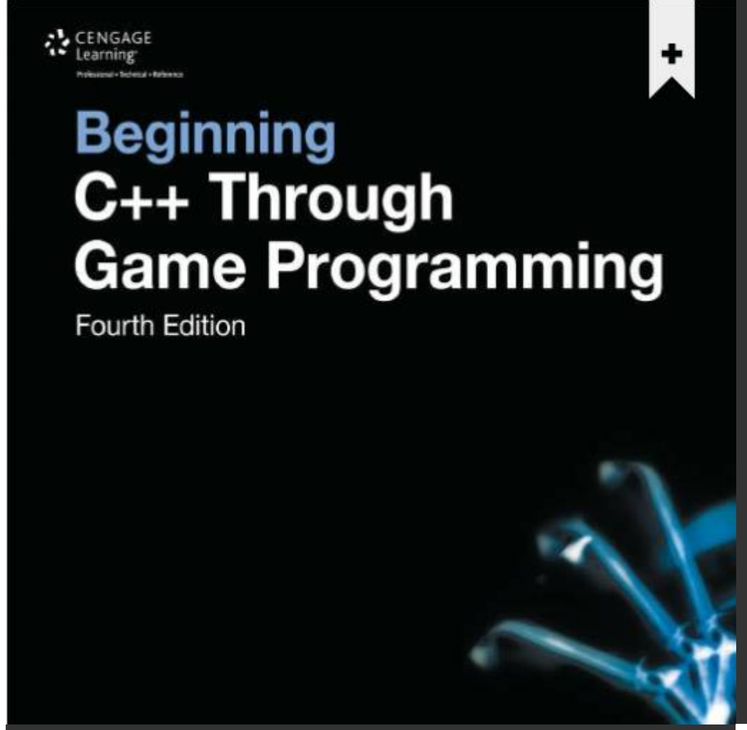
Contents

Recommended Reading

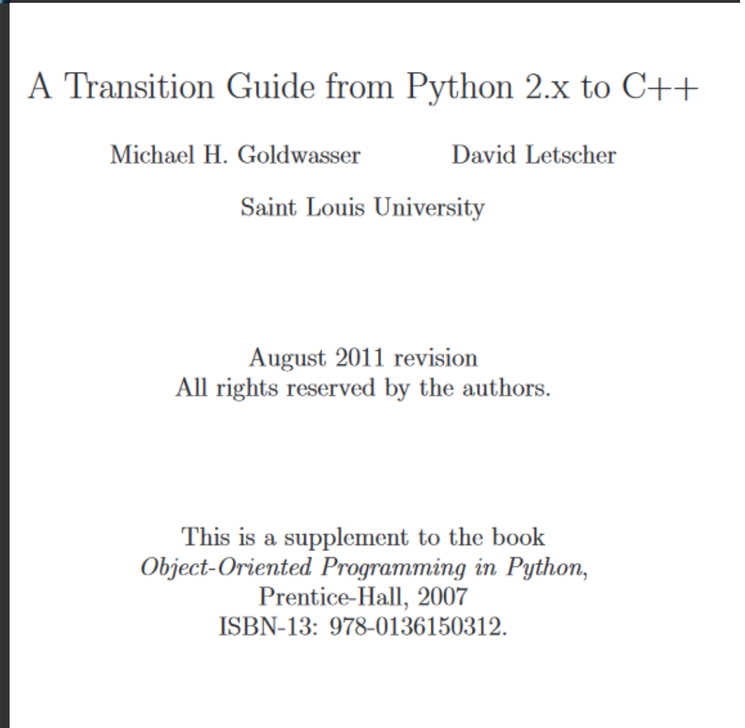


Beginning C++ Through Game Programming

Find on: FXPlus Online library

A transition guide from Python 2.x to C++

http://cs.slu.edu/~goldwamh/publications/python2cpp.pdf



A Transition Guide from Python 2.x to C++

Michael H. Goldwasser David Letscher

Saint Louis University

August 2011 revision

All rights reserved by the authors.

This is a supplement to the book *Object-Oriented Programming in Python*, Prentice-Hall, 2007

ISBN-13: 978-0136150312.

Data types

- **Integer**: Keyword used for integer data types is **int**. Integers typically requires 4 bytes of memory space and ranges from -2147483648 to 2147483647.

- **Character**: Character data type is used for storing characters. Keyword used for character data type is **char**. Characters typically requires 1 byte of memory space and ranges from -128 to 127 or 0 to 255.

- **Boolean**: Boolean data type is used for storing boolean or logical values. A boolean variable can store either *true* or *false*. Keyword used for boolean data type is **bool**.

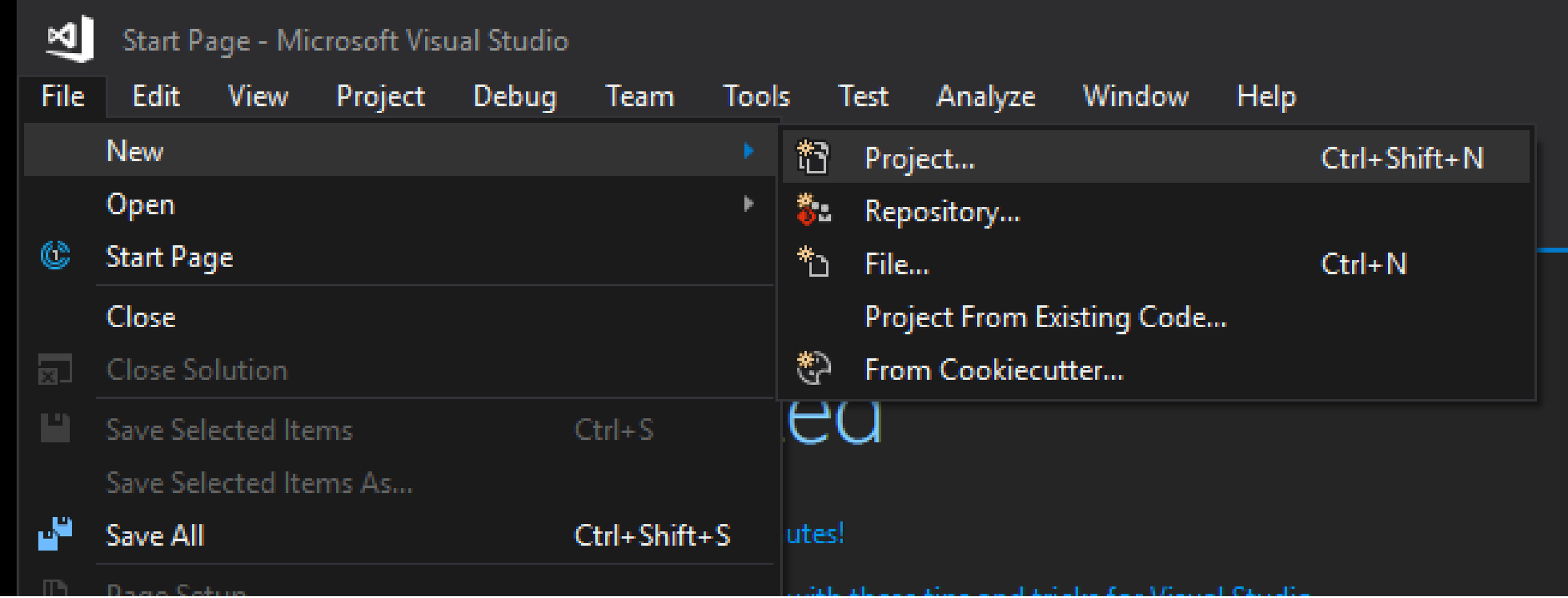
- **Floating Point**: Floating Point data type is used for storing single precision floating point values or decimal values. Keyword used for floating point data type is **float**. Float variables typically requires 4 byte of memory space.

- **Double Floating Point**: Double Floating Point data type is used for storing double precision floating point values or decimal values. Keyword used for double floating point data type is **double**. Double variables typically requires 8 byte of memory space.

- **void**: Void means without any value. void datatype represents a valueless entity. Void data type is used for those function which does not returns a value.

Control Statements

Operator	Symbol	Usage	Example
Equals	==	Returns true if the statements on either side of the operator are the same value, otherwise it returns false. Can be used with int, double/float, char, boolean, and string.	x == 'A'
Not Equal	!=	Returns true if the statements on either side of the operator are not the same, otherwise it returns false.	x != 2
Greater Than	>	Returns true if the first term is greater than the second, otherwise it returns false. Note: If both terms are equal it returns false.	x > 0
Less Than	<	Returns true if the first term is less than the second, otherwise it returns false.	x < 10
Greater Than or Equal	>=	Returns true if the first term is greater than or equal to the second, otherwise it returns false.	x >= 0
Less Than or Equal	<=	Returns true if the first term is less than or equal to the second, otherwise it returns false.	x <= -2
And	&&	Returns true if both terms on either side of the operator are true, otherwise it returns false. It is commonly used to concatenate statements together.	x && True (x == 2) && (y >= 0)
Or	||	Returns true if any of the terms on either side of the operator are true, otherwise it returns false.	x || y
Not	!	Negates the value of the statement that follows it.	!(x > 2)



Hello World

Creating a new C++ project in Visual Studio

Installed

- Visual C#
- Visual Basic
- Visual C++
  - Windows Desktop
  - Windows Universal
  - Cross Platform
  - MFC/ATL
  - Test
  - Other
- Visual F#
- SQL Server
- AWS
- Azure Data Lake
- JavaScript
- Azure Stream Analytics
- Python
- TypeScript
- Other Project Types

Online

Not finding what you are looking for?

Open Visual Studio Installer

Windows Console Application	Visual C++
Empty Project	Visual C++
Windows Desktop Application	Visual C++

Type: Visual C++

A project for a command-line .exe application that runs on Windows.

Name: Hello World

Location: C:\Users\profo\Documents\Visual Studio 2017\Projects

Browse...

Solution name: Hello World

☑ Create directory for solution

Cin
Cout

```
#include <string>                //needed for cin >> string
using namespace std;             //prevents need for std::
...
_____________________________________________
cout << "Hello";                 //display "Hello"
//std::cout<<"Hello";            //when not using 'using namespace std'
_____________________________________________
String username;
cin >> username;                 //get username from input
cout << "Hello, " + username;    //display "Hello, Joe Bloggs"
```

```
#include "pch.h"
#include <iostream>
#include <string>

using namespace std;


int main()
{
    cout << "enter your name: \n";
    string userName = "";
    cin >> userName;
    //getline(cin, userName);
    cout << "Hello, " + userName;
}
```

Hello, Joe

```
#include "pch.h"
#include <iostream>
#include <string>

using namespace std;


int main()
{
    cout << "enter your name: \n";
    string userName = "";
    //cin >> userName;
    getline(cin, userName);
    cout << "Hello, " + userName;
}
```

Hello, Joe Bloggs

```
#include "pch.h"
#include <iostream>
#include <string>

using namespace std;

void HelloWorld() {
    cout << "enter your name: \n";
    string userName = "";
    //cin >> userName;
    getline(cin, userName);
    cout << "Hello, " + userName;
}

int main()
{
    HelloWorld();
}
```

HelloWorld()

Hello, username

Functions should be declared before they are called

Loops

for

```
for (int i = 0; i < 5; i++)
{
            cout << "Hello, " + userName << endl;
}
```

while

```
int i = 0;
while(i < 5)
{
            i++;
            cout << "Hello, " + userName << endl;
}
```

Header Files
.h

Main.cpp

```
#include “AdderClass.h“

AdderClass AC;

int main() {
    int addResult = AC.adder(addValue1, addValue2);
}
```

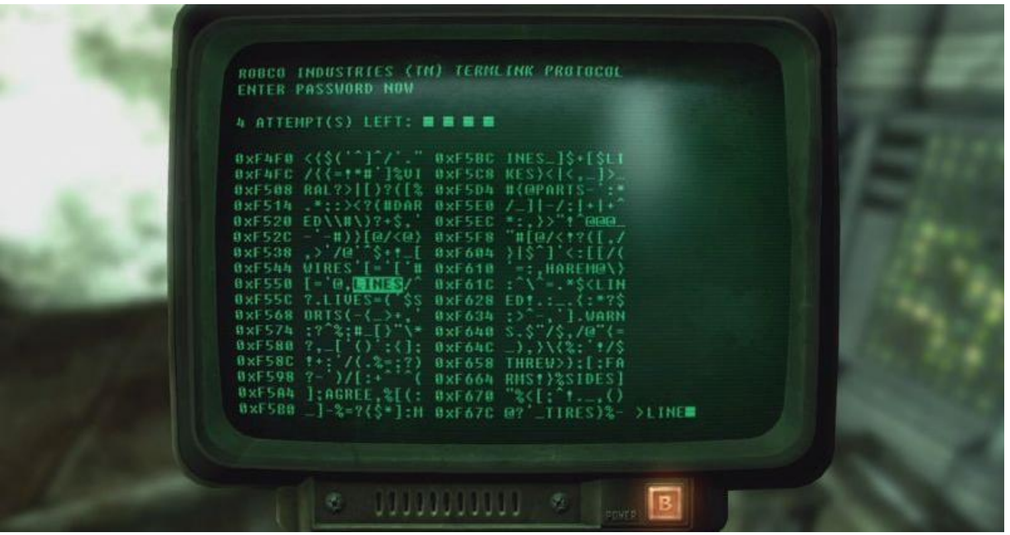
AdderClass.h

```
class AdderClass{
public:
    int adder(int a, int b);
};
```

NewClass.cpp

```
#include “AdderClass.h"

int testClass::adder(int a, int b) {
    return (a + b);
}
```



Terminal Hacking

https://github.com/Falmouth-Games-Academy/comp140-worksheetA/tree/master

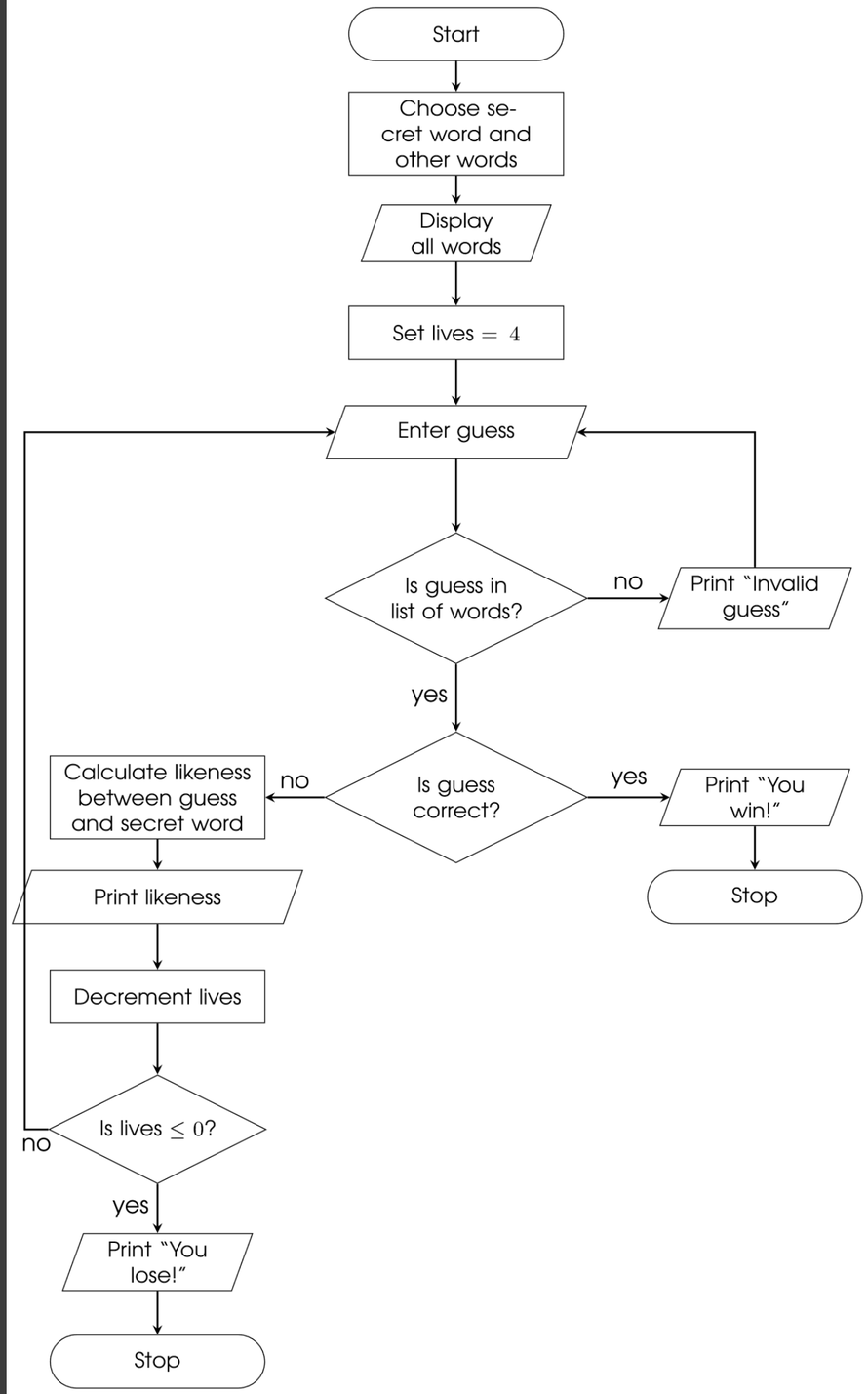
Reading in file data

Reading in files

```
std::ifstream wordFile(fileName);
```

Reading line by line

```
std::string word;
// Read each line in the file
while (std::getline(wordFile, word))
{
            ...
}
```



Terminal Hacking

Flowchart

Use your flowchart from Comp110: Flowcharts and Pseudocode,

or the attached flowchart