Control Statements in C

Control statements in C enable the user to execute a particular code block conditionally. C supports decision control statements that can alter the flow of a sequence of instructions. The control flow statements allow the user to execute the code as per the requirements without executing it in sequential order.

It is expected that questions can be formulated based on the control Statements in C in the GATE CSE exam. Control statements hinder the sequential flow of execution; using control statements in C programming, we can branch to a particular code, iterate a particular segment of code or jump to a segment of code based on the conditions.

Types of Control Statements in C

In C programming, we have three types of control statements. They allow the user to execute a particular code segment and skip the rest based on the programmer's need. There are three types of control statements in C, they are as follows:

- Decision Control Statements
- Iterative Statements
- Jump Statements

Decision control or branching means deciding what actions to be taken, looping means deciding how many times a particular code will execute, and jump statements mean transferring the control from one point to another. Now we shall discuss each control statement briefly to understand their functionality and workflow in the program

Decision Control Statements in C

The decision control statements in C help jump from one part of the program code to the other depending on whether the condition is satisfied. Having discussed the meaning and need of control statements in C, now let us discuss various decision control statements available in C for the GATE CSE syllabus. These decision control statements include:

- The "if" Statement
- The "if-else" statement
- The "if-else-if" statement
- Switch-case statement

The "if" Statement

The "if" statement is the simplest decision control statement frequently used to transfer the flow of control from one part to the other part of the code. The if block may include one statement or n statements enclosed within curly brackets.

Example of a C program with an if statement:

```
#include< stdio.h>

int main()

{

int a=10;

if(a>0)

a++;

printf(“%d”, a);

return 0;

}
```

Output: 11

The “if-else” Statement

In the if-else construct, first, the test expression is evaluated. If the expression is true, then if the block is executed otherwise, the else block is executed. The if-else construct is different from that of the if statement. An example of an if-else block is as follows:

```
#include< stdio.h>

int main()

{

int a=10;

if(a%2==0)

printf(“%d”, a is even);

else

printf(“%d”, a is odd);

return 0;

}
```

Output: a is even

The “if-else-if” Statement

In the ”if-else-if” construct, first, the test expression is evaluated. If the expression is true, then if the block is executed otherwise, the else block is executed; it supports an additional condition apart from the initial test condition. The “if-else-if” construct works similarly to a normal if statement.

Example:

```
#include< stdio.h>

int main()

{

int number;

printf(“Enter any number”);

scanf(“%d”, &number);

if(number==0)

printf(“The value is equal to zero”);

else if(number>0)

printf(“The value is positive”);

else

printf(“The value is negative”);

return 0;

}
```

Switch Case

A switch case statement is a multi-way decision statement that is an alternative to the if-else-if construct. The switch statements are mostly used in two scenarios:

- When there is only one variable to evaluate in the expression.
- When many conditions are tested simultaneously.

The general form or syntax of a switch case is as follows:

```
switch(variable)

{
```

```
case value 1:

statement block 1;

break;

case value 2:

statement block 2;

break;

case value N:

statement block N;

break;

default:

statement block D;

break;

}

Statement X;
```

Iterative Statements in C

The iterative statements are used to repeat the execution of a particular sequence of instructions until the specified expression becomes false. The C language supports three types of iterative statements also known as looping statements. The iterative statements present in C are as follows:

- The “for” loop
- The “while” loop
- The “do-while” loop

Jump Statements in C

To transfer the control from one block to another, we use jump statements; for transferring control, we have two types of jump statements: broken and continuous. The jump statements are also used in questions asked in the GATE exam. In C programming language, the break statement is used to terminate the execution of the nearest enclosing loop in which it appears.

The break statement is widely used in decision control statements and iterative statements. It is frequently used in switch cases and iterative statements like for, while, and do-while.