```
/*
 * C++ Programming Notes
 * Ravi Kumar Reddy K
 * github.com/ravikumark815
*/
```

Preset:

- Invented by Bjarne Stroustrup in 1979
- Middle Level Language
- Versions: C++ 14, C++11, C++99

Hello World:

```
#include <iostream>
using namespace std;

int imGlobal = 0;
const double PI = 3.141;

int main(int argc, char**argv) {
  cout << "Hello World\n";
  return 0;
}
```

- Namespaces
- main: Start executing from here
- Cout allows us to output information to console
- “<<” Stream insertion operator: Takes string on the right to cout stream
- “endl” Issue newline and force write to console
- argc: No of arguments passed to main
- argv: Array of pointers to strings in the arg vector
- int: Return an integer when done executing
- imGlobal: Global variable and accessible everywhere else.
- const double PI: Global variable whose value cannot be changed anywhere else

Comments:

```
/*
Multi
Line
Comment
*/
// Single Line Comment
```

Common Header files:

- #include <cstdlib> // Sorting, Searching, import c libraries, rand, memmgmt, and other general-purpose functions
- #include <iostream> // Read and Write data
- #include <string> // Work with strings
- #include <limits> // Min and max values
- #include <vector> // Work with vectors
- #include <sstream> // Work with string streams
- #include <numeric> // Work with sequences of values
- #include <ctime> // Work with time
- #include <cmath> //Common math functions

Data Types:

Type	Typical Bit Width	Typical Range
char	1byte	-127 to 127 or 0 to 255
unsigned char	1byte	0 to 255
signed char	1byte	-127 to 127
int	4bytes	-2147483648 to 2147483647
unsigned int	4bytes	0 to 4294967295
signed int	4bytes	-2147483648 to 2147483647
short int	2bytes	-32768 to 32767
unsigned short int	2bytes	0 to 65,535
signed short int	2bytes	-32768 to 32767
long int	8bytes	-9223372036854775808 to 9223372036854775807
signed long int	8bytes	same as long int
unsigned long int	8bytes	0 to 18446744073709551615
long long int	8bytes	-(2^63) to (2^63)-1
unsigned long long int	8bytes	0 to 18,446,744,073,709,551,615
float	4bytes	
double	8bytes	
long double	12bytes	
wchar_t	2 or 4 bytes	1 wide character

Data Type	Initializer
int	0
char	'\0'
float	0
double	0
pointer	NULL

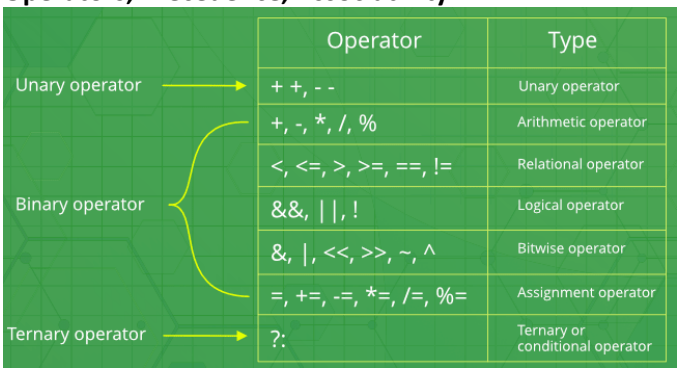
Variables:

- Definition: type variable_list = value;
- Ex: int i,j,k=10; char c,ch;
-

Input and Output:

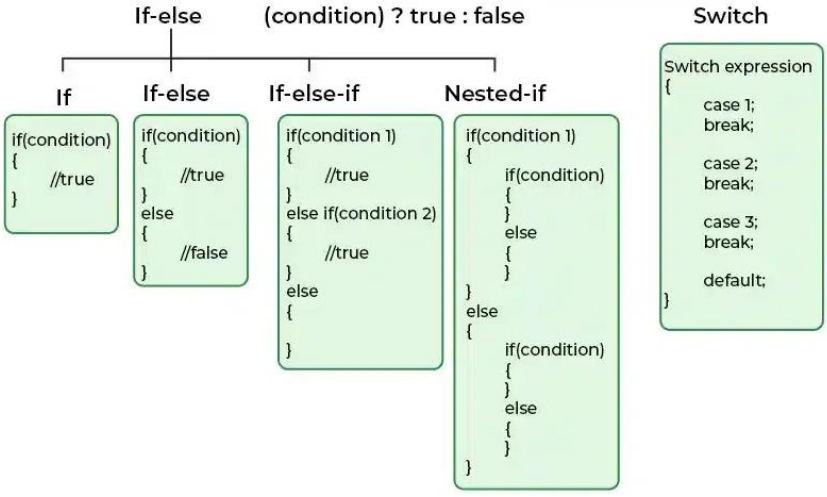
- cout << “Min int” << numeric_limits<int>::min();
- cout << “Max short int” << numeric_limits<short int>::max();
- printf(“Sum = %.7f\n”), (1.1111111+1.1111111)); // To print formatted output of float upto 7 decimal places
- cout << “int Byte:” << sizeof(int) << endl;
- printf(“%c %d %5d %.3f %s\n”, ‘A’, 10, 5, 3.1234, “Hi); // O/p: A 10   5 3.123 Hi //Right justify
- cin >> num_str; //to take in input for num1
- int num1 = stoi(num_str) //To convert num1 from string to int;
- bool res=true; cout.setf(ios::boolalpha); cout << res << endl; // To print booleans

Operators, Precedence, Associativity:

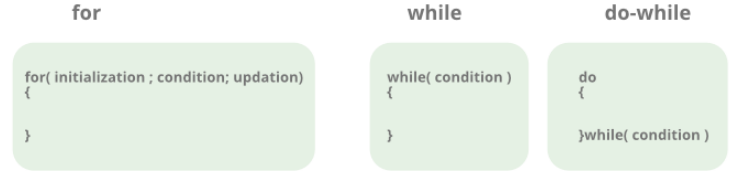


Precedence	Operator	Description	Associativity
1	: :	Scope resolution	Left to right
2	a++ a— type() type{} a() a[] . -> Member access	Postfix increment and decrement Function cast Function call Subscript	Left to right
3	++a - -a +a -a ! ~ (type) *a &a sizeof co_wait new new[ ] delete delete[ ]	Prefix increment and decrement Unary plus and minus Logical and bitwise NOT C-Style cast Dereference Address of Size-of Await expression Dynamic memory allocation Dynamic memory deallocation	Right to left
4	.* ->*	Pointer to member	Left to right
5	a*b a/b a%b	Multiplication, division, remainder	
6	a+b a-b	Addition , subraction	
7	<< >>	Bitwise left and right shift operators	
8	<= >	Three way comparision	
9	< <= > >=	Relational operators	
10	== !=	Equality and not equality check operators	
11	&	Bitwise AND	
12	^	Bitwise XOR	
13	|	Bitwise OR	
14	&&	Logical AND	
15	||	Logical OR	
16	a ? b:c throw co_yield = += -= *= /= %= <<= >>= &= ^= |=	Ternary conditional operator throw operator yield-expression Direct assignment Compound assignment by sum, difference Compound assignment by product,quotient,remainder Compound assignment by bitwise left and right shift Compound assignment bY bitwise AND, XOR, OR	Right to left
17	,	comma	Left to right

Conditional Statements:



Loops:




```
while (i <= 20){
    // If a value is even don't print it
    if((i % 2) == 0){
```

```
        i += 1;

        // Continue skips the rest of the code
        // and jumps back to the beginning
        // of the loop
        continue;
      }

      // Break stops execution of the loop and jumps
      // to the line after the loops closing }
      if(i == 15) break;

      cout << i << "\n";

      // Increment i so the loop eventually ends
      i += 1;
   }

// An abbreviated for loop
   int arr3[] = {1,2,3};
   for(auto x: arr3) cout << x << endl;

// Do while loops are guaranteed to execute at
   // least once
   // We'll create a secret number guessing game

   // We need to seed the random number generator
   // time() returns the number of seconds
   // since 1, 1, 1970
   // Include <ctime>
   srand(time(NULL));

   // Generate a random number up to 10
   int secretNum = rand() % 11;
   int guess = 0;
   do{
      cout << "Guess the Number : ";
      cin >> guess;
      if(guess > secretNum) cout << "To Big\n";
      if(guess < secretNum) cout << "To Small\n";
   } while(secretNum != guess);

   cout << "You guessed it" << endl;
```

Arrays:

```
void main(int argc, char**argv) {
            int array1 [10] = {1};       // Size
            int array2 [] = {1,2,3};     // Size for this would automatically be 3
            int array3 [5] = {8,9};      //
            cout << “First val: ” << array1[0] << endl;
            array1[0] = 7;
            int array4[2][3][3] = { {{1,2}, {3,4}}, {{5,6}, {7,8}} };    // Multidimensional arrays
            cout << array4[0][1][1] <<endl //prints 4
```

```
        return 0;
}
```

- Size once defined cannot be changed.

Vectors:

```
// ---------- VECTORS ----------
   // Vectors are used when you don't know how big the array
   // should be
   vector<int> vNums(2);

   // Add values
   vNums[0] = 1;
   vNums[1] = 2;

   // Add another to the end
   vNums.push_back(3);

   // Get vector size
   cout << "Vector Size : " << vNums.size() << endl;
```

String Streams:

```
// A stringstream object receives strings separated
   // by a space and then spits them out 1 by 1
   vector<string> words;
   stringstream ss("Some Random Words");
   string word;

   // A while loop will execute as long as there are
   // more words
   while(getline(ss, word, ' ')){
      words.push_back(word);
   }
// Cycle through each index in the vector using
   // a for loop
   for(int i = 0; i < words.size(); ++i){
      cout << words[i] << endl;
   }
```

Strings:

```
// A C++ string is a series of characters that
   // can be changed
   string str1 = "I'm a string";

   // Get the 1st character
   cout << "1st : " << str1[0] << endl;

   // Get the last character
   cout << "Last : " << str1.back() << endl;

   // Get the string length
   cout << "Length : " << str1.length() << endl;
```

```
   // Copy a string to another
   string str2 = str1;

   // Copy a string after the 1st 4 characters
   string str3(str2, 4);

   // Combine strings
   string str4 = str1 + " and your not";

   // Append to the end of a string
   str4.append("!");

   // Erase characters from a string from 1 index
   // to another
   str4.erase(12, str4.length() - 1);
   cout << "New String : " << str4 << endl;

   // find() returns index where pattern is found
   // or npos (End of String)
   if(str4.find("string") != string::npos)
      cout << "String Index : " <<
            str4.find("string") << endl;
// O/p: String Index: 6

   // substr(x, y) returns a substring starting at
   // index x with a length of y
   cout << "Substring : " <<
         str4.substr(6,6) << endl;
//O/p: Substring: string

   // Convert int to string
   string strNum = to_string(1+2);
   cout << "I'm a String : " << strNum << "\n";
//O/p: I'm a String: 3
```

Character functions

```
   char letterZ = 'z';
   char num5 = '5';
   char aSpace = ' ';
   cout << "Is z a letter or number " <<
         isalnum(letterZ) << endl;
   cout << "Is z a letter " <<
         isalpha(letterZ) << endl;
   cout << "Is 3 a number " <<
         isdigit(num5) << endl;
   cout << "Is space a space " <<
         isspace(aSpace) << endl;
```

Math Functions:

```
   cout << "abs(-10) = " << abs(-10) << endl;
   cout << "max(5, 4) = " << max(5, 4) << endl;
   cout << "min(5, 4) = " << min(5, 4) << endl;
   cout << "fmax(5.3, 4.3) = " << fmax(5.3, 4.3) << endl;
```

```
    cout << "fmin(5.3, 4.3) = " << fmin(5.3, 4.3) << endl;
    cout << "ceil(10.45) = " << ceil(10.45) << endl;
    cout << "floor(10.45) = " << floor(10.45) << endl;
    cout << "round(10.45) = " << round(10.45) << endl;
    cout << "pow(2,3) = " << pow(2,3) << endl;
    cout << "sqrt(100) = " << sqrt(100) << endl;
    cout << "cbrt(1000) = " << cbrt(1000) << endl;
// e ^ x
    cout << "exp(1) = " << exp(1) << endl;

    // 2 ^ x
    cout << "exp2(1) = " << exp2(1) << endl;

    // e * e * e ~= 20 so log(20.079) ~= 3
    cout << "log(20.079) = " << log(20.079) << endl;

    // 2 * 2 * 2 = 8
    cout << "log2(8) = " << log2(8) << endl;

    // Hypotenuse : SQRT(A^2 + B^2)
    cout << "hypot(2,3) = " << hypot(2,3) << endl;

// Also sin, cos, tan, asin, acos, atan, atan2,
    // sinh, cosh, tanh, asinh, acosh, atanh
```

Functions: