# Chapter-1 Review of C++ Programming

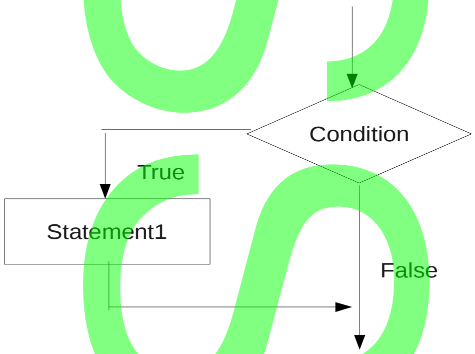
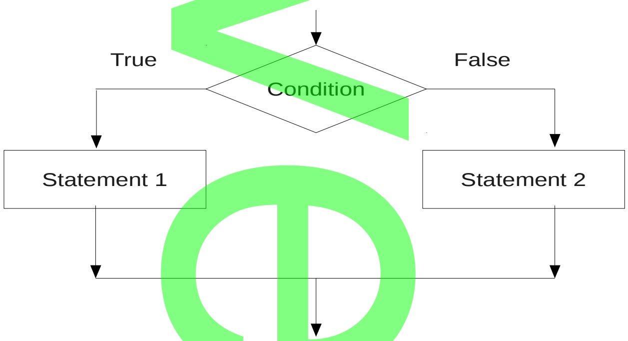
| | |
|---|---|
| **Introduction to C++** | |
| C++ is an Object Oriented Programming Language(OOP) developed by Bjarne Stroustrup. | യാനേ സ്ട്രോസ്ട്രൂപ്പ് എന്ന ശാസ്ത്രജ്ഞൻ വികസിപ്പിച്ച ഓബ്ജക്ട് അധിഷ്ഠിത പ്രോഗ്രാമിംഗ് ഭാഷയാണ് C++. |
| **Character Set** | |
| A character set is a set of characters a language can identify. It is also known as the alphabet of a programming language. They include letters, digits, special symbols, white spaces etc. | ഒരു ഭാഷയിൽ ഉപയോഗിക്കുന്ന വിവിധതരം അക്ഷരങ്ങളും ചിഹ്നങ്ങളും ക്യാരക്ടർ സെറ്റ് എന്ന് അറിയപ്പെടുന്നു. ഇതിൽ അക്ഷരങ്ങൾ, അക്കങ്ങൾ, ചിഹ്നങ്ങൾ, സ്പെയ്സ് എന്നിവ ഉൾപ്പെടുന്നു. |
| **Tokens** | |
| Tokens are the fundamental building blocks of a program. They are also known as Lexical unit. There are five types of tokens in C++. They includes, Keywords, Identifiers, Literals, Punctuators and Operators. | പ്രോഗ്രാമിൽ ഉപയോഗിക്കുന്ന അടിസ്ഥാനപരമായ വാക്കുകളെ ടോക്കൺ എന്ന് വിളിക്കുന്നു. C++ ൽ കീവേർഡ്, ഐഡന്റിഫയർ, ലിറ്ററൽ, പങ്ച്വേറ്റർ, ഓപ്പറേറ്റർ എന്നീ അഞ്ച് ടോക്കണുകളാണുള്ളത്. |
| **Keywords** | |
| Keywords convey a special meaning to the compiler. They are also known as reserved words. They are 63 keywords in C++. | കംപയിലർ പ്രത്യേക ഉപയോഗത്തിനായി മാറ്റിവച്ചിരിക്കുന്ന വാക്കുകളാണ് കീവേർഡ്. C++ ൽ 63 കീവേർഡുകളാണുള്ളത്. |
| **Identifiers** | |
| An identifier refers to the name of variable, function, array, class etc. The identifier assigned to memory location is called a variable. The identifier assigned to a statement is called a label.<br>Rules for naming an identifier<br>1) Keywords cannot be used as identifier.<br>2) It must begin with a letter or underscore.<br>3) Special symbols should not be used (except underscore).<br>4) Upper case and lower case letters are treated separately(case sensitive).<br>5) only characters, digits and under score are allowed in the name of identifier. | വേരിയബിൾ, ഫങ്ഷൻ, അറേ, ക്ലാസ് തുടങ്ങിയവയുടെ പേരുകളാണ് ഐഡന്റിഫയർ. മെമ്മറിക്ക് നൽകുന്ന ഐഡന്റിഫയറിനെ വേരിയബിൾ എന്നും സ്റ്റേറ്റ്മെന്റിന് നൽകുന്ന ഐഡന്റിഫയറിനെ ലേബൽ എന്നും വിളിക്കുന്നു.<br>ഐഡന്റിഫയറിന് പേര് നൽകാനുള്ള നിയമങ്ങൾ.<br>1. കീവേർഡുകൾ ഐഡന്റിഫയറാകാൻ പാടില്ല.<br>2. ആൽഫബെറ്റ്, അണ്ടർസ്കോർ എന്നിവയിൽ മാത്രമേ തുടങ്ങാവൂ<br>3.ചിഹ്നങ്ങൾ ഉപയോഗിക്കാൻ പാടില്ല<br>4.വലിയ അക്ഷരങ്ങളും ചെറിയ അക്ഷരങ്ങളും രണ്ടായിട്ടാണ് കണക്കാക്കുന്നത്.<br>5.ആൽഫബെറ്റ്, അണ്ടർസ്കോർ, അക്കങ്ങൾ എന്നിവ മാത്രമേ ഉപയോഗിക്കാവൂ |
| **Literals** | |
| Literals are data items whose value do not change during the execution of a program. They are also called constants. There are four types of constants in C++ | മൂല്യത്തിൽ മാറ്റം വരാത്ത ഡേറ്റായാണ് ലിറ്ററൽ. ഇവ കോൺസ്റ്റന്റ് എന്നും അറിയപ്പെടുന്നു. C++ ൽ ലിറ്ററലുകൾ നാല് തരത്തിലുണ്ട്. |
| **Integer constants** - An interger constant represents a whole number (ie , a number without fractional part). There are three types of integer constants, Decimal constant, Octal constant and Hexadecimal constant. | ദശാംശമില്ലാത്ത പൂർണ്ണസംഖ്യകളെയാണ് interger constant സൂചിപ്പിക്കുന്നത്. Decimal constant, Octal constant and Hexadecimal constant എന്നീ മൂന്നുവിധത്തിൽ integer constant ഉണ്ട്. |
| **Character constant** - A character constant is enclosed within single quotation mark. For example 'a','B' etc. | single quotation mark കളുടെ ഇടയിൽ കാണപ്പെടുന്ന ഒറ്റയ്ക്കിരിക്കുന്ന കീബോർഡ് ക്യാരക്ടർറിനെ character constant എന്ന് പറയാം. ഉദാ: 'a', 'B' മുതലായവ |
| **Floating point constant** - Floating point constants are also know as Real constant. Floating point constants are represented ordinary decimal form | ദശാംശമുൾപ്പെടെയുള്ള സംഖ്യകളാണ് Floating point constant. സാധാരണദശാംശരൂപത്തിലോ Mantissa, Exponent എന്നിവയോടെ ശാസ്ത്രീയരൂപത്തിലോ |

HSSLiVE.IN

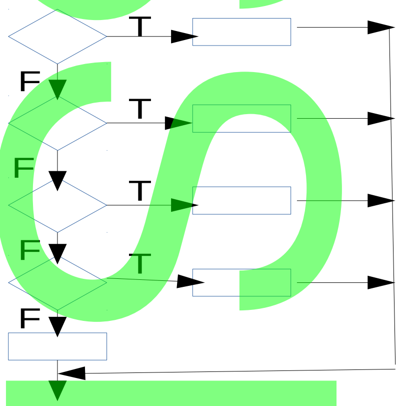
| | |
|---|---|
| and in scientific form using Mantissa and Exponent Parts. | ഇതിനെ കാണിക്കാം. |
| **String constant** - A string constant is enclosed within double quotation mark. Every string in C++ is terminated by a Null character(\0).For example "abc","Object" etc. | Double quotation mark നുള്ളിൽ കാണുന്ന ക്യാരക്ടറുകളുടെ കൂട്ടമാണ് string constant. Null character('\0') ലാണ് ഇത് അവസാനിക്കുന്നത്. |
| **Escape sequence** - C++ allows certain characters that cannot be directly typed or entered from keyboard (Non-graphic characters) such as new line, back space ,.etc. These characters begins with backslash(\) and are called escape sequence. | കീബോർഡുപയോഗിച്ച് നേരിട്ട് ടൈപ്പ് ചെയ്യാൻ സാധിക്കാത്ത Non-graphic character കളെ സൂചിപ്പിക്കുന്നതാണ് escape sequence. Backslash(\) ഉപയോഗിച്ച് ക്യാരക്ടർ കോൺസ്റ്റന്റ് അല്ലെങ്കിൽ string constant ൽ ഇത്തരം ക്യാരക്ടറുകളെ സൂചിപ്പിക്കാം. |
| **Operator** | |
| An operator is a symbol used to perform a specific task. The data on which an operator act upon is called operand. Based on the number of operands operators are classified into three, Unary operator, Binary operator and Ternary operator. | പ്രത്യേക ജോലി ചെയ്യുന്നതിനെ സൂചിപ്പിക്കുന്ന ചിഹ്നമാണ് operator. Operator കളുടെ കാണപ്പെടുന്ന സംഖ്യകളുടെയും വേരിയബിളുകളുടെയും എണ്ണമനുസരിച്ച് മൂന്നായി തരം തിരിക്കാം. |
| A unary operator acts on a single operand. A binary operator acts on two operands. A ternary operator(? :) acts on three operands. | ഒരു operand മാത്രമാണെങ്കിൽ unary operator, രണ്ട് operand കൾ ഉണ്ടെങ്കിൽ binary operator , മൂന്ന് operand കൾ ഉണ്ടെങ്കിൽ ternary operator(? :) |
| Based on the nature of operation operators are classified into three | ഉപയോഗമനുസരിച്ച് ഓപ്പറേറ്ററുകളെ മൂന്നായി തരം തിരിക്കാം. |
| Arithmetic operators - Arithmetic operator are used for addition, substraction, multiplication and division(+,-,*,/). C++ provides a special operator modulus operator(%) for getting remainder of division operation. For example 10 % 2 returns 0. | Arithmetic operators – ഗണിതസംബന്ധമായ സങ്കലനം, വ്യവലനം, ഗുണനം, ഹരണം എന്നിവ ചെയ്യുന്നതാണ് Arithmetic operators(+,-,*,/). ഹരണത്തിനുശേഷമുള്ള ശിഷ്ടം കണ്ടുപിടിക്കാൻ modulus operator (%) C++ നൽകുന്നു. |
| Relational operators - Relational operators are used to compare values. They produces only 'True' of 'False' values. The following are the relational operators in C++. (<, <=, >, >=, ==, !=) | Relational operators – രണ്ട് സംഖ്യകളെ താരതമ്യം ചെയ്യാനുപയോഗിക്കുന്നതാണ് Relational operators (<, <=, >, >=, ==, !=). ശരി, തെറ്റ് എന്നീ രണ്ട് ഫലങ്ങൾ മാത്രമാണിത് തരുന്നത്. |
| Logical operator - Logical operators allows us to combine two or more conditions. They are often referred to as Boolean operators. The logical operators are &&(logical AND),\|\|(logical OR),and ! (logical NOT). | Logical operator – രണ്ടോ അതിൽക്കൂടുതലോ condition കളെ യോജിപ്പിക്കാനുപയോഗിക്കുന്നതാണ് Logical operator. &&(logical AND),\|\|(logical OR),and ! (logical NOT) എന്നിവയാണ് Logical operator |
| **Input-Output operators** | |
| The Input-Output operators are used to take input and display output. The operator(>>) is used for taking input,and the operator(<<)is used to display the output.The >> operator is called extraction or get from operator.The operator << is called insertion or put to operator.It is a binary operator. | ഇൻപുട്ട് സ്വീകരിക്കാനുള്ള operator(>>) extraction അല്ലെങ്കിൽ get from എന്നും ഔട്ട്പുട്ട് കാണിക്കാനുള്ള operator(<<) insertion അല്ലെങ്കിൽ put to എന്നും അറിയപ്പെടുന്നു. ഇത് രണ്ടും ബൈനറി ഓപ്പറേറ്റർകൾ ആണ്. |
| **Input and output statements** | |
| The input operator >> is used with standard input stream cin and the output operator << is used with standard output stream, cout. The input operator takes the value through cin and stores in a variable.<br>cin >> number; [Input value to variable number]<br>cout << number+2; [output 2 added to value in | ഇൻപുട്ട് ഓപ്പറേറ്റർ >> ഇൻപുട്ടിനുവേണ്ടിയുള്ള cin സ്റ്റേറ്റ്മെന്റിനോടൊപ്പവും ഔട്ട്പുട്ട് ഓപ്പറേറ്റർ << ഔട്ട്പുട്ടിനുവേണ്ടിയുള്ള cout സ്റ്റേറ്റ്മെന്റിനോടൊപ്പവും ഉപയോഗിക്കുന്നു. |

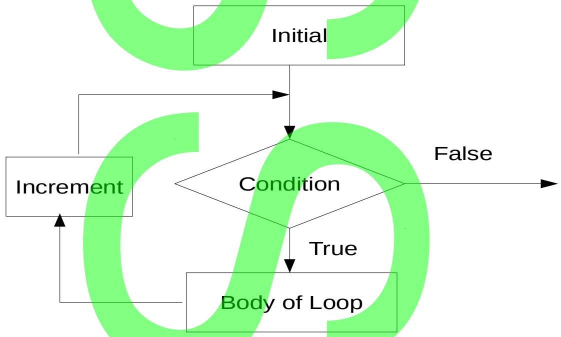
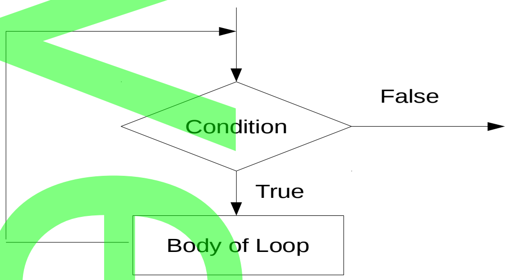
| | |
|---|---|
| variable number] | cin >> number; [number എന്ന വേരിയബിളിന് ഇൻപുട്ട് ]<br>cout << number+2; [number എന്ന വേരിയബിളിലെ മൂല്യത്തിനോട് 2 കൂട്ടിയത് ഔട്ട്പുട്ട് ] |
| **Assignment operator (=)** | |
| The assignment operator is used to assign a value to a variable.<br>For example a=10 Assigns the value 10 to a<br>*The symbol = assigns a value, where as the symbol == compare two values.* | വേരിയബിളിന് മൂല്യം നൽകുവാൻ ഉപയോഗിക്കുന്നതാണ് അസൈൻമെന്റ് ഓപ്പറേറ്റർ.<br>a=10 a എന്ന വേരിയബിളിൽ 10 എന്ന സംഖ്യ സൂക്ഷിക്കുന്നു.<br>*= മൂല്യം സൂക്ഷിക്കാനും == മൂല്യങ്ങൾ താരതമ്യപ്പെടുത്താനും ഉപയോഗിക്കുന്നു.* |
| **Expressions** | |
| An expression consists of operator and operand. Operands can be constants or variables. There are different types of expressions in C++:<br>Integral expression: All operands in the expressions are integers. An integral expression yields an integral result.<br>Floating point (decimal) expression: All operands in the expression are floating points(decimals). A floating point expression yields a floating point result.<br>Relational expression and Logical expression: The relational expression consists of relational operators. They are also called conditions or test expression. Relational expressions can be combined using logical operators. | ഓപ്പറേറ്ററും ഓപ്പറാന്റും കൂടിയതാണ് expression. ഓപ്പറാന്റ് എന്നത് വേരിയബിളോ കോൺസ്റ്റന്റോ ആകാം. C++ ലെ വിവിധതരം expression കൾ താഴെ നൽകുന്നു.<br>Integral expression: പൂർണ്ണസംഖ്യകൾ മാത്രമുപയോഗിച്ച് പൂർണ്ണസംഖ്യ ഫലം തരുന്നു.<br>Floating point (decimal) expression: ദശാംശമുള്ള സംഖ്യകളുപയോഗിച്ച് ദശാംശത്തിലുള്ള ഫലം നൽകുകയും ചെയ്യുന്നു.<br>Relational expression, Logical expression: റിലേഷനൽ ഓപ്പറേറ്ററുകൾ ഉപയോഗിച്ച് രൂപപ്പെടുത്തുന്നതാണ് Relational expression. ഇവയെ condition അല്ലെങ്കിൽ test expression എന്നും വിളിക്കാം. ഇവയെ ലോജിക്കൽ ഓപ്പറേറ്ററുകൾ ഉപയോഗിച്ച് സംയോജിപ്പിക്കാം |
| C++ short hands are used to simplify the coding of assignment statements.<br>For example x=x+20; can be written as X+=20; | C++ short hand കൾ ഉപയോഗിച്ച് അസൈൻമെന്റുകളെ ലഘുവായി എഴുതാം.<br>ഉദാ: x=x+20; എന്നത് X+=20; ആയി എഴുതാം. |
| **Increment and Decrement Operators** | |
| In C++ there are two special operators increment (++) and decrement (--) which increases the value by one and the decrement operator decreases the value by one. | C++ലെ Increment (++), decrement (--) എന്ന രണ്ട് പ്രത്യേകതരം ഓപ്പറേറ്ററുകൾ വേരിയബിളിന്റെ മൂല്യം ഒന്ന് വീതം കൂട്ടുകയോ കുറയ്ക്കുകയോ ചെയ്യുന്നു. |
| **Control Statements** | |
| Control statements are used to alter the normal sequence of execution of a program. They are classified into two types, | പ്രോഗാരാമിന്റെ സ്വാഭാവികമായ ഒഴുക്കിനെ മാറ്റം വരുത്തുന്നതാണ് കൺട്രോൾ സ്റ്റേറ്റ്മെന്റ്. ഇത് രണ്ടായി തരം തിരിക്കാം. |
| **Decision statements** | |
| The decision statements are based on a condition. They are also known as branching statements. The flow of control depends on the result of a particular condition. C++ supports three types of decision statements,<br>a) if statement<br>b) if – else statement<br>c) Switch statement | കണ്ടീഷൻ (സാഹചര്യം) അനുസരിച്ച് പ്രവർത്തിക്കുന്നതാണ് decision statement അല്ലെങ്കിൽ ബ്രാഞ്ചിംഗ്. കണ്ടീഷന്റെ ഫലം അനുസരിച്ച് പ്രോഗ്രാമിന്റെ പ്രവർത്തനപഥം മാറിക്കൊണ്ടിരിക്കുന്നു. C++ൽ മൂന്ന് തരം decision statement ആണുള്ളത്. |
| **The if statement** | |
| The if statement is a single path statement, which executes statement only if the condition is true. The | കണ്ടീഷൻ True എന്ന ഫലം തരികയാണെങ്കിൽ മാത്രം പ്രവർത്തിക്കുന്നതാണ് if സ്റ്റേറ്റ്മെന്റ്. പദഘടന താഴെ |

HSSLiVE.iN

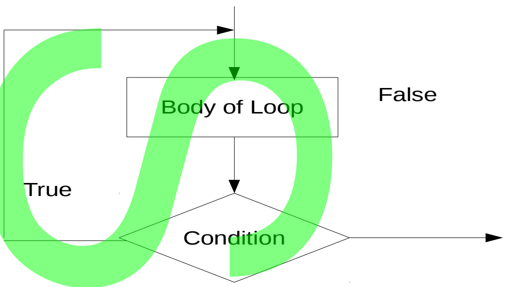
<table>
<tr><td>syntax is,<br>if(condition or expression)<br>{<br>statement1;<br>}<br>If the condition is true then the statement1 will be executed.</td><td>നൽകിയിരിക്കുന്നു.<br>if(condition or expression)<br>{<br>statement1;<br>}<br>condition true ആണെങ്കിൽ മാത്രം statement1 പ്രവർത്തിക്കും.</td></tr>
<tr><td colspan="2"><br></td></tr>
<tr><td colspan="2">if – else statements</td></tr>
<tr><td>The if... else statement is used to test a logical condition and choose one of the alternatives based on the result of logical condition. The syntax of if – else statement is<br>If (condition or expression)<br>{<br>statement 1;<br>}<br>else<br>{<br>statement 2;<br>}</td><td>കണ്ടീഷന്റെ ഫലം True അല്ലെങ്കിൽ False ആകുന്നത് അനുസരിച്ച് പ്രവർത്തികേണ്ട പ്രോഗ്രാം ഭാഗം തെരഞ്ഞെടുക്കാൻ സാഘിക്കുന്നതാണ് if... else സ്റ്റേറ്റ്മെന്റ്. പദഘടന താഴെ നൽകിയിരിക്കുന്നു.<br>If (condition or expression)<br>{<br>statement 1;<br>}<br>else<br>{<br>statement 2;<br>}</td></tr>
<tr><td colspan="2"><br></td></tr>
<tr><td colspan="2">The else-if ladder</td></tr>
<tr><td>The else-if ladder helps to select one out of many alternative block of statements for execution depending on mutually exclusive conditions. The syntax of the else... if ladder is,<br>If (test- expression )<br>{<br>Statement 1;<br>}<br>else if (test-expression)</td><td>ഒന്നിലധികം കണ്ടീഷനുകൾ ഒരേസമയം കൈകാര്യം ചെയ്യുമ്പോൾ ഉപയോഗിക്കാവുന്ന മാർഗമാണ് else-if ladder. ഇതിന്റെ പദഘടന നൽകുന്നു.<br>If (test- expression )<br>{<br>Statement 1;<br>}<br>else if (test-expression)</td></tr>
</table>

HSSLiVE.IN

| | |
|---|---|
| {<br>Statement 2;<br>}<br>else if (test-expression)<br>{<br>Statement 3;<br>}<br>else<br>{<br>Statements;<br>} | {<br>Statement 2;<br>}<br>else if (test-expression)<br>{<br>Statement 3;<br>}<br>else<br>{<br>Statements;<br>} |
| <br> | |
| **Switch statement** | |
| The switch enables to select one among several alternatives. It is a multi- path statement. The syntax for switch statement is<br>switch (expression)<br>{<br>case value1 :Statement 1;break;<br>case value2 :Statement 2;break;<br>case value3 :Statement 3;break;<br>..<br>..<br>default : Statement n;<br>}<br>The expression is evaluated and statements are executed according to the value. If no match is found, then default statement is executed.<br>Note :Switch can be used instead of multiple if else if statement. The expression in a switch statement can be either integer or a character. | ഒന്നിലധികം സാധ്യതകളെ കൈകാര്യം ചെയ്യാൻ സഹായിക്കുന്നതാണ് switch. ഇത് ഒരു multi- path സ്റ്റേറ്റ്മെന്റ് ആണ്. ഇസിനുള്ള പദഘടന താഴെ നൽകുന്നു.<br>switch (expression)<br>{<br>case value1 :Statement 1;break;<br>case value2 :Statement 2;break;<br>case value3 :Statement 3;break;<br>..<br>..<br>default : Statement n;<br>}<br>expression ന്റെ മൂല്യത്തിനനുസരിച്ച് ഓരോ case ലുമുള്ള സ്റ്റേറ്റ്മെന്റുകൾ പ്രവർത്തിക്കുന്നു. ഒന്നും ഒത്തുവരുന്നില്ലെങ്കിൽ default പ്രവർത്തിക്കുന്നു. multiple if else if നു പകരം ഉപയോഗിക്കാവുന്ന switch ൽ expression എന്നത് int അല്ലെങ്കിൽ char മാത്രമാണ് അനുവദിക്കപ്പെട്ടിട്ടുള്ളത്. |
| **Loops or Iterations** | |
| Loops or iterations are statements that execute repeatedly while some condition is met. The repetition stops only when the condition becomes false. | ലൂപ്പ് അല്ലെങ്കിൽ iteration എന്നത് ഏതെങ്കിലും condition പ്രകാരം ആവർത്തിച്ച് പ്രവർത്തിക്കുന്ന സ്റ്റേറ്റ്മെന്റുകൾ ആണ്. Condition False ആകുമ്പോൾ ലൂപ്പ് അവസാനിപ്പിക്കുന്നു. |
| **for statement** | |
| The for statement is the most commonly used statement in C++. It is an entry controlled loop. The | C++ ൽ ഏറ്റവും കൂടുതലായി ഉപയോഗിക്കുന്ന loop |

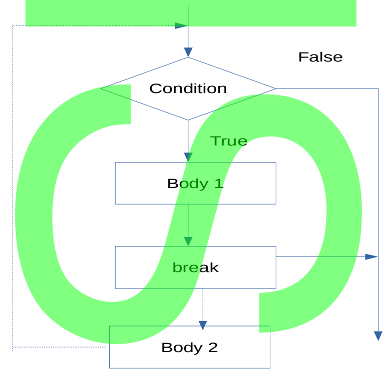
| | |
|---|---|
| syntax of for ... loop is<br>for(initialisation ;test expression ;updation)<br>{<br>Body of loop;<br>} | സ്റ്റേറ്റ്മെന്റാണ് for. ഇതിന്റെ പദഘടന താഴെ നൽകിയിരിക്കുന്നു.<br>for(initialisation ;test expression ;updation)<br>{<br>Body of loop;<br>} |
|  | |
| **While Statement** | |
| The while ... loop is an entry controlled loop.The condition is checked first and if it is true the loop will be executed.The synatx is<br>While (expression)<br>{<br>Body of loop;<br>}<br>Example:Program to print first 10 even numbers<br>#include<iostream.h> | Condition പരിശോധിച്ച് True ലഭിക്കുകയാണെങ്കിൽ ലൂപ്പിലേയ്ക്ക് പ്രവേശിക്കുകയും False ആകുമ്പോൾ പുറത്തേയ്ക്ക് പോകുകയും ചെയ്യുന്നതാണ് while ലൂപ്പിന്റെ ഘടന.<br>While (expression)<br>{<br>Body of loop;<br>} |
|  | |
| **do ... while loop** | |
| The do while loop executes statement before the expression is tested. The syntax is<br>do<br>{<br>Statements ;<br>}<br>While (expression);<br>Note: The do...while loop is an exit controlled loop in C++. | ലൂപ്പിലേയ്ക്ക് ആദ്യം തന്നെ പ്രവേശിക്കുകയും Condition പരിശോധിച്ച് True ലഭിക്കുകയാണെങ്കിൽ ലൂപ്പിനുള്ളിൽ നിലനിൽക്കുകയും False ആകുമ്പോൾ പുറത്തേയ്ക്ക് പോകുകയും ചെയ്യുന്നതാണ് do-while ലൂപ്പിന്റെ ഘടന.<br>do<br>{<br>Statements ;<br>}<br>While (expression); |

HSSLive.in

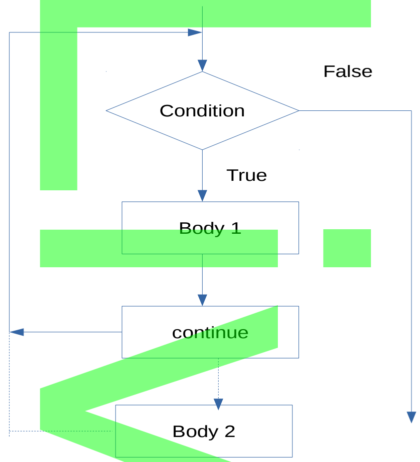



| | |
|---|---|
| Difference between do-while and while Loop | while, do-while തമ്മിലുള്ള വ്യത്യാസം |
| In the while loop the condition is tested and the body gets executed. Whereas in do..while loop the condition is checked at the end of the loop.The do...while loop executes atleast one time even if the condition is false. | while ലൂപ്പിൽ condition പരിശോധിച്ചതിനുശേഷം ലൂപ്പിൽ പ്രവേശിക്കുമ്പോൾ do-while ലൂപ്പിൽ പ്രവേശിക്കുകയും ഒരു തവണ പ്രവർത്തിക്കുകയും ചെയ്യുകഴിഞ്ഞ് മാത്രമാണ് condition പരിശോധിക്കുന്നത്. |
| **Conditional Operator (?:)** | |
| The conditional operator is a ternary operator (ie,it takes three operands).The general form is Condition? true-part:false-part;<br>The condition is a boolean expression. If it is true the result is the true- part,else the result is the false-part .<br>Note:The conditional Operator can be used as an alternative to if... else. | മൂന്ന് ഓപ്പറാന്റുകളുള്ള (ternary operator) ഏക ഓപ്പറേറ്ററാണ് Conditional Operator (?:). ഇതുപയോഗിക്കുന്നതിനുള്ള പൊതുവായ രൂപം Condition?true-part:false-part; if... else നു പകരം ഉപയോഗിക്കാൻ സാധിക്കുന്നതാണ് Conditional Operator. |
| **Nested Loop** | |
| A loop placed inside another loop is called a nested loop and the process of placing a loop inside another loop is called nesting. | ഒരു ലൂപ്പിനുള്ളിൽ മറ്റൊരു ലൂപ്പ് വരുന്നതിനെയാണ് nested loop എന്ന് വിളിക്കുന്നത്. |
| **Jump statements** | |
| Jump statements are used to jump unconditionally to a different statement. It is used to alter the flow of control unconditionally. There are four types of jump statements in C++, break, continue, goto and return. | യാതൊരു കണ്ടീഷനും കൂടാതെ പ്രോഗ്രാമിന്റെ ഒരു ഭാഗത്തുനിന്നും മറ്റൊരു ഭാഗത്തിലേയ്ക്ക് മാറാനുപയോഗിക്കുന്നതാണ് Jump സ്റ്റേറ്റ്മെന്റുകൾ. C++ൽ ഇത്തരം നാല് സ്റ്റേറ്റ്മെന്റുകളാണുള്ളത്. break, continue, goto and return. |
| break - The break statement is used to terminate a loop or switch statement abnormally. | Break – സ്റ്റേറ്റ്മെന്റ് ലൂപ്പിനെയും switch നെയും അസാധാരണമായി അവസാനിപ്പിക്കാൻ ഉപയോഗിക്കുന്നു. |

HSSLive.in




| | |
|---|---|
| continue - A continue statement is used to continue to the beginning of a loop. When a continue statement is executed in a loop it skips the remaining statements in the loop and proceeds with the next iteration of the loop. In a while and do.. while loop, the condition is evaluated immediately after the continue statement. In a for loop ,the update statement is executed after the continue statement. | continue - ലൂപ്പുകളുടെ സാധാരണ ഒഴുക്കിനെ മറികടന്ന് അടുത്ത iteration ലേയ്ക്ക് വഴിതിരിച്ചുവിടാനുപയോഗിക്കുന്നതാണ് continue. ഈ സ്റ്റേറ്റ്മെന്റ് കണ്ടാൽ തുടർന്നുള്ള സ്റ്റേറ്റ്മെന്റ്കളെ ഒഴിവാക്കുകയും While, do.. while കളിൽ കണ്ടീഷൻ നോക്കുകയും for ൽ update statement പ്രവർത്തിക്കുകയും ചെയ്യുന്നു. |




| | |
|---|---|
| goto - A goto statement is used for unconditional jump. It transfers the control from one part of the program to another. The syntax is,<br>goto label;<br>A label is an identifier followed by a colon. | goto ഉപയോഗിച്ച് പ്രോഗ്രാമിന്റെ ഏതെങ്കിലും ഭാഗത്തേയ്ക്ക് യാതൊരു കണ്ടീഷനും കൂടാതെ മാറാൻ കഴിയും. ഇത് നല്ലൊരു പ്രോഗ്രാമിംഗ് style അല്ല.<br>goto label; എന്നതാണിതിന്റെ പദഘടന.<br>Label എന്നത് ഒരു ഐഡന്റിഫയർ ആണ്. |
| return – return statement changes the control from a function to its parent program. (details in chapter Functions) | return – ഒരു ഫങ്ഷനിൽ നിന്നും പേരന്റ് പ്രോഗ്രാമിലേയ്ക്ക് തിരിച്ചു പോകാൻ ഉപയോഗിക്കുന്നതാണ് return. (വിശദാംശങ്ങൾ Functions എന്ന അധ്യായത്തിൽ) |

Hsslive.in