122COM: Introduction to C++

David Croft

Coventry University

david.croft@coventry.ac.uk

2016

Coventry
University

Overview

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

1 Languages

2 C++
 ■ Variables

3 Syntax
 ■ Conditionals
 ■ Arrays
 ■ Loops
  ■ while
  ■ for
 ■ Compiling
 ■ Debugging

4 Recap

Coventry
University

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
  while
  for
Compiling
Debugging

Recap

Highs and lows

- Programming languages split into levels.
- Low level languages are machine code, assembly language.
- High level languages are Python, C++, Java etc.

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

I

High level languages

`i`+=1

⇓

Assembly

GOTO 0x42

INCF 0x68 0x01

SLEEP

⇓

Machine code

01010011 01100101 01110010 01101001

01101111 01110101 01110011 01101100 01111001

00111111 01111001 00111111

⇓

Hardware

Coventry University

C++

David Croft

Languages
C++
Variables
Syntax
Conditionals
Arrays
Loops
    while
    for
Compiling
Debugging

Recap

Languages

I

	High level	Python, Ruby Java C++ C Forth, Basic	3rd generation
	Low level	Assembly	2nd generation
		Machine code	1st generation
		Hardware	

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

Low level

Machine code

- 1st generation.
- Really hard to understand.
- Really hard to write.
- The actual instructions to the hardware.

Assembly

- 2nd generation.
- Hard for humans to understand.
- Hard for humans to write.
- 1-to-1 correspondence with what is run.

C++

David Croft

Languages
C++
  Variables
Syntax
  Conditionals
  Arrays
  Loops
    while
    for
  Compiling
  Debugging

Recap

High level

Python, C, C++, Java, PHP, Perl etc.

- 3rd generation.
- Favour programmer, not machine.
- Easy for humans to understand…compared to the alternatives.
- Easy for humans to write…compared to the alternatives.
- Portable.
  - Different machine == different compiler.
  - Same C/Python/C++/Java code.

So far you have used Python.
Now going to learn C++.

- 🔴 Created somewhere in 1979-1983.
- 🔴 Based on C (created 1972).
- 🟦 Going to be learning C++11 (approved 2011).
- 🔴 C++14 has been approved (2014).
    - 🟦 No support yet.
- 🔴 99.9% backwards compatible.
    - 🟦 All the way to C.

- 🟦 Supports the same paradigms as Python.
    - 🟦 Objected oriented, functional, declarative etc.

Coventry
University

C++

David Croft

Languages
C++
Variables
Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

Expectations

C

- All students are expected to learn some C++.
- In future weeks we will be looking at generic programming concepts.
  - Sorting.
  - Searching.
  - Data structures.
- Those weeks will be taught in Python and C++.
  - Everyone else will have some mandatory C++ tasks.
  - BIT & MC students can choose Python or C++ most tasks.
  - Will be specified at the time.
- BIT & MC will not be examined on C++ code.
  - May be examined on language differences.
  - High/low languages.
  - Compiling.
  - Static/dynamic typing.
  - Stack/heap memory.

Coventry
University

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

About C++ C

Most significant difference...

- C++ is statically typed.
  - Python is dynamically typed.

- In Python variables keep track of values AND type.

```
var = 42        # type(var) = <type 'int'>
var = 'foo'     # <type 'str'>
var = 0.123     # <type 'float'>
```

- In C++ variables have one type forever.
  - Have to specify type when creating.

```
int    var1 = 42;
string var2 = "foo";
float  var3 = 0.123;
```

C++

David Croft

Languages
C++
Variables
Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging
Recap

Data types · C

In C++ have to specify a variable's type.

- So what types are available?
- Thousands (at least).
    - You can create your own.
- Few standard ones.
- Most basic data types are called primitive types.

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
  while
  for
Compiling
Debugging

Recap

Primitive types

I

🟢 Knowing what the different variables are.

🟡 Knowing all the primitives and the variations.

🔴 Knowing ranges/sizes.

Type	Bytes	Values
bool	1	true/false
char	1	'a', 'Z', '6', '+'
int	4	-2147483647 → 2147483647
unsigned int	4	0 → 4294967295
float	4	1.234, -0.0001
double	8	1.23456789, -0.000000001
void		

Sizes are correct for a 32bit machine.

Moving from Python to C++.

- Not as bad/scary as it seems.
- Same basic structure.
- Slightly different syntax.

Coventry
University

C++

David Croft

Languages
C++
  Variables
Syntax
  Conditionals
  Arrays
  Loops
    while
    for
  Compiling
  Debugging

Recap

Hello World! C

Python.

```python
print('Hello World!')
```

```python
import sys

def main():
    print('Hello World!')

if __name___ == '__main__':
    sys.exit(main())
```

C++.

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

int main()
{
    cout << "Hello World!" << endl;

    return 0;
}
```

- All programs in C++ **MUST** have exactly one `main()` function.
- C++ uses { and } instead of indentation.
  - You should still have indentation in C++ but is aesthetic only.
- Semi-colons at the end of lines.

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

if statements

Same rules as Python.

- Slightly different syntax.
- and is now &&.
- or is now ||.
- == is still ==.

```python
a = 1
b = 2


if a == b and b > 0:
    print('Hello World )
```

```cpp
int a = 1;
int b = 2;


if( a == b && b > 0 )
{
    cout << "Hello World!" << endl;
}
```

Coventry University

C++

David Croft

Languages
C++
Variables
Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

Arrays C

Similar to Python lists.

■ Can't be resized.

```python
sequence = [1, 2, 42, 69, 8]
sum = 0

for i in range(len(sequence)):
    sum += sequence[i]
```

```c
int sequence[5] = {1, 2, 42, 69, 8};
int sum = 0;

for( int i=0; i<5; i+=1 )
{
    sum += sequence[i];
}
```

Coventry
University

C++

David Croft

Languages
C++
  Variables
Syntax
  Conditionals
  Arrays
  Loops
    while
    for
  Compiling
  Debugging

Recap

Arrays II   A

Three ways to create a C array.

1 Just supply size

```cpp
int     arrayOfInt[3];
char    arrayOfChars[5];
float   arrayOfFloats[2];
```

2 Supply size and initialisation list

```cpp
int     arrayOfInt[3]     = { 42, 69, 12 };
char    arrayOfChars[5]   = { 'A', 'z', '9' };
float   arrayOfFloats[2]  = { 1.23, 0.001, 8.1 };
```

3 Just initialisation list (will figure out the size)

```cpp
int     arrayOfInt[]      = { 42, 69, 12 };
char    arrayOfChars[]    = { 'A', 'z', '9' };
float   arrayOfFloats[]   = { 1.23, 0.001, 8.1 };
```

C++

David Croft

Languages
C++
  Variables
Syntax
  Conditionals
  Arrays
  Loops
    while
    for
  Compiling
  Debugging

Recap

New and improved!

So far looked at the old style arrays.
- Carried forward from C.
- Still used today.
- C++03 introduced an alternative.
  - STL arrays.

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

int main()
{
    int oldArray[5] = {1,2,3,4,5};
    array<int,5> newArray = {1,2,3,4,5};

    cout << oldArray[0] << " " << newArray[0] << endl;

    return 0;
}
```

Coventry
University

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

There's two of them?

I

Two types of arrays.
- Old style arrays are still very common.
  - Legacy.
  - Want you to start off using the new ones.

- What was wrong with the old ones?
- New arrays are safer.
  - Avoid overflows.
- Easier to use.
  - Sorting, searching, reversing, iterating etc.
- Are backwards compatible with old code.

Coventry
University

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

Vizic - **Vectors** ⬤ I

Coventry
University

Problem, C++ arrays have a set size.

- Saw we had to provide a size when declaring arrays.

C++ does have 'arrays' that can be resized.

- Called vectors.
- Uses arrays inside.

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

int main()
{
    array<int,5> myArray = {1,2,3,4,5};
    vector<int> myVector = {1,2,3,4};

    myVector.push_back(5);

    cout << myArray[0] << endl;
    cout << myVector[0] << endl;
}
```

C++ vectors are the closest thing to Python lists.
- If you are moving to C++ from Python easier to use vectors?

- append() → push_back() and emplace_back()
- pop() → pop_back()
- slicing → resize()

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

while loops

Same rules as Python.

- Slightly different syntax.
- Brackets ().
- Braces {}.
- Semicolons ;.

```python
counter = 0
while counter < 10:
    print('Hello World!')
    counter += 1
```

```cpp
int counter = 0;
while( counter < 10 )
{
    cout << "Hello World!" << endl;
    counter += 1;
}
```

Coventry
University

C++

David Croft

for loops   C

Languages

C++
  Variables

Syntax
  Conditionals
  Arrays
  Loops
    while
    for
  Compiling
  Debugging

Recap

C++ has two kinds of for loops.
- One type similar to Python for loops.
  - Actually a range-based loop.
  - Will be covered later.

- One type similar to a while loop.

C++

David Croft

Languages
C++
  Variables
Syntax
  Conditionals
  Arrays
  Loops
    while
    for
  Compiling
  Debugging

Recap

for loops

C

The original C++ for loop.

- Seems very different to the python loop.
- Lots of commonalities.
- Also to while loops.

```python
for counter in range(10):
    print('Hello World!')
```

```python
for counter in range(0,10,1):
    print('Hello World!')
```

```cpp
for( int counter=0; counter<10; counter+=1 )
{
    cout << "Hello World!" << endl;
}
```

```cpp
int counter = 0;
while( counter < 10 )
{
    cout << "Hello World!" << endl;
    counter += 1;
}
```

C++

David Croft

Languages
C++
Variables
Syntax
Conditionals
Arrays
Loops
while
for
Compiling
Debugging

Recap

Ranged for loops

The new C++11 ranged for loop, for iterating over a sequence.

- Less powerful that the old style.
- Easier.
- while > for > ranged for
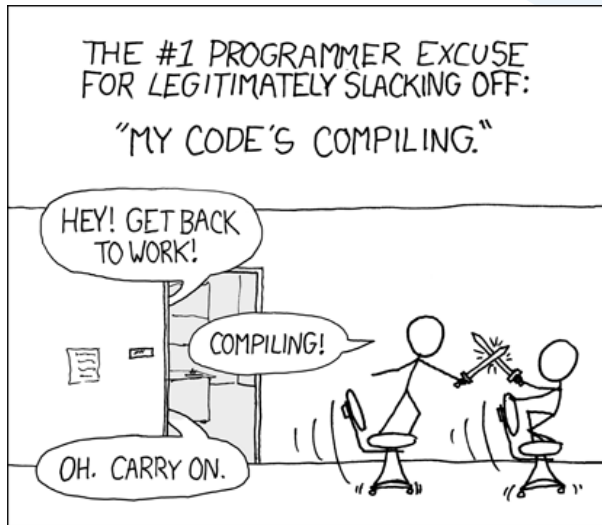
```
sequence = [1,2,3,4,5]
for i in sequence:
    print( i )
```

```cpp
int main()
{
    array<int,5> sequence =
        { 1, 2, 3, 4, 5 };
    for( int i : sequence )
    {
        cout << i << endl;
    }

    return 0;
}
```

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
   while
   for
Compiling
Debugging

Recap

Compiling C

C++ code has to be compiled before it is run.

- So does Python it just happens automatically.
- Compiler converts C++ code into machine code.
- Many IDEs handle compiling for you.
  - Visual Studio, Eclipse etc.

Coventry
University

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
   while
   for
Compiling
Debugging

Recap

gcc & g++   C

GNU C Compiler (created 1987).

- Linux, Mac and Windows.

How to compile using g++.

- Demo
- g++ –std=c++11 hello.cpp -o hello
    - g++ - the compiler program.
    - –std=c++11 - we want to use the C++11 standard of C++.
    - hello.cpp - the file we want to compile.
    - -o hello - the name of the executable to create.

Coventry
University

C++

David Croft

Languages

C++
Variables

Syntax
Conditionals
Arrays
Loops
   while
   for
Compiling
Debugging

Recap

Debugging   C

What if your code is wrong?

- Same as Python.
  - Syntax errors.
  - Runtime errors.
  - Logic errors.

  - Spot the errors.

```cpp
int main()
{
    cout << "Hi" << endl;

    for( int i=0; i>10; j+=1 )
    {
        cut << "Hello World!" << endl
    }

    return 0;
}
```

Quiz

C++

David Croft

Languages

C++
 Variables

Syntax
 Conditionals
 Arrays
 Loops
  while
  for
 Compiling
 Debugging

Recap

- C++ is a high level language.
  - 3rd generation.
- Compiled.
- Statically typed.
- Arrays cannot be resized.
  - Use new STL arrays.

Coventry
University

The End

Coventry
University