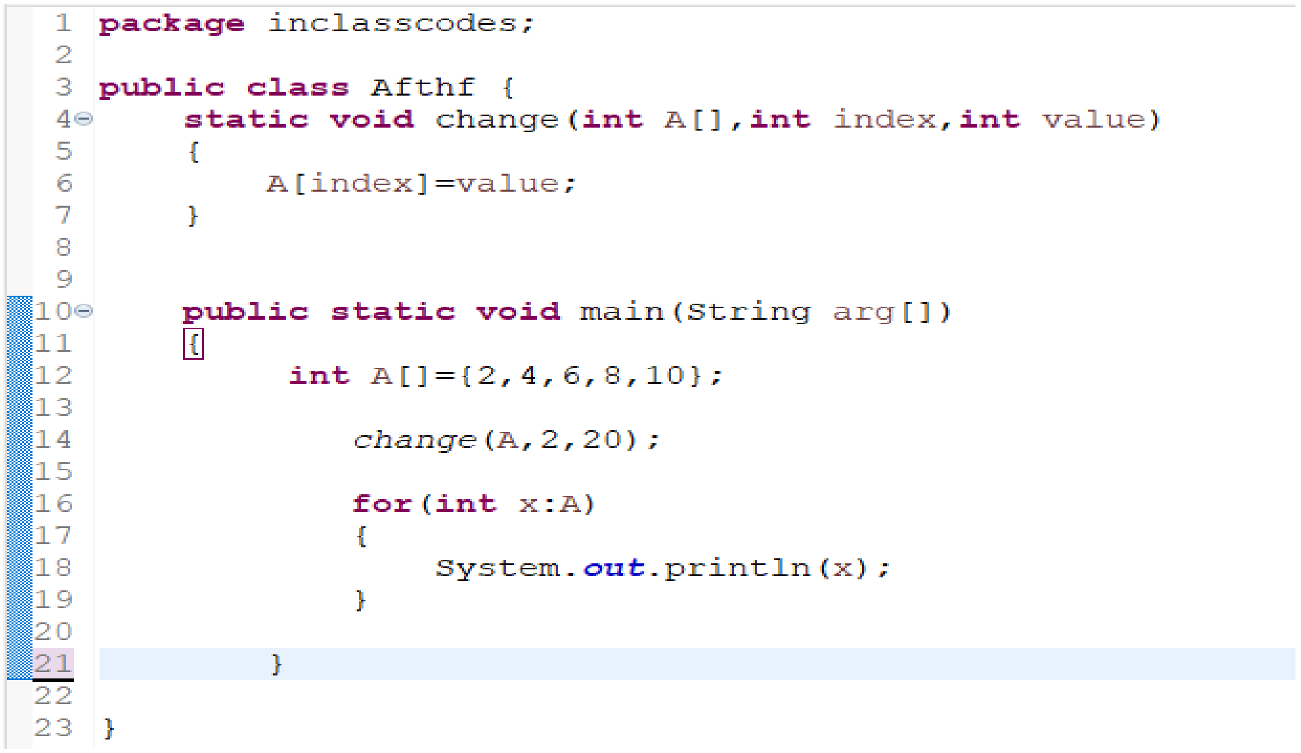
```
Public ~~cl~~ class ABC {
    Static void inc (int a)
    {
        a++
        S.O.P(a) ——> [O/P = 11]
    }
    Public static void main (string args[ ])
    {
        int a=10, b=15;
        inc (a);
        S.O.P (a) ——> [O/P = 10]
```

(*) Both (a) are not same so upper increment does not have involvement on down (a) output.

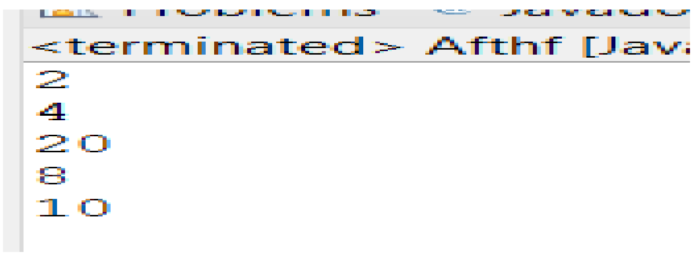
(*) The condition is [exempted] for [array] and [string].

UPPER WRITTEN THAT ARRAY IS EXEMPTED FROM THAT CASE THE EXAMPLE IS --------

92. PRACTISING OBJECT PASSING—

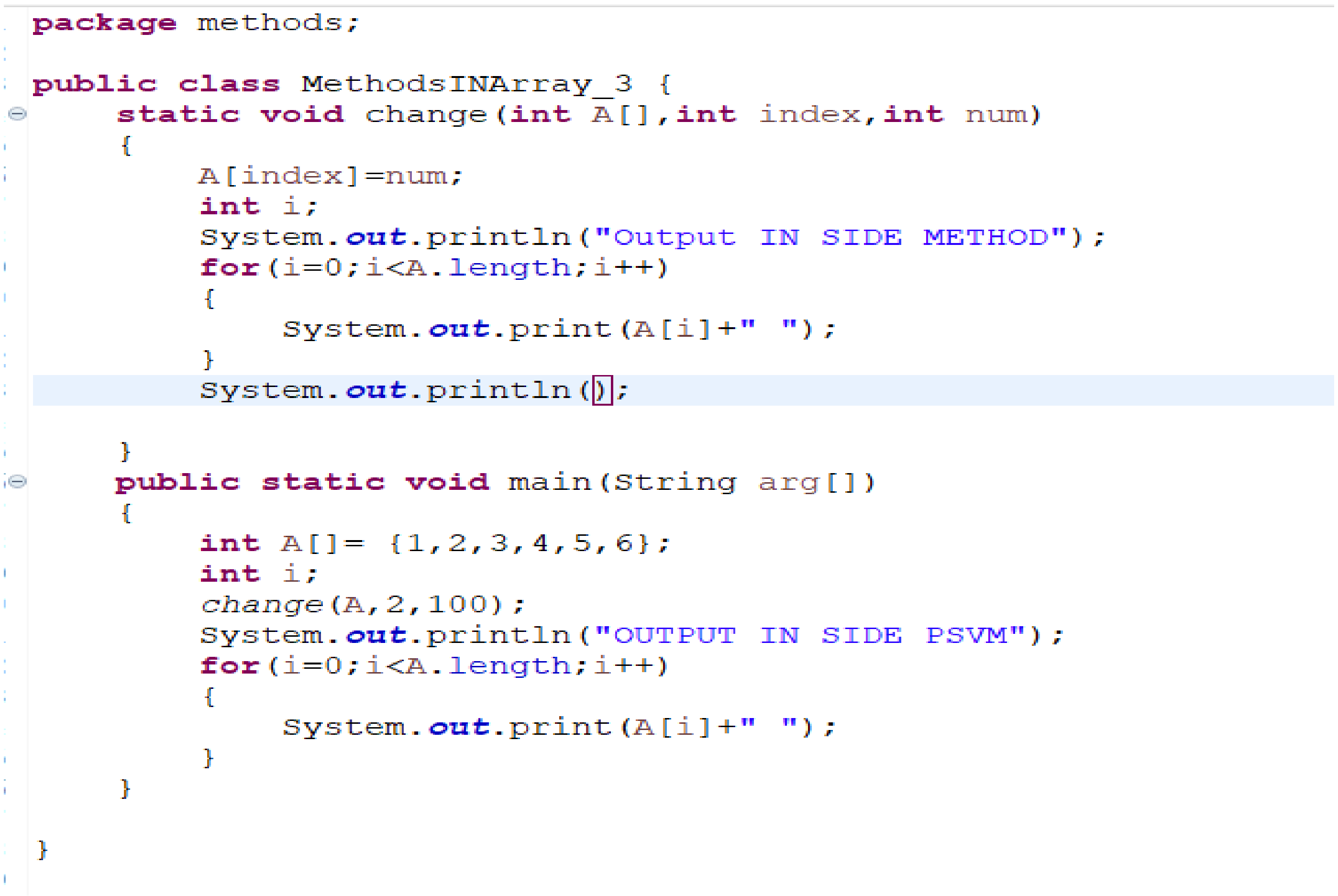


```java
package inclasscodes;

public class Afthf {
	static void change(int A[],int index,int value)
	{
		A[index]=value;
	}


	public static void main(String arg[])
	{
		 int A[]={2,4,6,8,10};

			change(A,2,20);

			for(int x:A)
			{
				System.out.println(x);
			}

		}

}
```

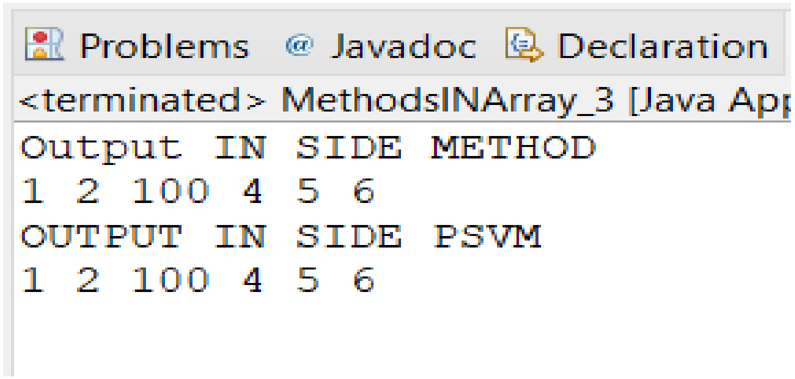


```
<terminated> Afthf [Jav
2
4
20
8
10
```

HERE ONCE OBSERVE IN DOWN EXAMPLE OUTPUT INSIDE METHOD AND OUTPUT INSIDE PSVM IS SAME----

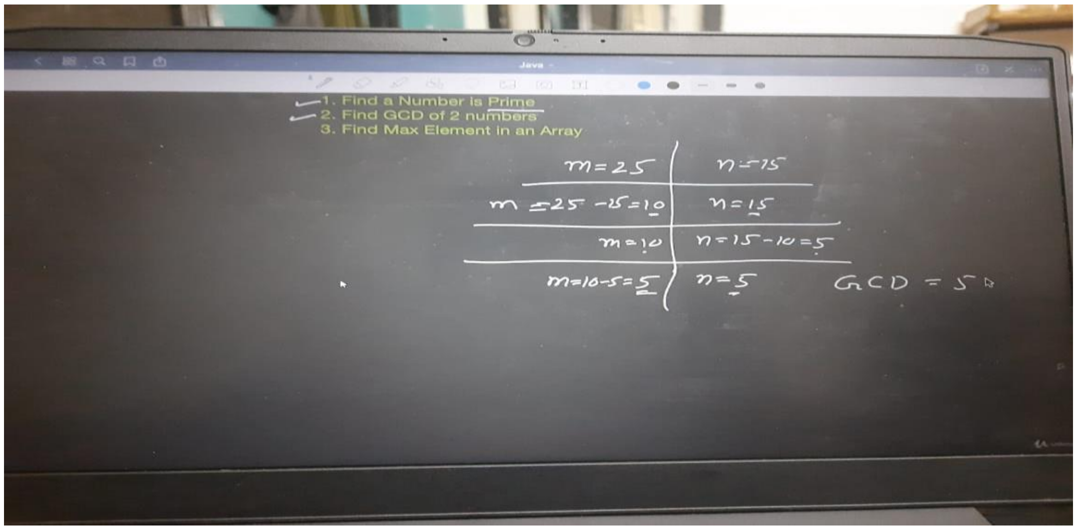


```java
package methods;

public class MethodsINArray_3 {
    static void change(int A[],int index,int num)
    {
        A[index]=num;
        int i;
        System.out.println("Output IN SIDE METHOD");
        for(i=0;i<A.length;i++)
        {
            System.out.print(A[i]+" ");
        }
        System.out.println();

    }
    public static void main(String arg[])
    {
        int A[]= {1,2,3,4,5,6};
        int i;
        change(A,2,100);
        System.out.println("OUTPUT IN SIDE PSVM");
        for(i=0;i<A.length;i++)
        {
            System.out.print(A[i]+" ");
        }
    }

}
```



```
<terminated> MethodsINArray_3 [Java Apr
Output IN SIDE METHOD
1 2 100 4 5 6
OUTPUT IN SIDE PSVM
1 2 100 4 5 6
```

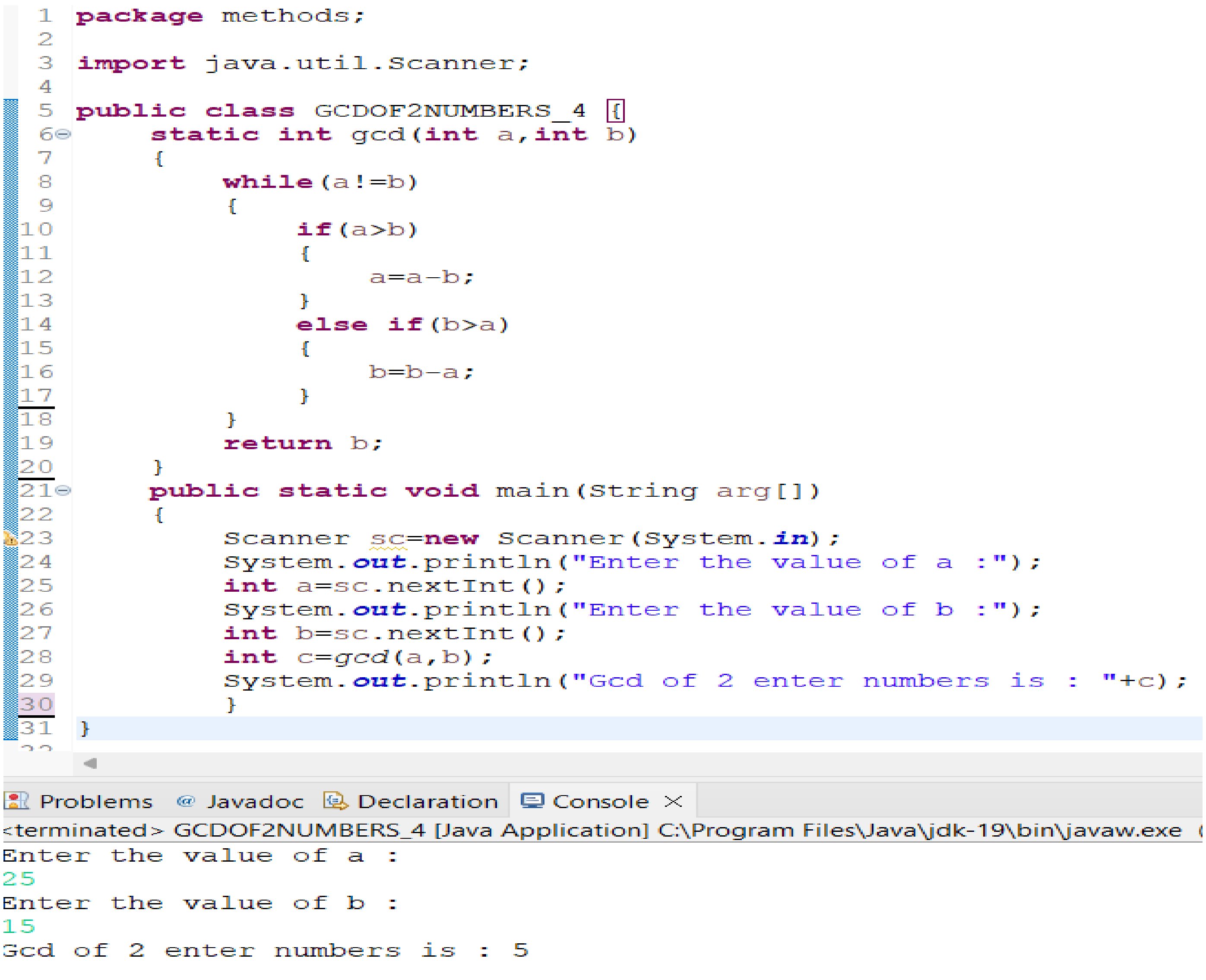
GCD OF TWO NUMBERS---



1. Find a Number is Prime
2. Find GCD of 2 numbers
3. Find Max Element in an Array

m=25	n=15
m =25 -15=10	n=15
m=10	n=15-10=5
m=10-5=5	n=5

GCD = 5

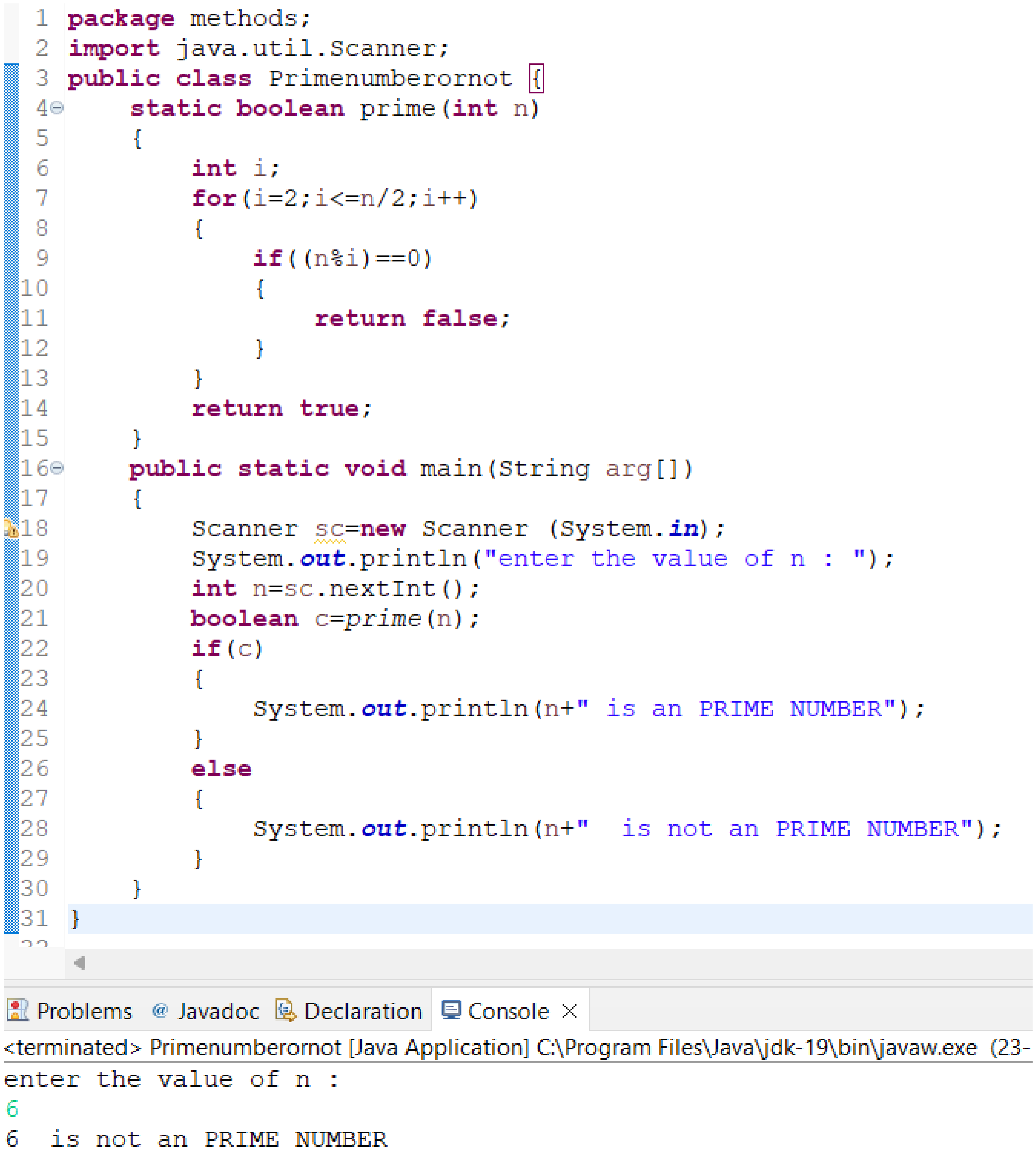


```java
package methods;

import java.util.Scanner;

public class GCDOF2NUMBERS_4 {
	static int gcd(int a,int b)
	{
		while(a!=b)
		{
			if(a>b)
			{
				a=a-b;
			}
			else if(b>a)
			{
				b=b-a;
			}
		}
		return b;
	}
	public static void main(String arg[])
	{
		Scanner sc=new Scanner(System.in);
		System.out.println("Enter the value of a :");
		int a=sc.nextInt();
		System.out.println("Enter the value of b :");
		int b=sc.nextInt();
		int c=gcd(a,b);
		System.out.println("Gcd of 2 enter numbers is :  "+c);
		}
}
```

Problems   Javadoc   Declaration   Console

```
<terminated> GCDOF2NUMBERS_4 [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe
Enter the value of a :
25
Enter the value of b :
15
Gcd of 2 enter numbers is : 5
```

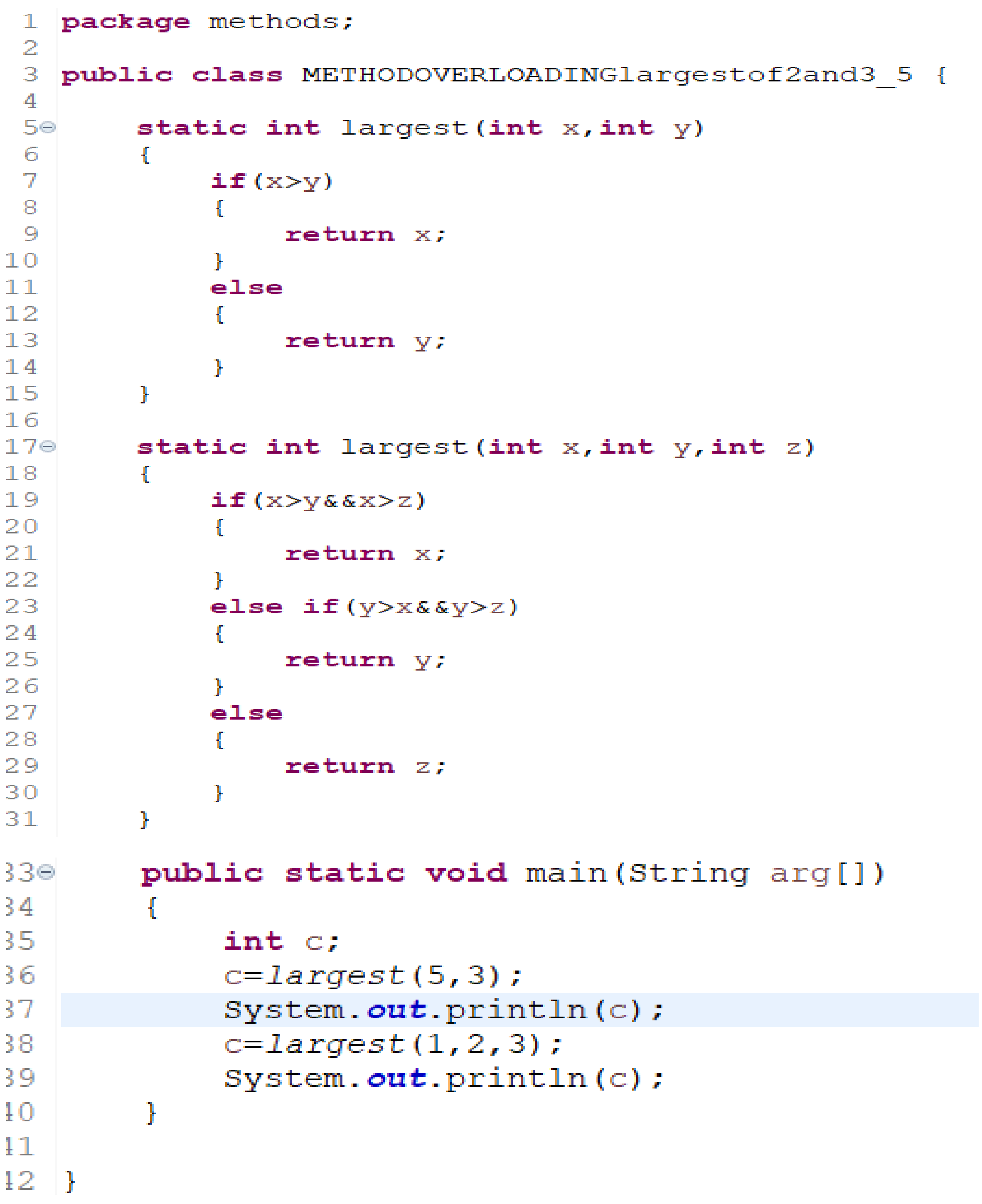
PRIME NUMBER-------



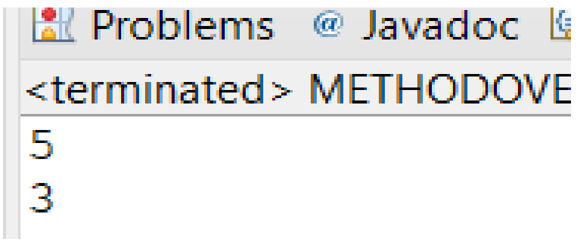
```java
package methods;
import java.util.Scanner;
public class Primenumberornot {
    static boolean prime(int n)
    {
        int i;
        for(i=2;i<=n/2;i++)
        {
            if((n%i)==0)
            {
                return false;
            }
        }
        return true;
    }
    public static void main(String arg[])
    {
        Scanner sc=new Scanner (System.in);
        System.out.println("enter the value of n : ");
        int n=sc.nextInt();
        boolean c=prime(n);
        if(c)
        {
            System.out.println(n+" is an PRIME NUMBER");
        }
        else
        {
            System.out.println(n+"  is not an PRIME NUMBER");
        }
    }
}
```

```
<terminated> Primenumberornot [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe  (23-
enter the value of n :
6
6  is not an PRIME NUMBER
```

96 METHOD OVERLOADING---

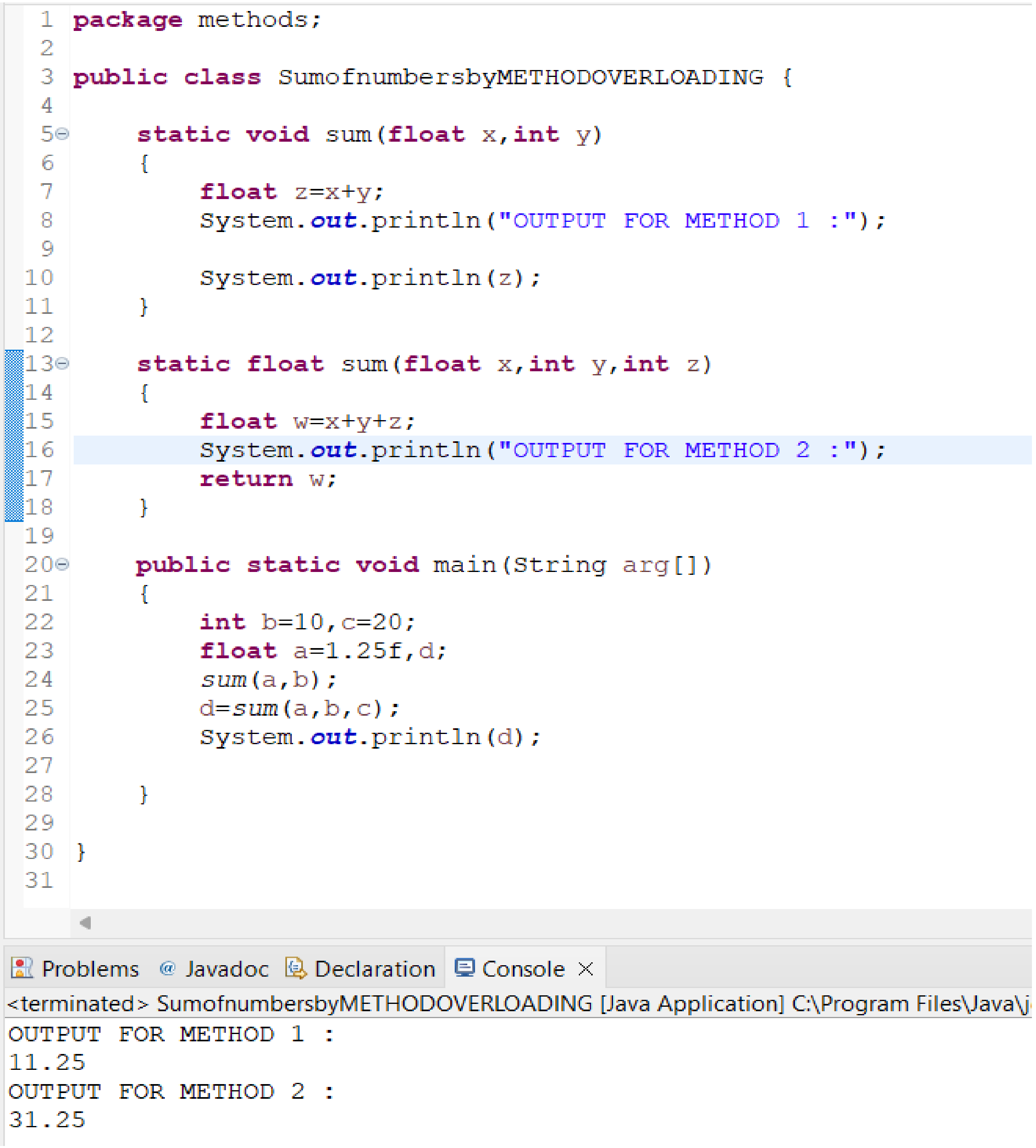


```
package methods;

public class METHODOVERLOADINGlargestof2and3_5 {

    static int largest(int x,int y)
    {
        if(x>y)
        {
            return x;
        }
        else
        {
            return y;
        }
    }

    static int largest(int x,int y,int z)
    {
        if(x>y&&x>z)
        {
            return x;
        }
        else if(y>x&&y>z)
        {
            return y;
        }
        else
        {
            return z;
        }
    }
    public static void main(String arg[])
    {
        int c;
        c=largest(5,3);
        System.out.println(c);
        c=largest(1,2,3);
        System.out.println(c);
    }

}
```



```
<terminated> METHODOVE
5
3
```

SUM OF (2 AND 3) NUMBERS IN METHOD OVERLOADING—



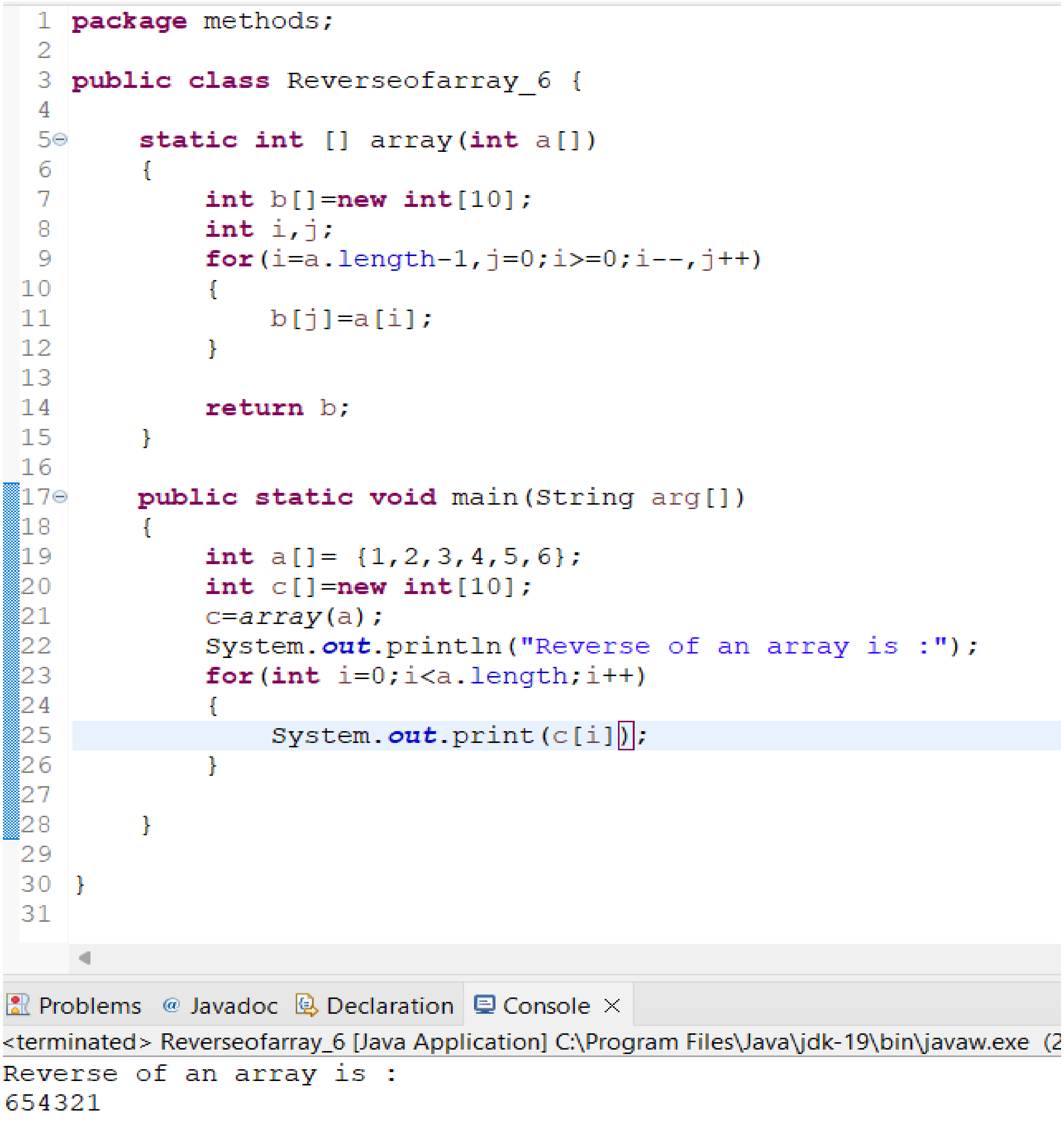
```java
package methods;

public class SumofnumbersbyMETHODOVERLOADING {

    static void sum(float x,int y)
    {
        float z=x+y;
        System.out.println("OUTPUT FOR METHOD 1 :");

        System.out.println(z);
    }

    static float sum(float x,int y,int z)
    {
        float w=x+y+z;
        System.out.println("OUTPUT FOR METHOD 2 :");
        return w;
    }

    public static void main(String arg[])
    {
        int b=10,c=20;
        float a=1.25f,d;
        sum(a,b);
        d=sum(a,b,c);
        System.out.println(d);

    }

}
```

Problems Javadoc Declaration Console

<terminated> SumofnumbersbyMETHODOVERLOADING [Java Application] C:\Program Files\Java\j

```
OUTPUT FOR METHOD 1 :
11.25
OUTPUT FOR METHOD 2 :
31.25
```

REVERSE OF ARRAY-----

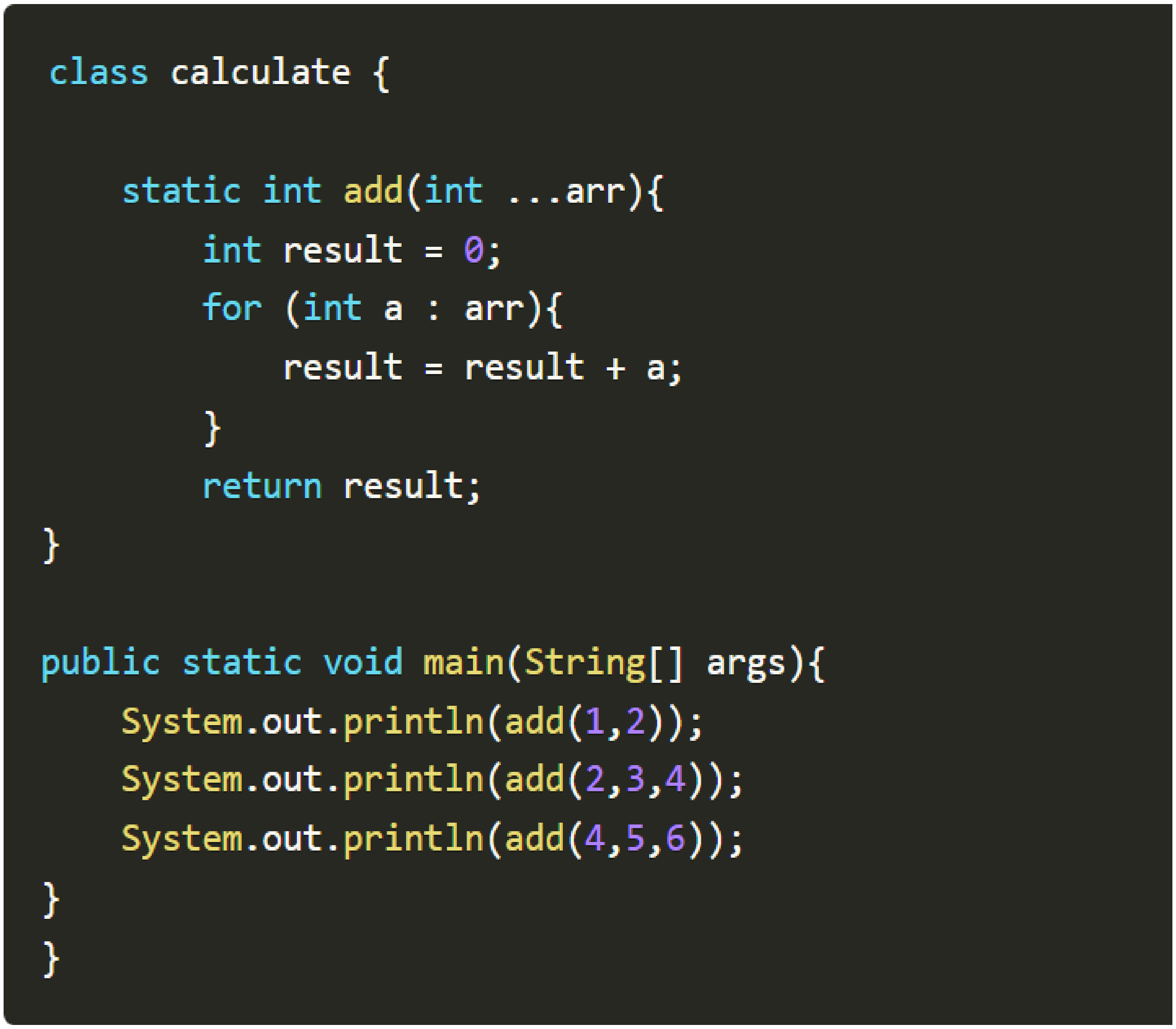


```java
package methods;

public class Reverseofarray_6 {

    static int [] array(int a[])
    {
        int b[]=new int[10];
        int i,j;
        for(i=a.length-1,j=0;i>=0;i--,j++)
        {
            b[j]=a[i];
        }

        return b;
    }

    public static void main(String arg[])
    {
        int a[]= {1,2,3,4,5,6};
        int c[]=new int[10];
        c=array(a);
        System.out.println("Reverse of an array is :");
        for(int i=0;i<a.length;i++)
        {
            System.out.print(c[i]);
        }

    }

}
```

```
<terminated> Reverseofarray_6 [Java Application] C:\Program Files\Java\jdk-19\bin\javaw.exe  (2
Reverse of an array is :
654321
```

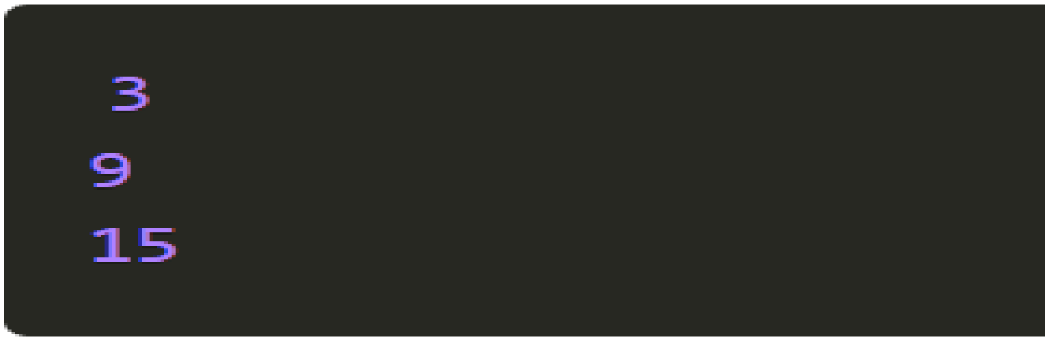
VARIABLE ARGUMENTS EXAMPLE--

Example of Varargs In Java :



```java
class calculate {

    static int add(int ...arr){
        int result = 0;
        for (int a : arr){
            result = result + a;
        }
        return result;
}

public static void main(String[] args){
    System.out.println(add(1,2));
    System.out.println(add(2,3,4));
    System.out.println(add(4,5,6));
}
}
```

Output :



```
 3
9
15
```