Date-31/03/2017

Q] What is an operator? Why is it required in programming?

> An operator is a symbol which tells the compiler which operation to perform on the data depending on the type of operators used (Arithmetic, Logical, Conditional, etc.).It represents an action to be performed on the given data.

E.g. (4) + (5)…here the numbers 4 and 5 are operands and ‘+’ is the operator which represents ‘addition’….so compiler will add 4 and 5 and will give the result.\

Q] What is conditional construct? Why is it required in programming?

>It is a feature of programming where the flow of program depends upon whether the conditions specified evaluates to true or false. Depending on that result the compiler proceeds further. It basically transfers execution control to the correct path based on comparison result.

If any program or scenario needs the data to be compared and move forward OR if it needs to satisfy a condition to move forward…we use conditional construct. I.e. it will move forward only if the condition is verified and satisfied.

E.g. A person can vote only if he/she is 18 years old…otherwise he/she is not eligible for voting altogether.

Q] What is looping construct? Why is it required in programming?

>In a program certain aspects might be repetitive … so to save the programmer time and to make the program less bulky we put those common statements in a loop which keeps on executing itself till the outer condition which defines the loop is true. After the outer constraint becomes false the control exits the loop.

Thus looping saves the programmer from writing code multiple times that has repetitive in nature. It helps us execute a portion of a program only once, based on a condition OR execute a portion of the program repetitively, based on condition.

Q] What are different phases of looping?

> Any basic structure of a loop has 2 phases-

1. Test Condition 2. Body of Loop

Test condition: Specifies the condition to be satisfied to enter/execute the loop. The test condition may or may not have 3 sections (initialization, constraints/limits, iteration) depending upon the type of loop used.

E.g. While loop has only single condition constraints like

Do While (I<10) {

Loop body

}

For loop has 3 sections

For (i=0, i<=10, i++){

Loop body

}

Body of loop: Contains instructions/codes to be executed when the condition is satisfied.

If the outer condition is not satisfied…the compiler will skip the loop altogether and move forward.

Q] Difference between switch case and if else?

>

|  |  |
| --- | --- |
| If-else | Switch |
| Which statement will be executed depend upon the output of the expression inside if statement. | Which statement will be executed is decided by user input. |
| if-else statement uses multiple statement for multiple choices | switch statement uses single expression for multiple choices. |
| Either if statement will be executed or else statement is executed. | switch statement executes one case after another till a break statement is appeared or the end of switch statement is reached. |
| If the condition inside if statements is false, then by default the else statement is executed if created. | If the condition inside switch statements does not match with any of cases, for that instance the default statements is executed if created. |

Q] What is difference between while, do..while and for loop ?

>

|  |  |  |
| --- | --- | --- |
| While loop | Do..while loop | For loop |
| while ( condition) { statements; //body of loop } | do{ . statements; // body of loop. . } while( Condition ); | for(initialization; condition; iteration){ //body of 'for' loop } |
| The iterations do not occur if, the condition at the first iteration, appears false. | The iteration occurs at least once even if the condition is false at the first iteration. | iteration statement is written at top, hence, executes only after all statements in loop are executed. |
| controlling condition appears at the start of the loop. | controlling condition appears at the end of the loop. | Controlling condition is mentioned at start after intialization |