**Reserved Words**

|  |  |  |  |
| --- | --- | --- | --- |
| C Language | C++ Language | Proposed Language | Description |
| main | main | !START | The function that is called by the operating system when the end-user runs the program. |
| *Data Types* | | | |
| int | int | number | A data type that holds the value of whole numbers. |
| float | float | decimal | A data type that holds the value of decimal numbers. |
| string | string | string | A data type that holds the value of characters. |
| bool | bool | boolean | A data type that holds the value of ‘true’ and ‘false’. |
| void | void | null | A special type that represents the absence of value. |
| *Data Structures* | | | |
| struct | struct | file | A structure type that allows the end-user to combine data items of different kinds. |
| *I/O Statements* | | | |
| printf | cout | write | A function that can print the data or user defined message on console or monitor. |
| scanf | cin | read | A function that can read the input value from the keyword. |
| *Conditional Statements* | | | |
| if | if | if | Executes a statement or block of statements if a specified condition is true. |
| else | else | else | Executes a statement or block of statements if a specified condition is false. |
| else if | else if | elseif | Performed after an if statement that if true, performs a function. |
| switch | switch | switch | Allows a variable to be tested for equality against a list of values. |
| case | case | choice | Value that is tested in a switch statement. |
| default | default | default | Performs a task when none of the choices is true. |
| *Looping Statements* | | | |
| for | for | for | Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable. |
| while | while | while | Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body. |
| do | do | do | Like a while statement, except that it tests the condition at the end of the loop body. |
| *Loop Control Statements* | | | |
| break | break | stop | Terminates the switch statement. |
| continue | continue | jump | Forces the next iteration of the loop to take place, skipping any code in between. |
| *Other Statements* | | | |
| true | true | true | Boolean value. |
| false | false | false | Boolean value. |
| return | return | return | Stops execution and returns to the calling function. |
| - | - | start | Marks the start of a function. |
| - | - | end | Marks the end of a function. |
| - | - | END! | Marks the end of the main function (!START). |
| && | && | and | Yields true if both operands are true, otherwise false. |
| || | || | or | Yields true if either of its operands is true, thus being false only when both operands are false. |
| - | - | fixed | Used to declare constant variables. |
| \n | \n | nl | Used to have a new line in the program. |
| \t | \t | tab | Used to have a tab in the program. |

**Reserved Symbols**

***Arithmetic Operator***

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| + | Adds two operands. | 10 + 5 will give 15 |
| - | Subtracts second operand from the first. | 10 – 5 will give 5 |
| \* | Multiplies both operands. | 10 \* 5 will give 50 |
| / | Divides numerator by de-numerator. | 10 / 5 will give 2 |
| % | Yields the remainder after integer division. | 10 % 5 will give 0 |
| ++ | Increases integer value by one. | 10++ will give 11 |
| -- | Decreases integer value by one. | 10-- will give 9 |

***Assignment Operator***

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| = | Simple assignment operator, assigns values from right side operands to left side operand. | A = 10 + 5 will assign value of 10 + 5 into A |
| += | Add *and* assignment operator, it adds right operand to the left operand and assign the result to left operand. | A += 10 is equivalent to A = A + 10 |
| -= | Subtract *and* assignment operator, it subtracts right operand from the left operand and assign the result to left operand. | A -= A is equivalent to A = A - 10 |
| \*= | Multiply *and* assignment operator, it multiplies right operand with the left operand and assign the result to left operand. | A \*= A is equivalent to A = A \* 10 |
| /= | Divide *and* assignment operator, it divides left operand with the right operand and assign the result to left operand. | A /= 10 is equivalent to A = A / 10 |

***Relational Operator***

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| == | Checks if the values of two operands are equal or not, if yes then condition becomes true. | (10 == 5) is false |
| != | Checks if the values of two operands are equal or not, if values are not equal then condition becomes true. | (10 != 5) is true |
| > | Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true. | (10 > 5) is true |
| < | Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true. | (10 < 5) is not true |
| >= | Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true. | (10 > 5) is true |
| <= | Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true. | (10 < 5) is not true |

***Logical Operator***

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| ! | Used to reverse the current state of the operand. | !(10 == 10 && 5 == 10) is true |

***Unary Operator***

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| ++ | Increases integer value by one. | 10++ will give 11 |
| -- | Decreases integer value by one. | 10-- will give 9 |

***Others***

|  |  |
| --- | --- |
| Operator | Description |
| . | Terminator |
| , | Separator |
| “” | Used in defining the value of a string |
| ( ) | Used in different operations |
| [ ] | Used in declaring arrays |
| @ | Used to access an element of a file |