Control Structures in C
Part - I

FAQs

- **What you mean by selective control structures?**
  The selective constructs are used to select statements for execution based on the output of the conditional test given by the expression. C supports 4 selective constructs including conditional expression, if-else, if-else-if and switch-case construct.

- **Explain in detail the use of if-else construct in C.**
  Suppose we want to execute one group of statements if the expression evaluates to true and another group of statements if the expression evaluates to false then we can use **if**-**else** statement.
  The general form of if-else construct is :

```
if (test expression)
{
        True-block statements
}
else
{
        False-block statements
}
```

  If the test expression is true, then the true-block statements, immediately following the **if** statement are executed; otherwise, the false block statements are executed. Here either true block or false block will be executed, not both.

- **What you mean by nested if-else construct?** Explain its usage.
  When a series of decisions are involved in a program, we may have to use more than one if-else statement in the nested form. When an if-else construct appears as a statement in another if-else, it is known as nested if-else construct. General form of if-else construct is shown in the screen.

```
if(test condition a)
{
        if(test condition b)
        {
                Statement 1;
        }
        else
        {
                Statement 2;
        }
}
else
{
        Statement 3;
}
```

  If the ***condition a*** is false then the statement 3 will be executed; otherwise it continues to perform the second test. If the ***condition b*** is true the statement 1 will be executed; otherwise statement 2 will be evaluated and then the control is transferred to the statement x.

- **Illustrate the use of turnery expressions.**

  The conditional operators **?** and **:** named **ternary operator** since they take three arguments. The general form of turnery expression is,

  expression 1 ? expression 2 : expression 3

What this expression says is: “if **expression 1** is true (that is, if its value is non-zero), then the value returned will be **expression 2**, otherwise the value returned will be **expression 3**”.

.

- **Explain the use of switch-case construct in C.**
  Switch-case construct is a multi-way decision statement in C. The **switch** statement testes the value of a given expression against a list of case values and the block of statements associated with the matching case value is executed. Three keywords **switch, case,** and **default,** go together to make up the control statement.

  The general format is:

```
switch (expression )
{
case 1 :
statements;
        break;
case 2 :
statements;
        break;

case 3 :
statements;
        break;

default :
statements ;
        break;
}
```

  Each **case** is labeled by an integer or character expression yielding an integer value known as case labels. The expression is evaluated first and its value is then matched against the case labels. If a case matches the expression value, execution starts at that case. All case expressions must be different. The case labeled default is executed if none of the other cases are satisfied. A **default** is optional; if it isn't there and if none of the cases match, no action at all takes place. Cases and the default clause can occur in any order.
  If a break is not included in each case statement, whenever a match is found, the program executes the statements following the case and also all the subsequent case statements and default statements. Even though a break statement is not necessary in the default part, it is a good programming practice to add it always.
  The colon (:) must be placed at the end of each case label and default. The braces following the labels are optional.