C++ Fundamentals

Notes

- Reading:
  - Cplusplus.com Tutorial: Basics of C++
  - 5 sections: Program structure - Basic input / output
- Old notes: Fundamentals
  - Variables & operators
  - Strings
  - Input & output

Primitive types

- **Primitive types** are basic, built-in types. **Literals** are values for primitives.
- All variables must be **declared** with their type. Some primitive types:
  - int integer 4-bytes
  - float decimal numbers 4-bytes
  - char character 1-byte (use single-quote for char literal, e.g. 'A')
- Each type takes uses a certain amount of memory. This also determines the values that can be represented. Use **sizeof** to see how much memory each type uses on a particular platform.
- **Initialize variables** by giving them a value. Uninitialized variables will have random values.
- You can assign values to different primitive types all you want,
  - chars have an integer value (ASCII value)
  - assigning float to int will truncate (round down)
  - assigning char to int will assign ASCII value

demo	notes / output

`int a;`
 	
`cout << sizeof(a) << endl;`
`cout << a << endl;`
 	
`a = 5;`
	`// declare integer x`
	
	`4  // bytes (32 bits)`
	`?  // random value`
	
	`// initialize x`

```
float b = 3.14;

char c = 'A';

int d = 'A';
cout << d << endl;
```

```
// declare and initialize float y

// declare and initialize char z

// assign char value to int
65 // ASCII value of character 'A'
```

Operators

- Virtually identical in use to Python operators. The following are some distinctions.
- Increment / decrement: many ways...
- Exponentiation: include math library <math.h>

```
#include <iostream>
#include <math.h>
using namespace std;

int main(){
    int x = 2;

    // increment 3 times
    x++;
    x += 1;
    x = x + 1;
    cout << x << endl;

    // decrement 3 times
    x--;
    x -= 1;
    x = x - 1;
    cout << x << endl;

    // include math.h for pow function
    cout << pow( x, 3 ) << endl;
}
```

```
5

2

8
```

Chars & strings

- chars are initialized using single quotes, and strings are initialized using double-quotes.
- string elements are chars, and are accessible by index
- strings are mutable (contrast with Python)

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

void strings(){
        char c = 'a';
        string s = "Hello";
        char c2 = s[0];
        s[0] = 'M';

        cout << c << ' ' << s << ' ' << c2 << endl;
}
```

```
// chars use single-quote
// strings use double-quote
// accessing a char by string index
// c++ strings are mutable

a Mello H
```

Input & output

- **Notes:**
  - C++ has multiple input / output libraries.
  - The C++ standard is **iostream -** you need to know this.
  - Some C++ programs use the C-standard IO Library, **stdio.h** This is optional for you.
- **Whitespace, Printing:**
  - printing characters are letters, numbers, symbols, and whitespace
  - **whitespace** includes spaces, tabs, etc.
- non-printing characters include:
  - **newline ('\n')** which a programmer can send to the output stream, indicating that the console should insert a line break.
  - **carriage return ('\r')** which is inserted into the input stream when the user presses [ENTER].

I/O Streams

- An **output stream** is as it sounds - a stream of symbols that are sent to output.
  - **cout** is the standard output stream, which is printed to the console window.
  - **<<** is the **stream insertion operator,** which is used to insert items into an input stream
- An **input stream** is a stream of symbols that are collected from input.
  - **cin** is the standard input stream, which is read from the user input in the console.
  - **>>** is the **stream extraction operator.**
  - **>>** will pass over any whitespace and non-printing characters.
  - to see if input failed, check **cin.fail()**
- These operators can be chained.

Demo

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

int main()
{
	int i;
	char c;
	string s;
	string junk;

	cout << "Hello from cout!\n"
		<< "Give me an int: ";
	cin >> i;

	if( cin.fail() ){ 		// check for bad input
		cout << "Fail!\n";
		cin.clear();		// reset input stream
	}
	else{
		cout << "Got " << i << endl;
	}


	cout << "Give me a char: " ;
	cin >> c;
	cout << "Got '" << c << "'\n";

	cout << "Give me a string: ";
	cin >> s;
	cout << "Got \"" << s << "\"\n";

	return 0;
}
```

Functions

Review

- A **function** is a block of code that can be executed on command.
  - The function is executed when it is **called** / **invoked**
- A **function** may have zero, one, or more **parameters.**
  - A value passed to a function is called an **argument.**

C++ Functions

- The **function signature (header)** is the first line with the return type, function name, and list of parameters in parentheses ( ).
  - Functions that do not return any data should have return type **void.**

- The **function body** is the block of code after the header, enclosed in curly braces { }.

```
// function example
#include <iostream>
using namespace std;

int addition(int a, int b)
{
        int r;
        r=a+b;
        return r;
}

int main()
{
        int z;
        z = addition (5,3);
        cout << "The result is " << z;
        return 0;
}
```