CS 520 Advanced Programming Languages
Fall Semester, 2009
Doc 8 C++ Classes
Sept 30, 2009

Reading

C++ Primer, 4 Ed, Lippman, Lajoie, Moo

Past Lectures
Chapters 2, 4, 5, 6 ,7

Future
Chapters
12
13
14
16
9 -11
17 parts
18 parts

Static Local Objects

```
int count_calls() {
        static int count = 0;
        return ++count;
}

int main() {
        for (int k = 0;k <= 5;k++)
                std::cout << count_calls() << '\n';
}
```

Output

1

2

3

4

5

6

Classes

```
class Stack  {
public:
        void push(int);
        float pop();
        Stack();
private:
        float stack[100];
        int topOfStack;
};

Stack::Stack():topOfStack(0) {}

void Stack::push(int item) {
        stack[topOfStack++] = item;
}

float Stack::pop() {
        return stack[--topOfStack];
}
```

```
int main() {
        Stack tryThisOut;
        Stack yours, mine;

        tryThisOut.push(5.0);
        yours.push(3.3);
        tryThisOut.push(9.9);

        std::cout
                << tryThisOut.pop()
                << std::endl;
}
```

Members

```
class Stack  {
public:
      void push(int);
      float pop();
      Stack();
private:
      float stack[100];
      int topOfStack;
};
```

Function members

Data members

Access Levels

Public member

accessible from inside and outside class

Private member

accessible from inside class only

Protected member

public to derived class, private to others

Multiple Access Sections

```
class Foo {
      int noAccessGiveMeansThisIsPrivate;
public:
      int x;
      int y;
private:
      std::string name;
public:
      int z;
};
```

Convention
public first
private last

Implicit Inline

```
class Stack  {
private:
        float stack[100];
        int topOfStack;

public:
        void push(int item) {
                stack[topOfStack++] = item;
        }

        float pop() {
                return stack[--topOfStack];
        }

        Stack():topOfStack(0) {}
};
```

Implicit Inline - Bad Bad Bad

Large programs become hard to read

Compile issues

Better but not Good

```
class Stack  {
public:
        void push(int);
        float pop();
        Stack();
private:
        float stack[100];
        int topOfStack;
};

Stack::Stack():topOfStack(0) {}

void Stack::push(int item) {
        stack[topOfStack++] = item;
}

float Stack::pop() {
        return stack[--topOfStack];
}
```

Use Separate Files

stack.h

```
#ifndef _stack_h_
#define _stack_h_

class Stack  {
public:
        void push(int item);
        float pop();
        Stack();
private:
        float stack[100];
        int topOfStack;
};

#endif
```

stack.cc

```
#include "stack.h"

Stack::Stack():topOfStack(0) {}

void Stack::push(int item) {
        stack[topOfStack++] = item;
}

float Stack::pop() {
        return stack[--topOfStack];
}
```

main.cc

```
#include <iostream>
#include "stack.h"

int main() {
	Stack tryThisOut;

	tryThisOut.push(5.0);
	tryThisOut.push(9.9);

	std::cout << tryThisOut.pop() << std::endl;
}
```

Compiling

C++ File Extensions

file.cc
file.cp
file.cxx
file.cpp
file.CPP
file.c++
file.C

You compiler may vary

GCC

GNU Compiler Collection

http://gcc.gnu.org/

gcc

C Compiler

c++, g++

C++ compiler

Location on rohan

/usr/local/bin/

Simple Compile

hello.cc

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

int main() {
	cout << "Hello" << endl;
}
```

```
->g++ hello.cc
->a.out
Hello
```

```
->g++ hello.cc -o a.out
->a.out
Hello
```

```
->g++ hello.cc -o hello
->hello
Hello
```

stack.h

```
#ifndef _stack_h_
#define _stack_h_

class Stack  {
public:
        void push(int item);
        float pop();
        Stack();
private:
        float stack[100];
        int topOfStack;
};

#endif
```

stack.cc

```
#include "stack.h"

Stack::Stack():topOfStack(0) {}

void Stack::push(int item) {
        stack[topOfStack++] = item;
}

float Stack::pop() {
        return stack[--topOfStack];
}
```

main.cc

```
#include <iostream>
#include "stack.h"

int main() {
        Stack tryThisOut;

        tryThisOut.push(5.0);
        tryThisOut.push(9.9);

        std::cout << tryThisOut.pop() << std::endl;
}
```

->g++ stack.cc main.cc
->a.out
9

g++ stack.cc main.cc -o stack
->stack
9

Linking Binaries

```
->g++ -c stack.cc

->g++ stack.o main.cc -o runStack
->runStack
9
```

More On Classes

C++ is Not Java

```
class Stack  {
public:
        void push(int item);
        float pop();

private:
        float stack[100];
        int topOfStack = 0;          //Compile Error
};
```

Constructors

ClassName: initializers { assignments }

```
Stack::Stack():topOfStack(0) {}
```

```
Stack::Stack() {
    topOfStack = 0;
}
```

Initializing

Variables of Class type are always initialized

const & reference types can not be assigned

Initializer Example

```
class Foo {
public:
        int k;
        const int const_k;
        int &reference_k;
        Stack list;
        Foo(int);
};

Foo::Foo(int start) {
        //list is already initialized
        k = start;
        const_k = start;          //error
        reference_k = start;      //error
}
```

```
Foo::Foo(int start):
        const_k(start  * 10),
        reference_k(start),
        list(start)
{
        k = start;
}
```


Compiler gave warning when list(start) was not last.