VTC Academy
Think Ahead, Beyond Boundaries

BASIC PROGRAMMING LANGUAGE

LESSON 4

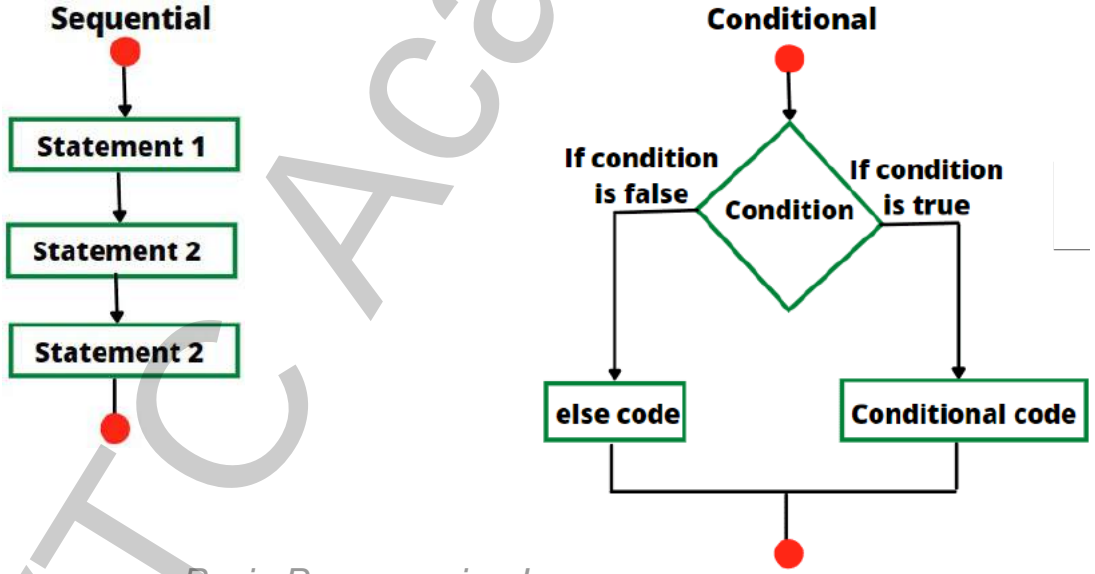
Decision Making Statements

CONTENT

VTC Academy
Think Ahead, Beyond Boundaries

Decision Making Statements

- Decision making statements enable us to change the flow of the program.
- Decision making statements in programming languages decide the direction of the flow of program execution base on a condition true or false.

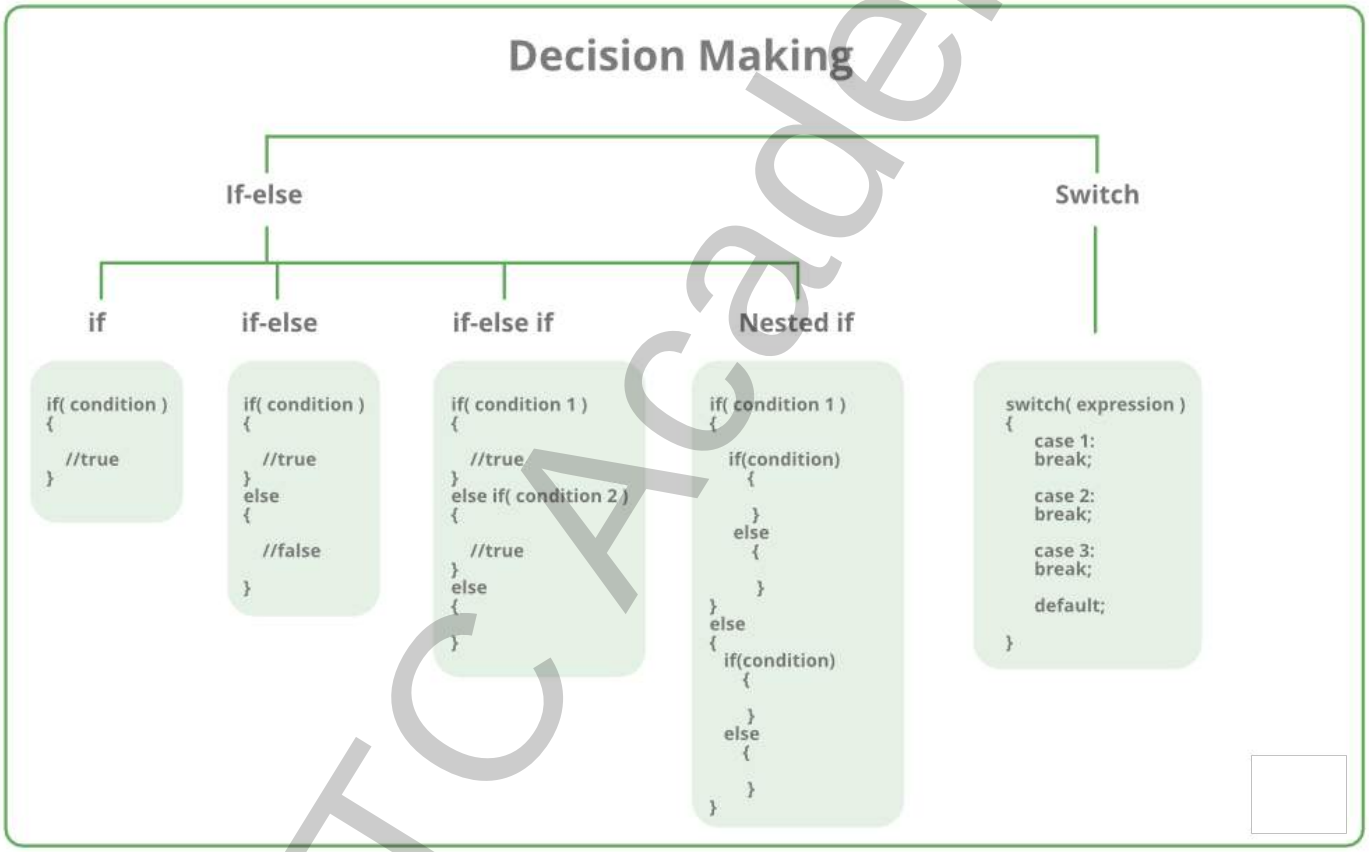


Decision Making Statements

VTC Academy
Think Ahead, Beyond Boundaries

- Decision making statements available in C are:
  - `if` statement
    - `if...else` statement
    - Multi `if` statement
    - Nested `if` statement
  - `switch...case` statement
    - `break` statement

Decision Making Statements



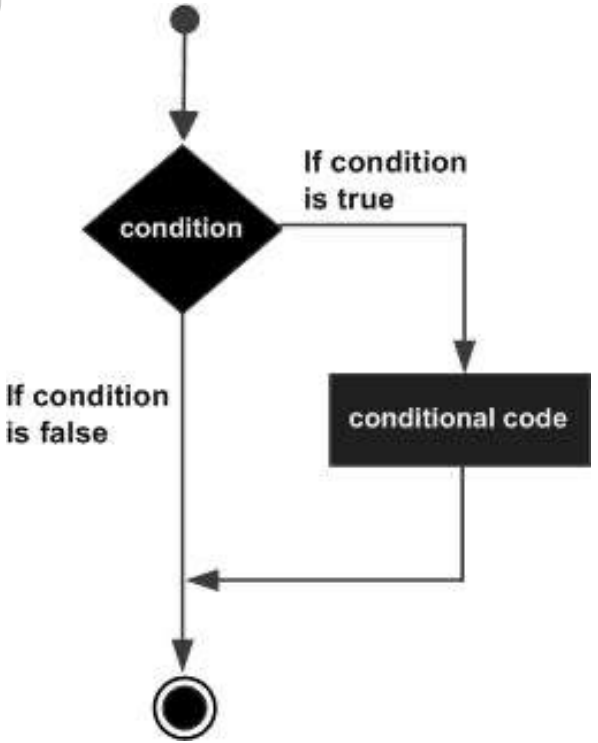
if Statement

- If statements in C programming are used to make decisions based on the conditions.
- C programming language assumes any non-zero and non-null values as true, and if it is either zero or null, then it is assumed as false value.
- Types of `if` statement:
  - `if` statement
  - `if...else` statement
  - Multi `if` statement
  - Nested `if` statement

if Statement

- If boolean expression is true, the code in `if` block will be executed.
- `if` statement syntax:

```
if (boolean_expression) {
    /* execute if the boolean expression is
    true */
}
```



if Statement Example

VTC Academy
Think Ahead, Beyond Boundaries

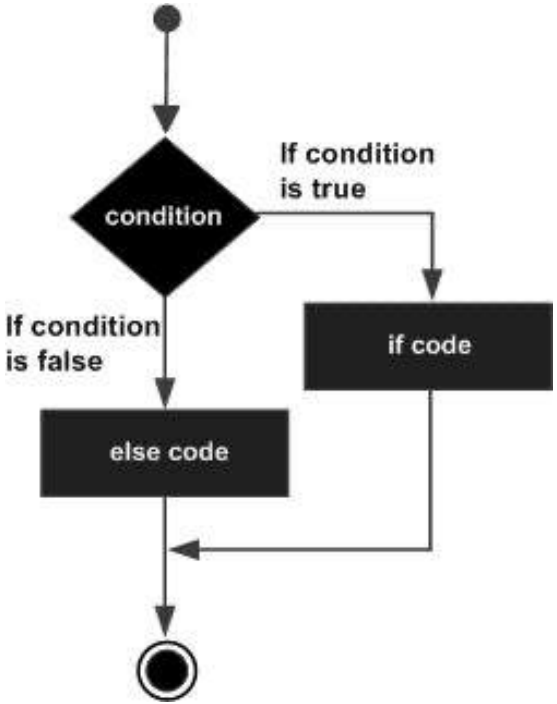
```c
#include <stdio.h>

int main ()
{
    int a = 10;
    if (a < 20)
    {
        printf("a is less than 20\n");
    }
    printf("Value of a is %d\n", a);
    return 0;
}
```

if…else Statement

- An `if` statement can be followed by an optional `else` statement, which executes when the boolean expression is `false`.
- Syntax:

```
if (boolean_expression) {
    /* execute if the boolean expression
    is true */
}
else {
    /* execute if the boolean expression
    is false */
}
```



if…else Statement

- The `if` expression evaluates to true, the block following the `if` statement or statements are executed.
- If the `if` expression does not evaluate to `true` then the statements following the else expression take over control.
- The `else` statement is optional. It is used only `if` a statement or a sequence of statements are to be executed in case the `if` expression evaluates to `false`.

if…else Statement Example

VTC Academy
Think Ahead, Beyond Boundaries

```c
#include <stdio.h>
int main()
{
    int a = 50;
    if (a < 30)
    {
        printf("a is less than 30\n");
    }
    else
    {
        printf("a is not less than 30\n");
    }
    printf("Value of a is %d\n", a);
    return 0;
}
```

if...else if...else Statement

VTC Academy
Think Ahead, Beyond Boundaries

- An if statement can be followed by an optional else if...else statement, which is very useful to test various conditions using single `if...else if` statement.
- When using `if...else if..else` statements:
  - An `if` can have zero or one else's and it must come after any else if's.
  - An `if` can have zero to many else if's and they must come before the else.
  - Once an else if succeeds, none of the remaining else if's or else's will be tested.

if...else if...else Statement

VTC Academy
Think Ahead, Beyond Boundaries

- Syntax:

```
if (boolean_expression_1) {
    // execute if the boolean expression 1 is true
}
else if (boolean_expression_1) {
    // execute if the boolean expression 2 is true
}
else {
    // executes when the none of the above condition is true
}
```

if...else if...else Statement Example

VTC Academy
Think Ahead, Beyond Boundaries

```c
#include <stdio.h>
int main()
{
    int a = 1;
    if (a == 3) {
        printf("Value of a is 3\n");
    } else if (a == 3) {
        printf("Value of a is 6\n");
    } else if( a == 30 ) {
        printf("Value of a is 9\n");
    } else {
        printf("None of the values is matching\n");
    }
    printf("Exact value of a is %d\n", a);
    return 0;
}
```

Nested if

- Nested `if` statements in C programming mean that you can use `if` or `if…else` statements inside another `if` or `else` block.
- Syntax:

```
if (boolean_expression_1) {
    // nested if
    if ((boolean_expression_2)) {
        /* execute if the boolean expression 1 & 2 are true */
    }
}
else {
    /* execute if the boolean expression 1 is false */
}
```

Nested if

- The nested `if` is an `if` statement, which is placed within another `if` or `else`.
- In C, an `else` statement always refers to the nearest `if` statement that is within the same block as the else statement and is not already associated with an `if`.

Nested if Example

VTC Academy
Think Ahead, Beyond Boundaries

```c
#include <stdio.h>
int main() {
    int a = 100;
    int b = 200;
    if (a == 100) {
        if (b == 200) {
            printf("Value of a is 100 and b is 200\n");
        }
    }
    printf("Exact value of a is %d\n", a);
    printf("Exact value of b is %d\n", b);
    return 0;
}
```

switch Statement

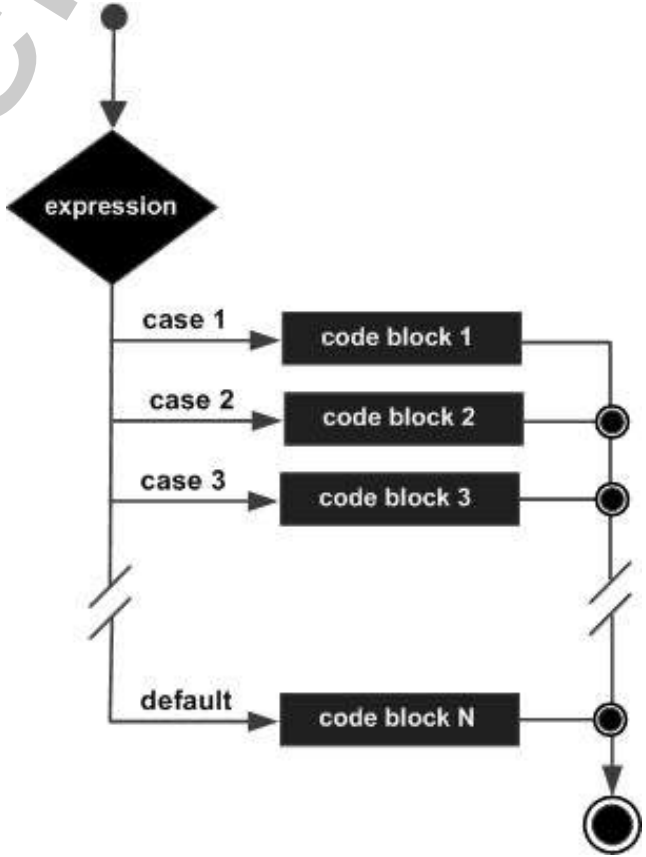
VTC Academy
Think Ahead, Beyond Boundaries

- A switch statement allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each switch case.
- The switch statement is a multi-way decision maker that tests the value of an expression against a list of integers or character constants
- When a match is found, the statements associated with that constant are executed

switch Statement

- Syntax:

```
switch (expression) {
    case constant_expression_1:
            statement(s);
            break;
    case constant_expression_2:
            statement(s);
            break;
    default:
            statement(s);
}
```



break Statement

VTC Academy
Think Ahead, Beyond Boundaries

- When a `break` statement is reached, the switch terminates, and the flow of control jumps to the next line following the `switch` statement.
- Not every case needs to contain a break. If no break appears, the flow of control will fall through to subsequent cases until a break is reached.
- The default case can be used for performing a task when none of the cases is true. No break is needed in the default case.

switch Statement Example

```c
char grade = 'B';
switch(grade) {
    case 'A':
        printf("Excellent!\n");
        break;
    case 'B':
    case 'C':
        printf("Well done\n");
        break;
    case 'D':
        printf("You passed\n");
        break;
    case 'F':
        printf("Better try again\n" );
        break;
    default:
        printf("Invalid grade\n");
}
```

Summary

- Decision making statements in programming languages decide the direction of the flow of program execution base on a condition true or false.
- Decision-making statements available in C are:
  - `if` statement
    - `if...else` statement
    - Multi `if` statement
    - Nested `if` statement
  - `switch...case` statement
    - `break` statement

VTC Academy
Think Ahead, Beyond Boundaries