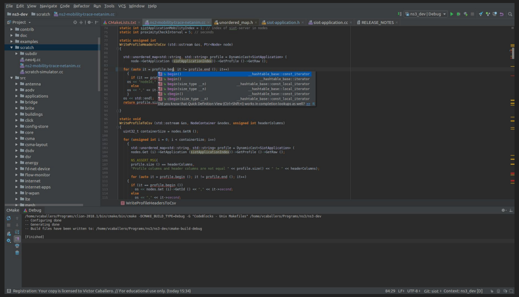
Introduction to C++

Variables, conditions, data structures

Akis Zachariadis - Games Dept. Coordinator

C++ History

- Bjarne Stroustrup
- 1979 | 1985
  - Developed at the Bells Labs
- Extension of C
  - C with “classes”
  - Object Oriented, Generic, Functional
- Current Standard C++20
  - We will use C++11 features!



IDEs

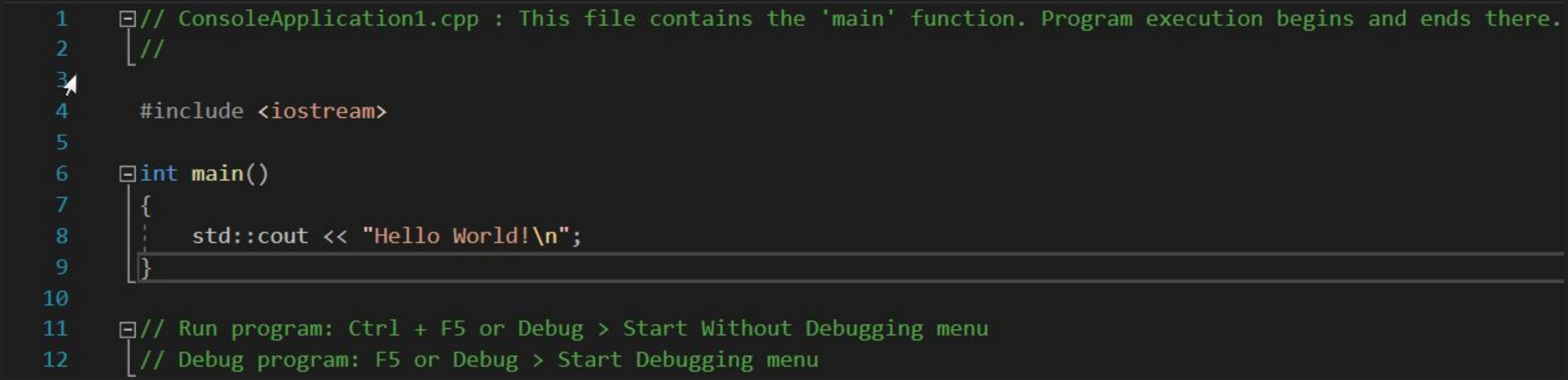
- Visual Studio
    - Modify the installation to include Desktop C++ Development
- Visual Studio Code
- CLion | Jetbrain
- Eclipse

Why we need C++

- Unreal!
- Computer Graphics
- Shaders
    - Unity and Unreal use CG which is a subset of C

Let’s Start

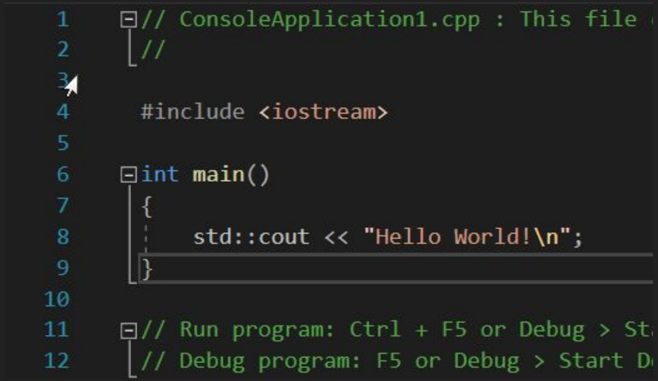
- Create on visual studio a new Console Application for C++
- It should something like that:



```
// ConsoleApplication1.cpp : This file contains the 'main' function. Program execution begins and ends there.
//

#include <iostream>

int main()
{
    std::cout << "Hello World!\n";
}

// Run program: Ctrl + F5 or Debug > Start Without Debugging menu
// Debug program: F5 or Debug > Start Debugging menu
```

Let's Start

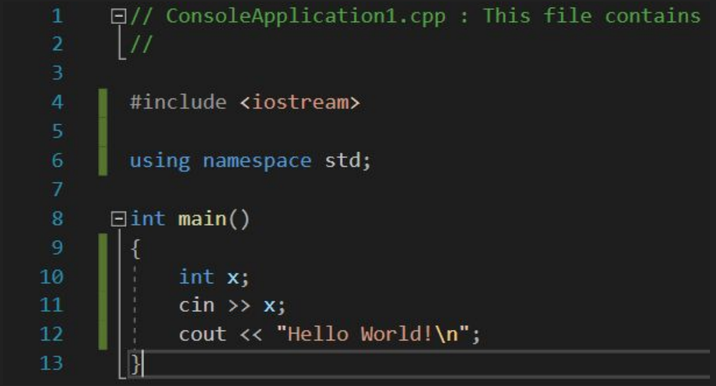
- Line 4: Includes the iostream library (File i/o)
- Line 6: A specialized function.
  - Sometimes returns a number for debugging
- Line 8: Types a message to the console
  - Std:: means find in the standard library the cout
  - Cout types a string or char value to the console



```
1   // ConsoleApplication1.cpp : This file
2   //
3
4    #include <iostream>
5
6   int main()
7   {
8       std::cout << "Hello World!\n";
9   }
10
11  // Run program: Ctrl + F5 or Debug > St
12  // Debug program: F5 or Debug > Start D
```

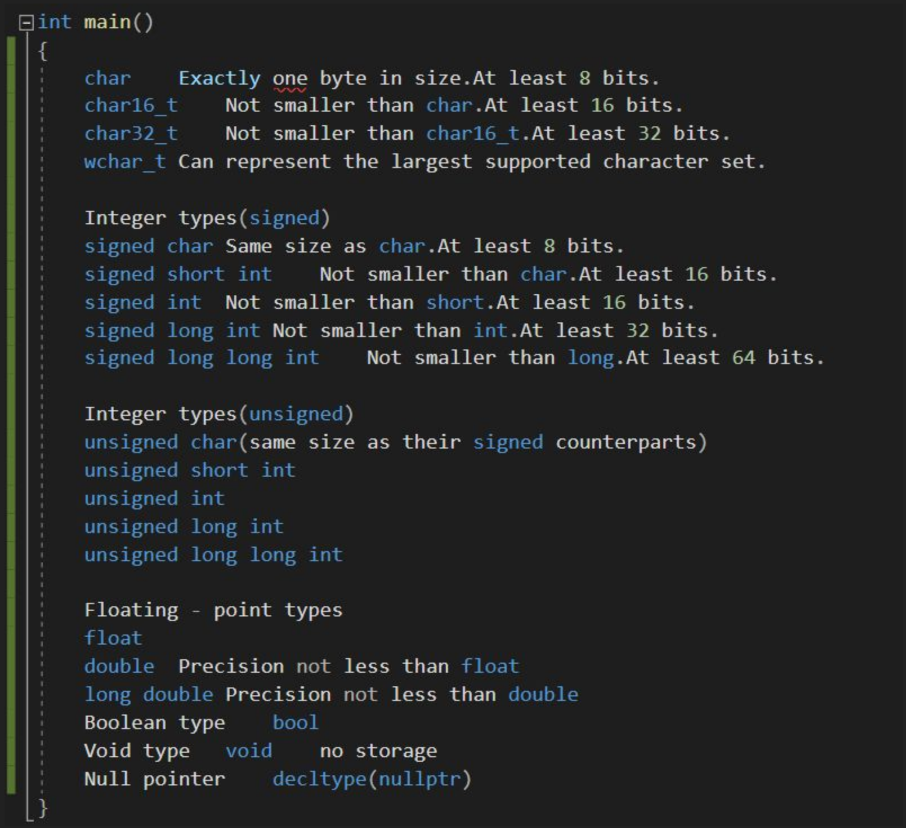
Let’s Read Something

- Line 6: Add the namespace standard
- Line 10: Declare an integer
- Line 11: Reads the input
    - >> Imagine the data flow going from console to the variable



```
1   // ConsoleApplication1.cpp : This file contains
2   //
3
4   #include <iostream>
5
6   using namespace std;
7
8   int main()
9   {
10      int x;
11      cin >> x;
12      cout << "Hello World!\n";
13  }
```

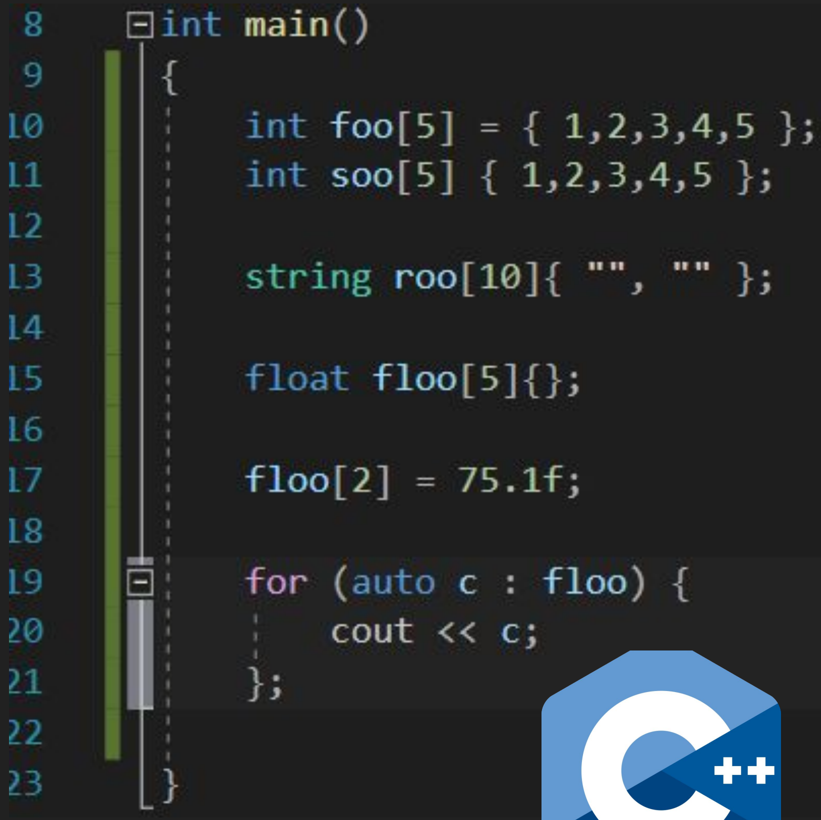
Data Variables



```
int main()
{
    char    Exactly one byte in size.At least 8 bits.
    char16_t    Not smaller than char.At least 16 bits.
    char32_t    Not smaller than char16_t.At least 32 bits.
    wchar_t Can represent the largest supported character set.

    Integer types(signed)
    signed char Same size as char.At least 8 bits.
    signed short int    Not smaller than char.At least 16 bits.
    signed int  Not smaller than short.At least 16 bits.
    signed long int Not smaller than int.At least 32 bits.
    signed long long int    Not smaller than long.At least 64 bits.

    Integer types(unsigned)
    unsigned char(same size as their signed counterparts)
    unsigned short int
    unsigned int
    unsigned long int
    unsigned long long int

    Floating - point types
    float
    double  Precision not less than float
    long double Precision not less than double
    Boolean type    bool
    Void type   void    no storage
    Null pointer    decltype(nullptr)
}
```

Data Structures

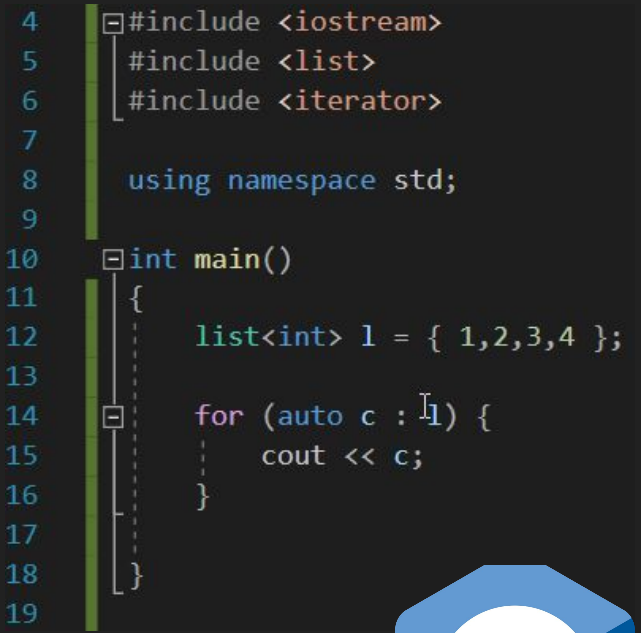
- Line 10 : Initialize an int array with 5 objects
- Line 11: A different way to init an array
- Line 13: Init a string array
- Line 15: Init a float array
- Line 17: Access an array
- Line 19: Read an array with foreach
    - In C++ there isn't a foreach identifier
    - We use the name for
    - The auto identifier means var
    - C++ converts c var to float



```
 8  int main()
 9  {
10      int foo[5] = { 1,2,3,4,5 };
11      int soo[5] { 1,2,3,4,5 };
12
13      string roo[10]{ "", "" };
14
15      float floo[5]{};
16
17      floo[2] = 75.1f;
18
19      for (auto c : floo) {
20          cout << c;
21      };
22
23  }
```

Data Structures | List

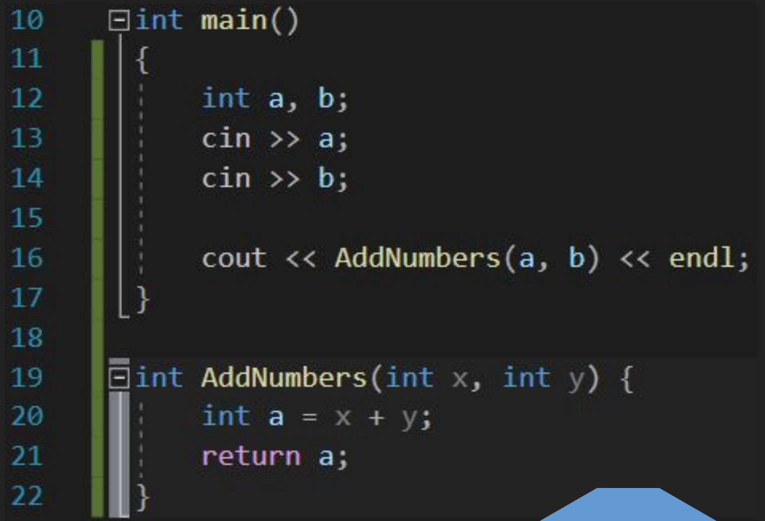
- Line 5-6 : Include the library for list functions
- Line 12: Initialize an int list
- Line 14: Print list elements with foreach loop



```
4   #include <iostream>
5   #include <list>
6   #include <iterator>
7
8   using namespace std;
9
10  int main()
11  {
12      list<int> l = { 1,2,3,4 };
13
14      for (auto c : l) {
15          cout << c;
16      }
17
18  }
19
```

Functions | Let’s Create a Simple Calc

- Line 12 : Create 2 int values
- Line 13 - 14: Read the Input
- Line 16: Type the result from the function AddNumber
    - Endl ends the line of the message
- Line 19: Function AddNumber with type int



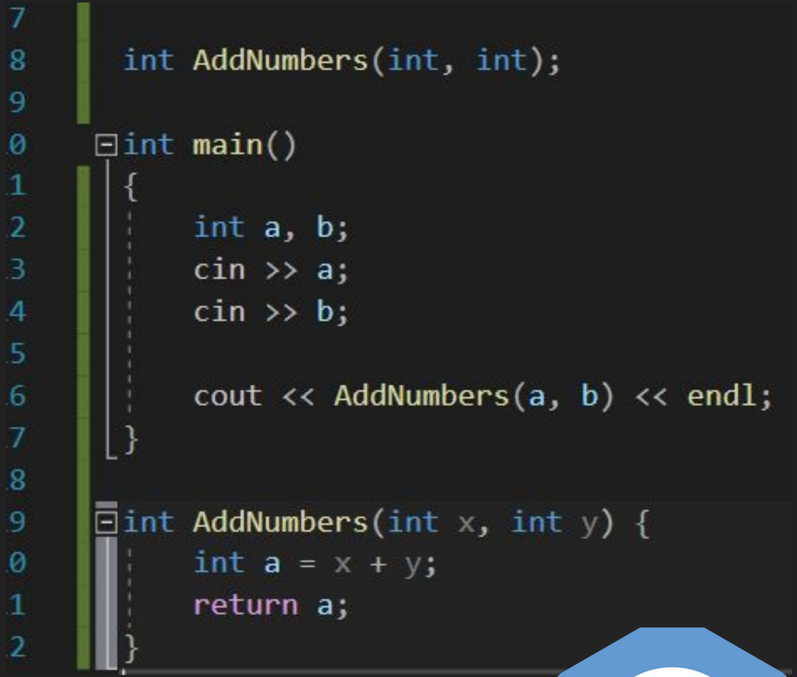
```
10  int main()
11  {
12      int a, b;
13      cin >> a;
14      cin >> b;
15
16      cout << AddNumbers(a, b) << endl;
17  }
18
19  int AddNumbers(int x, int y) {
20      int a = x + y;
21      return a;
22  }
```

Try to run this program. Does it run correctly or they are some errors?

Functions Identifiers

- In C++ we need function identifiers
- Functions identifiers are the templates of our functions

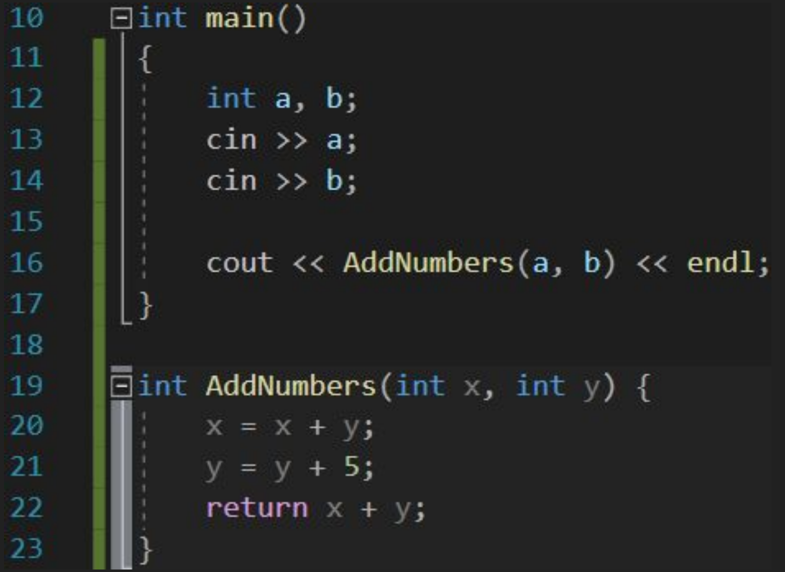
- C++ File Structure
  - Headers Files
  - Cpp Files



```cpp

int AddNumbers(int, int);

int main()
{
    int a, b;
    cin >> a;
    cin >> b;

    cout << AddNumbers(a, b) << endl;
}

int AddNumbers(int x, int y) {
    int a = x + y;
    return a;
}
```

Passing values

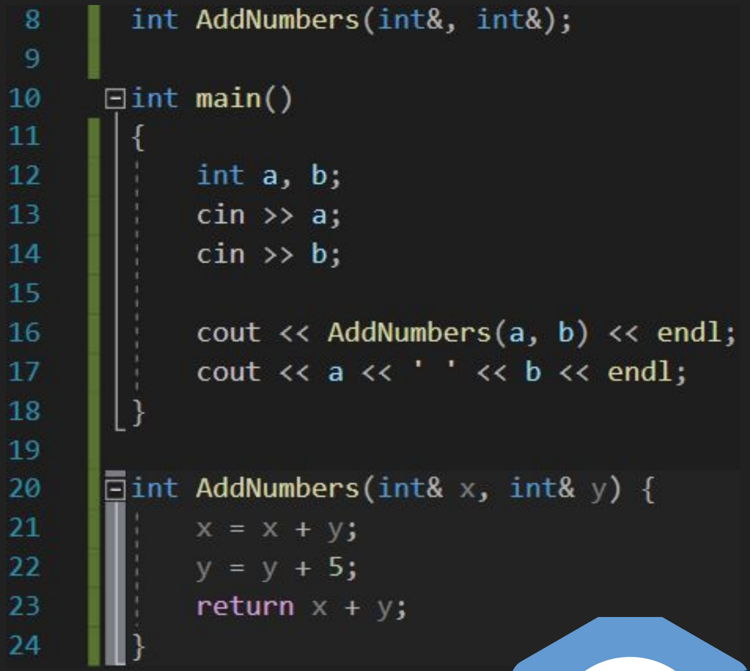
- Passing values
- The compiler duplicates the values from a,b to the values of x,y
- After the function returns the value, a and b Values remains the same



```
int main()
{
    int a, b;
    cin >> a;
    cin >> b;

    cout << AddNumbers(a, b) << endl;
}

int AddNumbers(int x, int y) {
    x = x + y;
    y = y + 5;
    return x + y;
}
```

Passing references

- You can pass a variable as reference with the Identifier &
- The compiler passes the variables a,b directly To the function
- After the function returns the value, a and b Values are not the same



```
int AddNumbers(int&, int&);

int main()
{
    int a, b;
    cin >> a;
    cin >> b;

    cout << AddNumbers(a, b) << endl;
    cout << a << ' ' << b << endl;
}

int AddNumbers(int& x, int& y) {
    x = x + y;
    y = y + 5;
    return x + y;
}
```

Challenges | Part 1

Να δημιουργηθεί πρόγραμμα που να καλωσορίζει τον χρήστη, και θα τον καθοδηγεί για να επιλέξει το κατάλληλο χρώμα LightSaber ανάλογα στοιχεία του χαρακτήρα του.

Αναλυτικότερα το πρόγραμμα θα πρέπει να :

- Να εμφανίζει εισαγωγικό μήνυμα για τη λειτουργία του προγράμματος.
- Να ζητάει όνομα από το χρήστη.
- Να ρωτάει τον χρήστη ποιός είναι ο χαρακτήρας του με βάση τα παρακάτω
  - Prefer negotiation and meditation, strong force abilities (Green)
  - Always angry and wants to destroy EVERYTHING (Red)
  - Jedi Guardians fighting for the light side and are skillful swordsmen (Blue)
  - Using Light and Dark side force, doing everything in order to keep the balance (Purple) (Chaotic Goo
- Να εμφανίζει τι χρώμα θα έχει το Lightsaber του χρήστη καθώς και το όνομά του.

Challenges | Part 2

Για την συγκεκριμένη άσκηση θα δημιουργήσετε έναν τηλεφωνικό κατάλογο ως πρόγραμμα κονσόλας. Το πρόγραμμα αρχικά θα καλωσορίζει τον χρήστη και κατόπιν θα των ρωτάει ποιος είναι ο (σταθερός) αριθμός των καταχωρήσεων.

Κάθε καταχώρηση περιέχει ένα όνομα και ένα τηλέφωνο.

Αρχικά ο κατάλογος είναι κενός. Στην συνέχεια, το πρόγραμμα θα εμφανίζει ένα μενού με τις βασικές επιλογές: εμφάνιση του καταλόγου, εισαγωγή/μεταβολή καταχώρησης, διαγραφή καταχώρησης, αναζήτηση με βάση το όνομα, έξοδος.

Για την υλοποίηση θα χρησιμοποιήσετε δύο arrays, ένα string[] και ένα int[], για τα ονόματα και τα τηλέφωνα, αντίστοιχα.

Next

- Object Oriented
- Classes and Structs
- Pointers

References

- https://en.wikipedia.org/wiki/C%2B%2B
- http://www.cplusplus.com/