Ques Take an input "sum". Find number of pairs whose sum is equal to the input.

For example
↳ 5 can be represented as sum of (1,4) and (2,3).
10 can be represented as sum of (1,9) (2,8) (3,7) (4,6) (5,5).

for input 5, output should be 2,
" 10, output should be 5.

(num1 + num2) → sum

↳ count of pairs where $\boxed{num1 + num2 == x}$

$\boxed{1 \to x-1}$ $(1 \to x-1)$

$\boxed{x + \text{positive number}}$ → result

sum 10

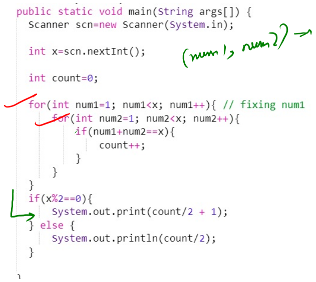
(1, x-1)

(1, x-1)

(1 – x-1)

num1 + num2 ==

num1 ⇒ ~~1~~ 2
↳ num2 → (1 → x-1) [num1 + num2 == x]



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

  int x=scn.nextInt();

  int count=0;

  for(int num1=1; num1<x; num1++){ // fixing num1
     for(int num2=1; num2<x; num2++){
        if(num1+num2==x){
           count++;
        }
     }
  }
  if(x%2==0){
     System.out.print(count/2 + 1);
  } else {
     System.out.println(count/2);
  }
}
```

(num1, num2) → ⇒ x

num1 ⇒ (1, x-1)

num1 ⇒ ~~1 2 3 4~~ 5

$\boxed{x = 6}$

count = ~~0 1 2 3 4~~ 5

6 ⇒ (1,5) (2,4)
(3,3)
(4,2) (5,1)

6 → 5 → $\frac{5}{2} + 1$
↘ 3

5 → 4
↘ 2

(1,4) (2,3)
(3,2) (4,1)

5 → (1,4) (2,3) → 4
(3,2) (4,1)

(num1, num2)

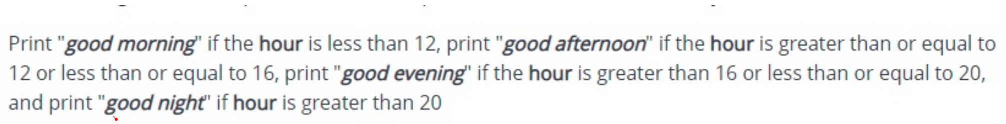
6 ⇒ (1,5) (2,4) (3,3)
(5,1) (4,2)

7 → (1,6) (2,5) (3,4)
(4,3) (5,2) (6,1)

8 ⇒ (1,7) (2,6) (3,5) (4,4)
(5,3) (6,2) (7,1)

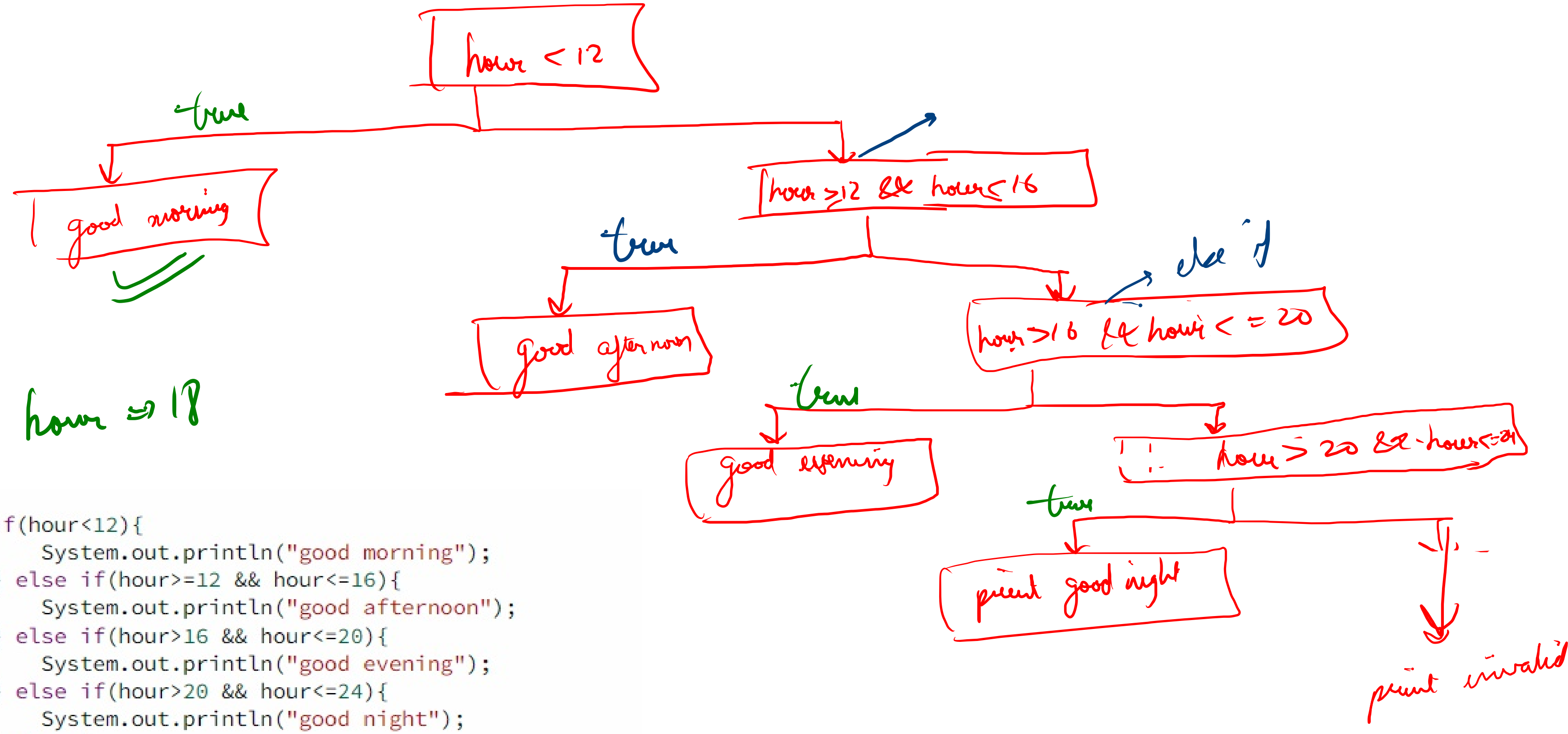
9 → (1,8) (2,7) (3,6) (4,5)
(5,4) (6,3) (7,2) (8,1)

10 ⇒ (1,9) (2,8) (3,7) (4,6) (5,5)
(6,4) (7,3) (8,2) (9,1)



Print "***good morning***" if the **hour** is less than 12, print "***good afternoon***" if the **hour** is greater than or equal to 12 or less than or equal to 16, print "***good evening***" if the **hour** is greater than 16 or less than or equal to 20, and print "***good night***" if **hour** is greater than 20

→ flow diagram → code



hour => 18

```
if(hour<12){
    System.out.println("good morning");
} else if(hour>=12 && hour<=16){
    System.out.println("good afternoon");
} else if(hour>16 && hour<=20){
    System.out.println("good evening");
} else if(hour>20 && hour<=24){
    System.out.println("good night");
} else {
    System.out.println("invalid input");
}
```

1) addition → Jack (int)

2) subtraction → Ram (int)

3) multiplication → geek (int)

functions / methods

↓

input → work → output

data → processing → result.

1) add 2 and 3 → Jack

2) add 5 and 8 → Jack

3) mul 5 and 8 → geek

4) add 2 and 4 → jack

Syntax

return data type

function name

```
public static int addition (int a, int b){
    int sum = a + b;
    return sum;
}
```

parameters / input

work / data processing

→ Reusability.

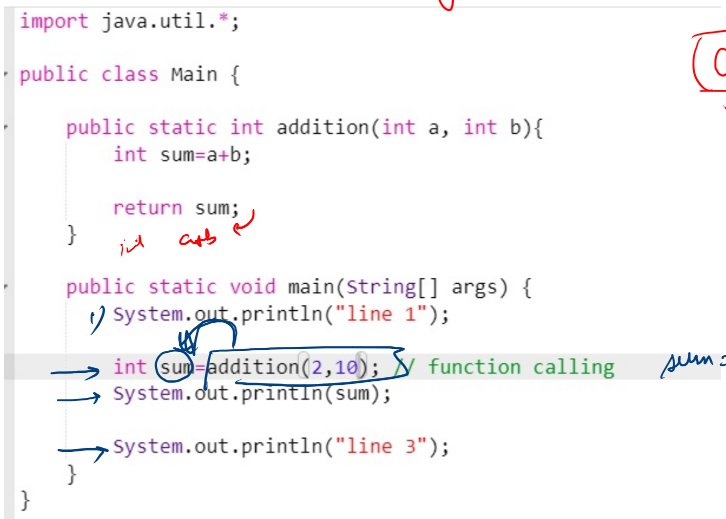
parameters → (data type + name)

void → nothing to return

Java file

int sum = fun_name (2,10);

Compiler



```
import java.util.*;

public class Main {

    public static int addition(int a, int b){
        int sum=a+b;

        return sum;
    }

    public static void main(String[] args) {
        System.out.println("line 1");

        int sum=addition(2,10); // function calling
        System.out.println(sum);

        System.out.println("line 3");
    }
}
```

int args

1)

sum=12

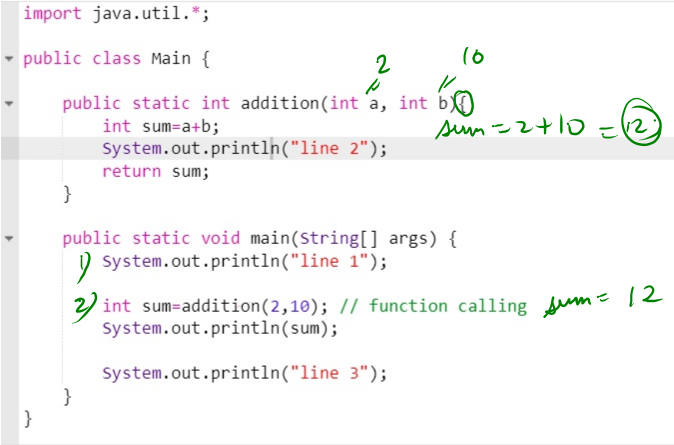
→ line 1

→ 12

→ line 3

find subtraction by using method [return type → int]
[return type → void]

compiler



```
import java.util.*;

public class Main {

    public static int addition(int a, int b){
        int sum=a+b;
        System.out.println("line 2");
        return sum;
    }

    public static void main(String[] args) {
        System.out.println("line 1");

        int sum=addition(2,10); // function calling
        System.out.println(sum);

        System.out.println("line 3");
    }
}
```

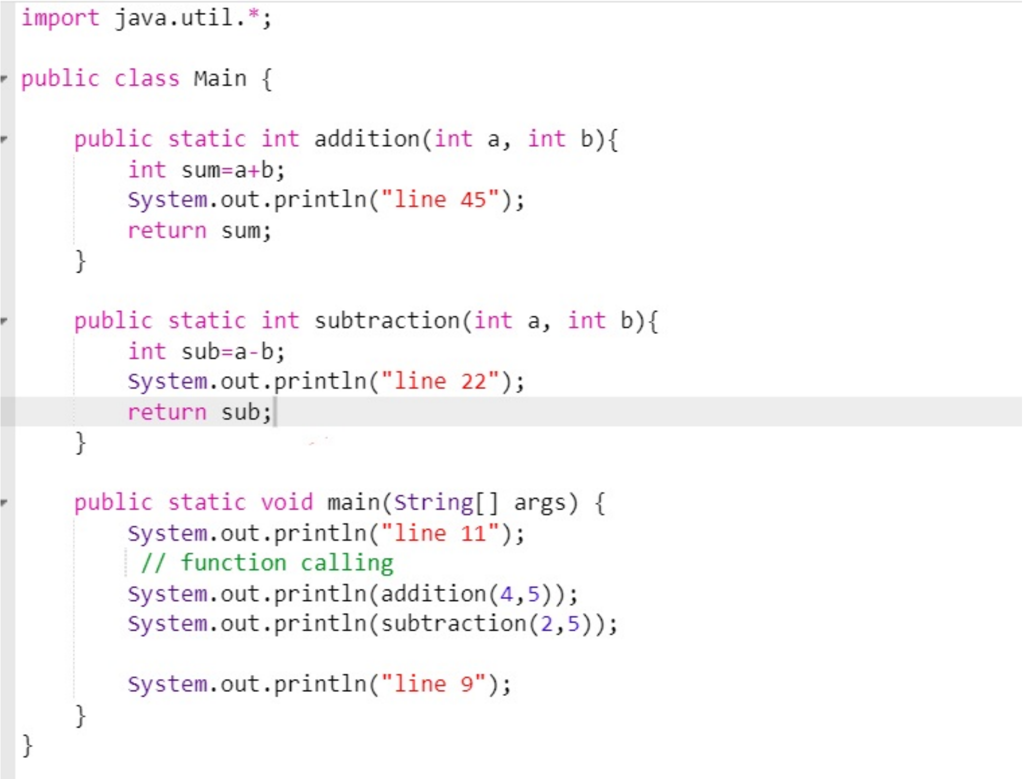
2 10

sum = 2+10 = 12

1)

2) sum= 12

line 1
line 2
12
line 3



```java
import java.util.*;

public class Main {

    public static int addition(int a, int b){
        int sum=a+b;
        System.out.println("line 45");
        return sum;
    }

    public static int subtraction(int a, int b){
        int sub=a-b;
        System.out.println("line 22");
        return sub;
    }

    public static void main(String[] args) {
        System.out.println("line 11");
         // function calling
        System.out.println(addition(4,5));
        System.out.println(subtraction(2,5));

        System.out.println("line 9");
    }
}
```

compiler

line 11

line 45

9

line 22

-3

line 9

1) Find out number of lines
2) Count stars and spaces in every line
3) Find how number of stars and spaces are changing.
4) Code this pattern

Patterns

Take one input = 5

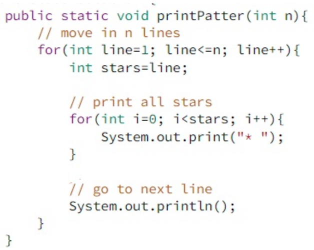
=) 4

```
*
*  *
*  *  *
*  *  *  *
```

n= 5

```
1)  *                     →  1 star
2)  *  *                  →  2 star
3)  *  *  *               →  3 star
4)  *  *  *  *            →  4 star
5)  *  *  *  *  *         →  5 star
```

line number == number of stars ?



```
public static void printPatter(int n){
    // move in n lines
    for(int line=1; line<=n; line++){
        int stars=line;

        // print all stars
        for(int i=0; i<stars; i++){
            System.out.print("* ");
        }

        // go to next line
        System.out.println();
    }
}
```

```
*
*  *
*  *  *
```

```
*
*  *
*  *  *
*  *  *  *
*  *  *  *  *
```

n= 5

```
*  *  *  *  *
*  *  *  *
*  *  *
*  *
*
```