

# CL1002 – Programming Fundamentals Lab



## Lab # 06

### Conditional Structures

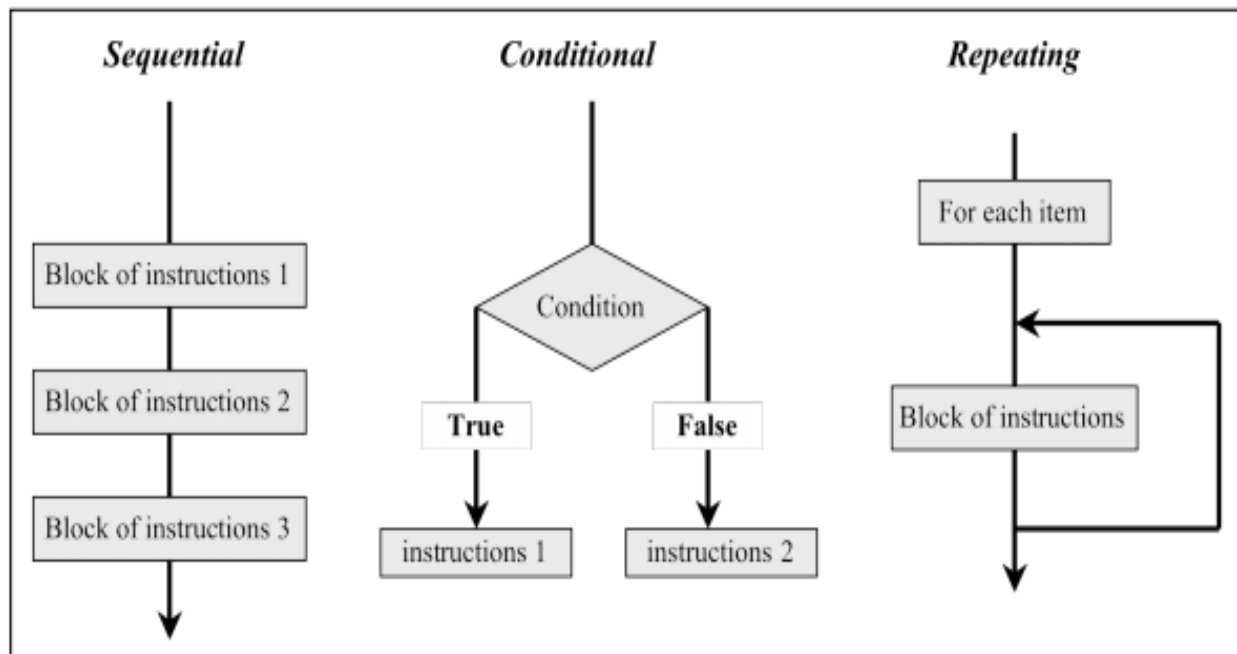
Instructor: Engr. Muhammad Usman

Email: [usman.rafiq@nu.edu.pk](mailto:usman.rafiq@nu.edu.pk)

Department of Computer Science,  
National University of Computer and Emerging Sciences FAST Peshawar

## Control Structures

Algorithms require two important control structures: iteration(repeating) and selection(conditional). Both are supported by C in various forms. The programmer can choose the statement that is most useful for the given circumstance.



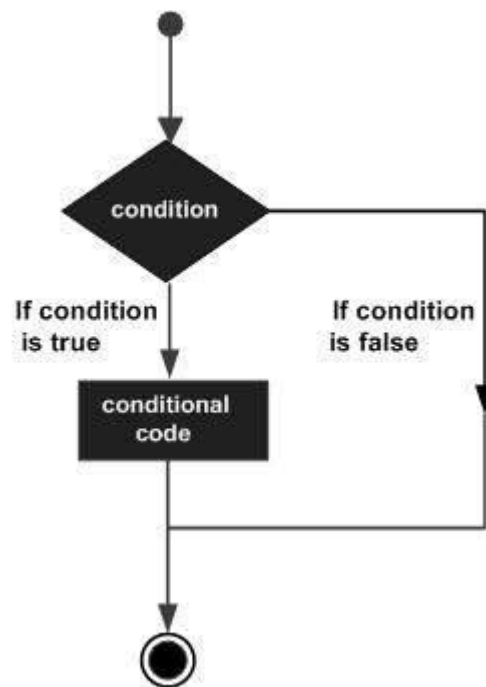
## Selection Statements / Decision Making

There come situations in real life when we need to make some decisions and based on these decisions, we decide what should we do next. Similar situations arise in programming also where we need to make some decisions and based on these decisions we will execute the next block of code.

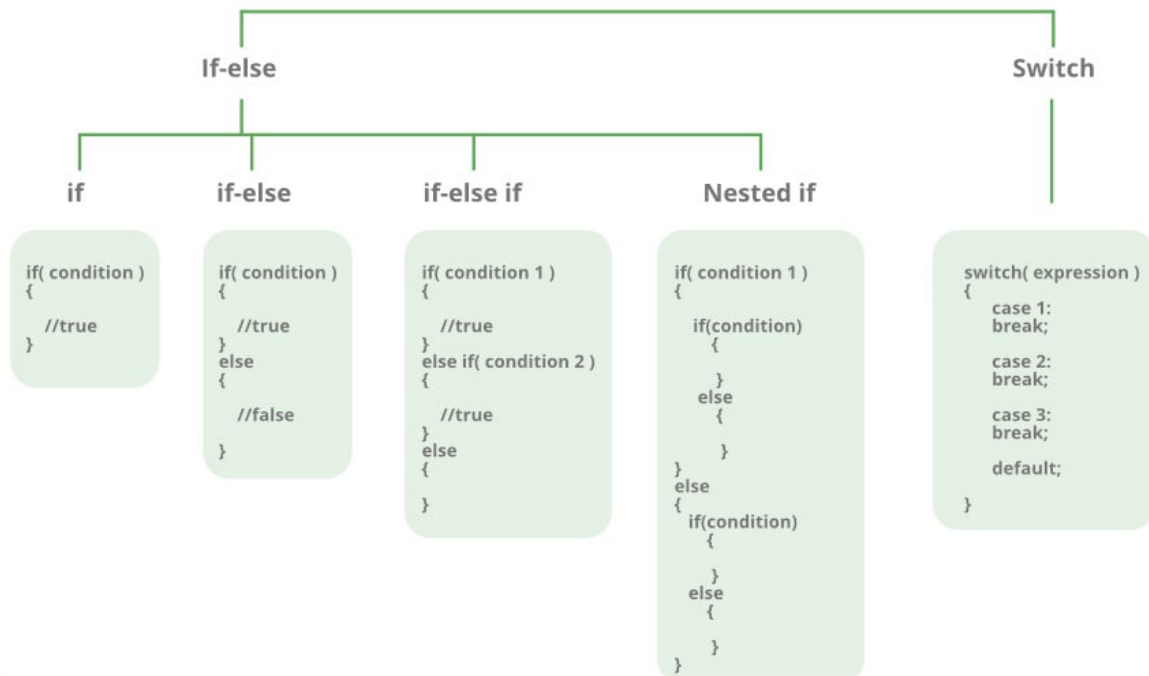
For example, in C if x occurs then execute y else execute z. There can also be multiple conditions like in C if x occurs then execute p, else if condition y occurs execute q, else execute r. This condition of C else-if is one of the many ways of importing multiple conditions.

Decision making is the most important aspect of almost all the programming languages. As the name implies, decision making allows us to run a particular block of code for a particular decision. Here, the decisions are made on the validity of the particular conditions. Condition checking is the backbone of decision making.

Following is the general form of a typical decision making structure found in most of the programming languages.



## Decision Making



## C if else Statement

The if-else statement in C is used to perform the operations based on some specific condition. The operations specified in if block are executed if and only if the given condition is true.

There are the following variants of if statement in C language.

- If statement
- If-else statement
- If else-if ladder
- Nested if

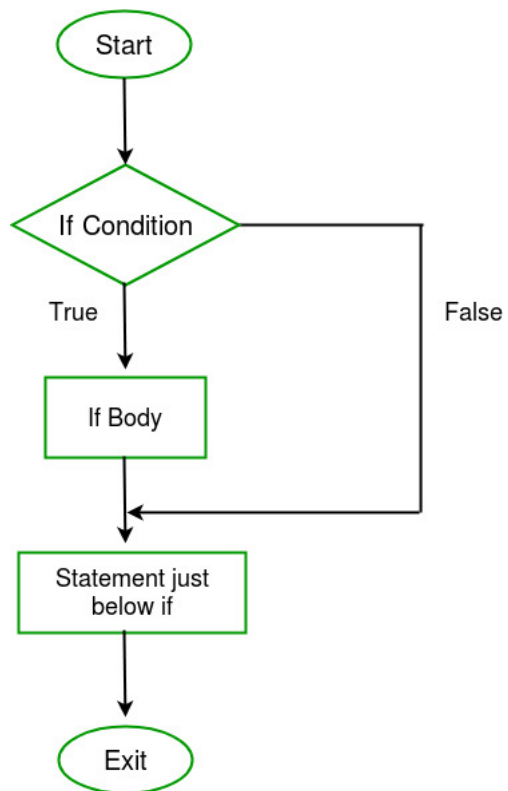
### 1. If Statement in C/C++ (One-Way Decision)

if statement is the most simple decision-making statement. It is used to decide whether a certain statement or block of statements will be executed or not i.e if a certain condition is true then a block of statement is executed otherwise not.

#### Syntax:

```
if(condition)
{
    // Statements to execute if
    // condition is true
}
```

#### Flowchart of if statement in C



### Example 1

```
// C program to illustrate If statement
#include <stdio.h>
int main()
{
    int i = -10;
    if (i > 0) {
        printf("i is positive ");
    }
    printf("I am Not in if");
}
```

### Output

I am Not in if

As the condition present in the if statement is false. So, the block below the if statement is not executed.

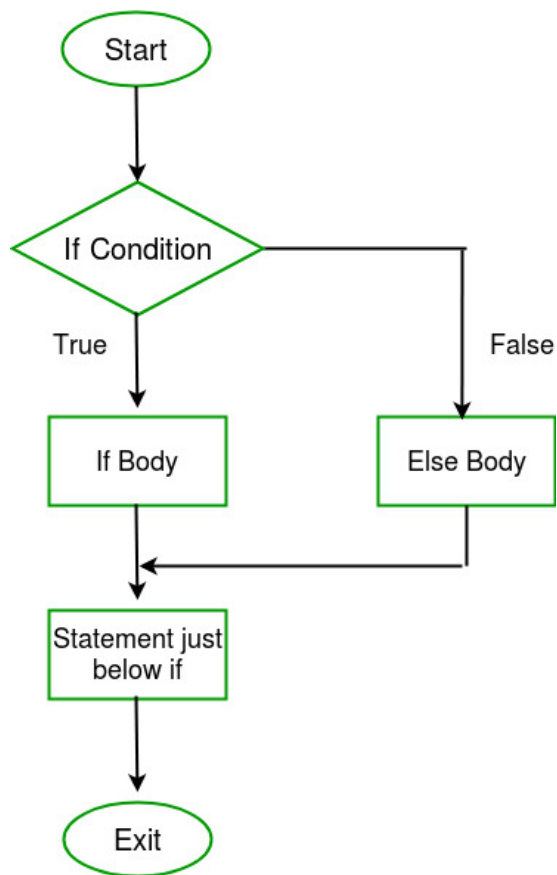
## 2. if-else in C/C++ (Two-Way Decision)

The if statement alone tells us that if a condition is true it will execute a block of statements and if the condition is false it won't. But what if we want to do something else if the condition is false. Here comes the C else statement. We can use the else statement with the if statement to execute a block of code when the condition is false.

### Syntax:

```
if (condition)
{
    // Executes this block if
    // condition is true
}
else
{
    // Executes this block if
    // condition is false
}
```

### Flowchart:



## Example 2

```
// C program to illustrate If statement
#include <stdio.h>

int main()
{
    int i = 20;

    if (i >= 0) {

        printf("i is positive or zero");
    }
    else {

        printf("i is negative");
    }

    return 0;
}
```

```
}
```

## Output

```
i is positive or zero
```

The block of code following the else statement is executed as the condition present in the if statement is false.

### 3. if-else-if ladder in C/C++ (Multi Way Decisions)

Here, a user can decide among multiple options. The C if statements are executed from the top down. As soon as one of the conditions controlling the if is true, the statement associated with that if is executed, and the rest of the C else-if ladder is bypassed. If none of the conditions is true, then the final else statement will be executed.

#### Syntax:

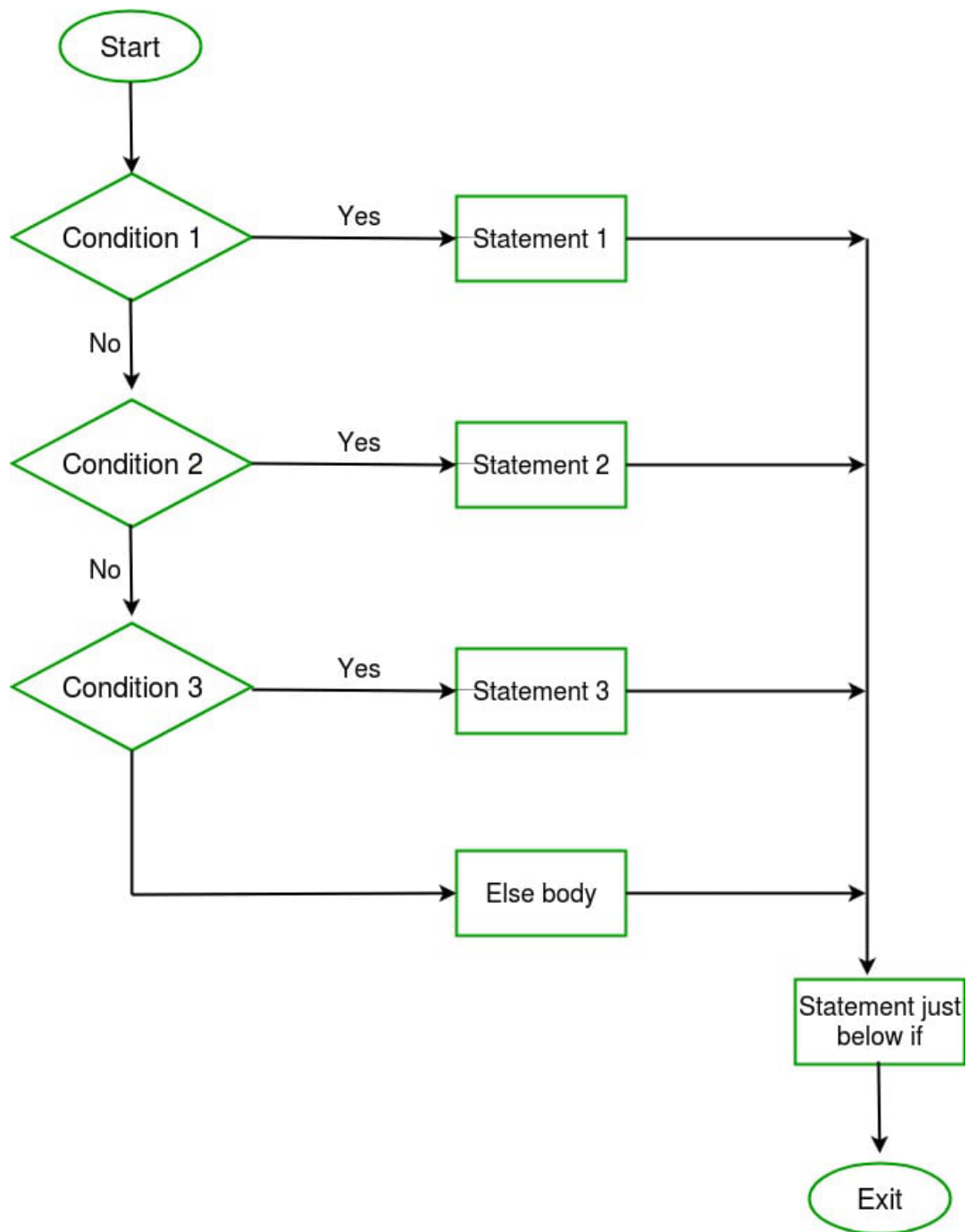
```
if (condition)
    statement;

else if (condition)
    statement;

.
.

else
    statement;
```





### Example 3:

```
// C program to illustrate If elseif statement
#include <stdio.h>
int main()
{
    int i = -10;
```

```
if (i > 10){  
    printf("i is Positive");  
}  
else if (i == 0){  
    printf("i is zero");}  
else{  
    printf("i is negative");}  
}
```

### Output

i is negative

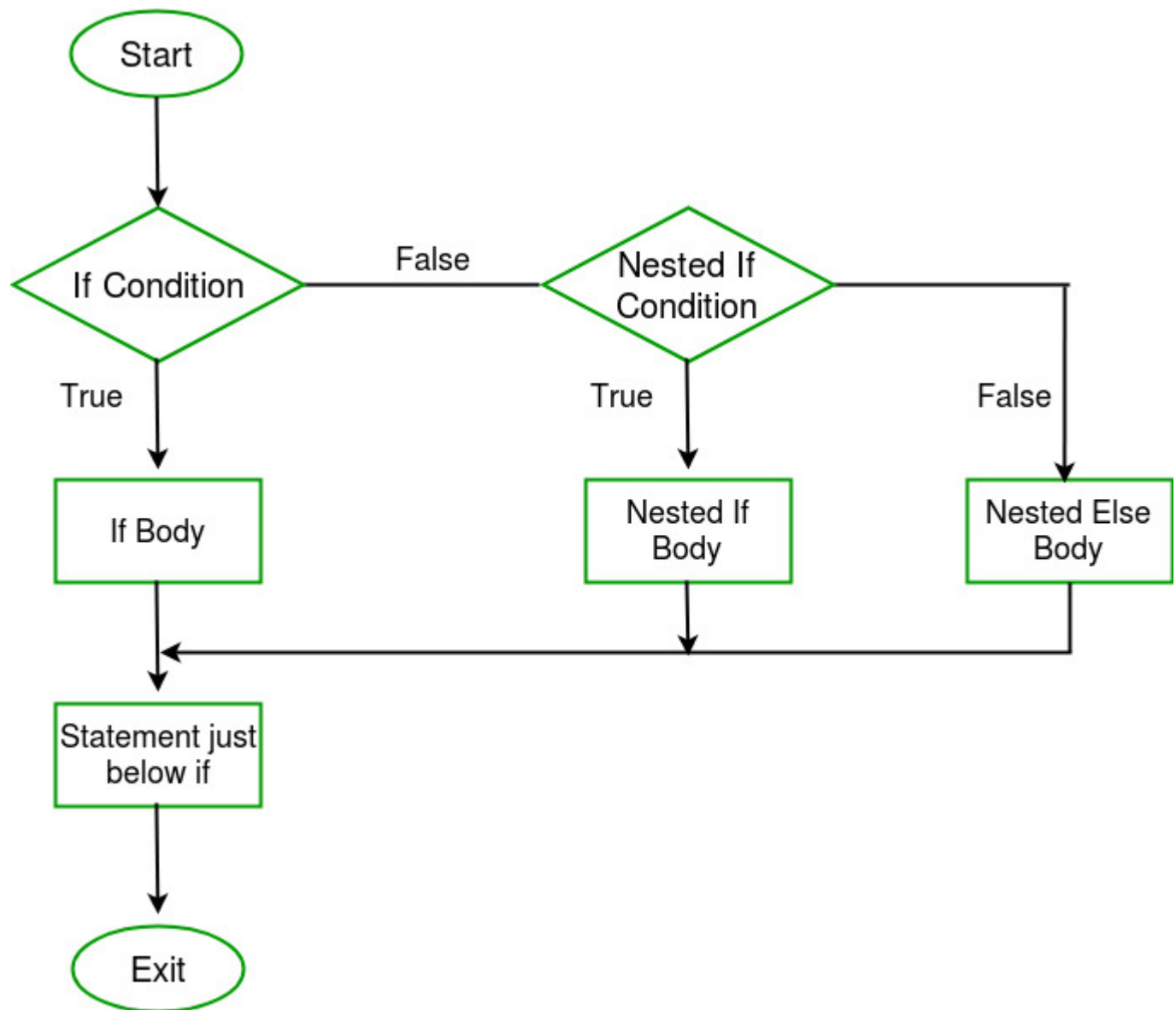
## 4. nested-if in C/C++

A nested if in C is an if statement that is the target of another if statement. Nested if statements mean an if statement inside another if statement. Yes, both C and C++ allow us to nested if statements within if statements, i.e, we can place an if statement inside another if statement.

### Syntax:

```
if (condition1)  
{  
    // Executes when condition1 is true  
    if (condition2)  
    {  
        // Executes when condition2 is true  
    }  
}
```

### Flowchart



#### Example 4:

```
// C program to illustrate nested-if statement
#include <stdio.h>

int main()
{
    int i = 10;

    if (i > 0) {
        // First if statement
```

```

    if (i < 15)
        printf("i is smaller than 15\n");
    // Nested - if statement
    // Will only be executed if statement above
    // is true
    if (i < 12)
        printf("i is smaller than 12 too\n");
    else
        printf("i is greater than 15");
}

return 0;
}

```

### Output

i is smaller than 15

i is smaller than 12 too

## C Logical Operators

An expression containing logical operator returns either 0 or 1 depending upon whether expression results true or false. Logical operators are commonly used in decision making in C programming.

| Operator | Meaning   | Example  |
|----------|---|--|
| &&       | Logical AND. True only if all operands are true     | If c = 5 and d = 2 then, expression ((c==5) && (d>5)) equals to 0. |
|          | Logical OR. True only if either one operand is true | If c = 5 and d = 2 then, expression ((c==5)    (d>5)) equals to 1. |
| !        | Logical NOT. True only if the operand is 0          | If c = 5 then, expression !(c==5) equals to 0.                     |

### Example 5

```
// program that grant access to children aged between 8 - 12
#include <stdio.h>
int main()
{
    int age = 18;

    if(age>=8 && age<=12) {
        printf("You are allowed Welcome!\n");
    }
    else{
        printf("Sorry! You are not allowed. Bye !\n");
    }

    return 0;
}
```

### Output

Sorry! You are not allowed. Bye !

### Example 6

```
#include <stdio.h>
int main()
{
    int a = 5,b=-3;

    if(a>0 || b>0) {
        printf("Either of the number is greater than 0\n");
    }
    else{
        printf("No number is greater than 0\n");
    }

    return 0;
}
```

## Output

Either of the number is greater than 0

## References:

<https://www.geeksforgeeks.org/decision-making-c-cpp/>

<https://www.javatpoint.com/c-if-else>