Welcome to CSci 1113

Introduction to C/C++ Programming for Scientists and Engineers
Instructor (me)

James Parker
Walter Library B18T

Primary contact:
jparker@cs.umn.edu
Problem Solving
With C++,
Walter Savitch,
9th edition
CSE Labs account

You need a CSE Labs account to participate in labs in this course

Lab attendance is mandatory
http://cse1labs.umn.edu
http://cse.labs.umn.edu

CSE Public Computer Classrooms and Labs

Account Information

The College of Science and Engineering provides and maintains publicly accessible computing labs in support of education, work, and limited research (see Usage Policies) for its faculty, staff, and students.

Eligibility

CSE Labs accounts are available to:

- All students who are currently enrolled in the College of Science and Engineering
- Faculty in the College (email operator at operator@cse.labs.umn.edu - we will open one for you)
- Students taking courses within the college that require CSE Labs access

Account Creation

CSE Labs use is open to any student currently enrolled in the College of Science and Engineering.

You can open a CSE Labs Account using one of the following two methods:

- Online: You can open an account online by completing the CSE Labs Account Creation Form.
- Labs: You can also open a CSE Labs account by visiting one of the CSE Labs. Enter "CSLabs" for the username, and "cse" for the password, to log into one of the workstations. Once you have logged in, complete the Account Creation Form that is presented.

Acceptable Use Policies
http://cse1abs.umn.edu
Welcome to the Fall2012 CSE Labs Account Creation Form.

Use this form to initiate or change your CSE Labs account for the Fall2012 semester. CSE Labs use is open to any student currently enrolled in the College of Science and Engineering.

Please enter the following information:

- Your student email **username**.
- Your **password** for your general UMN email account. (To verify your eligibility for a CSE Labs account.)

Username: park0580@umn.edu
Password: 

If you do not know what your username is, or you are having problems see the **U of M Student Internet Account Initiation Form**.

For further information send email to **operator@cselabs.umn.edu** or stop by the Systems Staff Office in Keller Hall 1-213.

For a list of our hours see **Systems Staff Contact Information and Hours**.

Submit
CSELabs account

CSELabs account used in lab (first lab ensures account working)

Register ASAP

Problems? Bug operator@cselabs.umn.edu
Class website

www.cs.umn.edu/academics/classes
Or google “umn.edu csci class”

Syllabus, schedule, other goodies

Moodle page will have grades and
Possibly homework submission
CSci 1113: C++ Programming

Class Announcements

09/08/2015
ALL YOUR BASE ARE BELONG TO US!
Syllabus

5% Labs (Wednesdays/Thursdays)
30% Homework (due Wednesday)
5% Quiz (Oct. 6)
15% Midterm 1 (Oct. 20)
15% Midterm 2 (Nov. 24)
30% Final (Dec. 22)
Syllabus

Each week there will be either a homework due or a test (except next).

Homework is due Wednesday at 5:30 P.M. on moodle.

Late homework is not accepted, but we will drop the lowest.
Labs will only be graded within your designated lab.

Homework must be coded individually.

Don't cheat.
Syllabus

Grading scale:

- 93% A
- 90% A-
- 87% B+
- 83% B
- 80% B-
- 77% C+
- 73% C
- 70% C-
- 67% D+
- 60% D
- Below F
Schedule

Ch. 1: Introduction, Programs, Compilers
Ch. 2: Input/Output, Data, Expressions
Ch. 3: Control Flow (if and loops)
Ch. 4, 5: Functions (return values)
Ch. 6: File I/O
Ch. 7, 8: Arrays and Strings
Ch. 9: Pointers and Dynamic Arrays
Ch. 10&11: Classes and Operator Overloading
Ch. 14&15: Recursion & Inheritence
Syllabus

Any questions?
What can I program?

If you can think of an explicit process (of simple steps) to solve your problem, then it can be programed.
Banana Nut Bread

Ingredients

* 3 or 4 ripe bananas, smashed
* 1/3 cup melted butter
* 1 cup sugar
* 1 egg, beaten
* 1 teaspoon vanilla
* 1 teaspoon baking soda
* Pinch of salt
* 1 1/2 cups of all-purpose flour
Banana Nut Bread

Directions
1. Preheat the oven to 350°F (175°C).
2. Mix butter into the mashed bananas in a large mixing bowl.
3. Mix in the sugar, egg, and vanilla.
4. Sprinkle the baking soda and salt over the mixture and mix in.
5. Add the flour and nuts last, mix.
6. Pour mixture into a buttered 4x8 inch loaf pan.
ATMs

How do you get change for $18.26 with the least amount of bills and coins?
Repetitive tasks

If you feel like a mindless zombie when you do it a lot, you can probably program it.
Repetitive tasks
Repetitive tasks
Repetitive tasks
Auto leveling?
Software vs Hardware

Software - the more intangible code on a computer

Hardware - the physical Parts of the computer
Hardware interaction

Input

CPU

Memory

Output
Memory addressing

Data is stored in “addresses” inside the memory

Later in this class, we will use these addresses to manipulate and share data
Memory addressing
Object oriented programming

OOP - focus on data and how they interact

To make algorithms for OOP, it is often useful to identify the data you are working with and their relationships before programming
Object oriented programming

Data for...

Banana nut bread?
ATM?
Ball game?
Data for...

Banana nut bread? Ingredients
ATM?
Ball game?
Object oriented programming

Data for...

Banana nut bread?  Ingredients
ATM?             Dollars & coins
Ball game?
Object oriented programming

Data for...

Banana nut bread?  Ingredients
ATM?  Dollars & coins
Ball game?  Balls & mouse
Object oriented programming

Data for...

Banana nut bread? Ingredients
ATM? Dollars & coins
Ball game? Balls & mouse

Lots of pixels (tiny color dots)
Compiling

Converting code to binary is called compiling.
Compiling

Often this compiled code Will not work on other computers
Compiling

C++ is a high level language (human readable)

Compiling changes a high level language into a low level language that is easier for the computer (computer cannot run high level)
Compiling

Your source code is the original language you wrote your program in (the C++ code for us)

You must recompile the source code every time you save a change before running the program again
#include <iostream>
using namespace std;

int main ()
{
    cout << "Hello World! ";
    return 0;
}
MODEL SMALL
IDEAL
STACK 100H

DATASEG
MSG DB 'Hello, World!', 13, '$'

CODESEG
Start:
MOV AX, @data
MOV DS, AX
MOV DX, OFFSET MSG
MOV AH, 09H ; output ascii string
INT 21H
MOV AX, 4C00H
INT 21H
END Start
Ease of use
Why C++?

Speed

Control

Libraries
Speed

Not all programming languages need to compile code as C++ (Java, Python)

Compiling can greatly increase speed
C++ allows you great control over your data (and its interpretation)

This comes with a burden of responsibility to properly manage your data.
Libraries

C++ is a widely used language and you can easily find supporting code segments for your program.

Continues to develop and increase performance.
Java vs C++

**Java**
- Goes anywhere
- Comfy

**C++**
- Fast
- Fine tuned
Object Oriented

Main focus is on **objects** and how they interact

Reusable groups of actions between objects are called **functions**

These actions can take additional information called **arguments**
One format is:
`object.function(argument, argument...);`

Example:
`James.teaches(CSci 1113);`
`teach(CSci 1113);`

The dot (period) shows that “teaching” is an action done by “James”
Banana Nut Bread

Ingredients

* 3 or 4 ripe bananas, smashed
* 1/3 cup melted butter
* 1 cup sugar
* 1 egg, beaten
* 1 teaspoon vanilla
* 1 teaspoon baking soda
* Pinch of salt
* 1 1/2 cups of all-purpose flour
* 1 cup of nuts
Banana Nut Bread

Directions
1. Preheat the oven to 350°F (175°C).
2. Mix butter into the mashed bananas in a large mixing bowl.
3. Mix in the sugar, egg, and vanilla.
4. Sprinkle the baking soda and salt over the mixture and mix in.
5. Add the flour and nuts last, mix.
6. Pour mixture into a buttered 4x8 inch loaf pan.
Banana Nut Bread

Directions
1. Preheat the oven to 350°F (175°C).
2. Mix butter into the mashed bananas in a large mixing bowl.
3. Mix in the sugar, egg, and vanilla.
4. Sprinkle the baking soda and salt over the mixture and mix in.
5. Add the flour and nuts last, mix.
6. Pour mixture into a buttered 4x8 inch loaf pan.
Banana Nut Bread

Directions
1. Preheat the oven to 350°F (175°C).
2. Mix butter into the mashed bananas in a large mixing bowl.
3. Mix in the sugar, egg, and vanilla.
4. Sprinkle the baking soda and salt over the mixture and mix in.
5. Add the flour and nuts last, mix.
6. Pour mixture into a buttered 4x8 inch loaf pan.
Banana Nut Bread

Pseudo code directions
1. oven.preheat(350);
2. bowl.mix(butter, bananas);
3. bowl.mix(sugar, egg, vanilla);
4. bowl.sprinkle(baking soda, salt);
5. bowl.mix(flour, nuts);
6. bowl.pour(pan);
7. pan.bake(60);
8. pan.cool();
Banana Nut Bread

Pseudo code directions #2

1. oven.preheat(350);
2. bowl.add(butter, bananas);
3. bowl.mix();
4. bowl.add(sugar, egg, vanilla);
5. bowl.mix();
6. bowl.sprinkle(baking soda, salt);
7. bowl.add(flour, nuts);
8. bowl.mix();
9. pan.pour(bowl);
10. pan.bake(60);
11. pan.cool();
mashedBananas = bananas.mashed();
bowl.add(butter, mashedBananas);
same as:
bowl.add(butter, bananas.mashed());

Kitchen.bowl.add(butter, bananas.mashed());

hand.mix(butter, mashedBananas);
bowl.add(hand.mix(butter, mashedBananas));
Magic 8 ball
Magic 8 ball

What a rip off!
#include <iostream>
using namespace std;

int main()
{
    cout << "Maybe.";
    return 0;
}
Keyboard input

```cpp
cout << "word"
    - prints "word" to the screen

cin >> x
    - store what is typed into "x"
    (x is some object or data)
```
Types of errors

Syntax error - code will not compile
  e.g.   `cout("hi");`

Runtime error - code crashes after starting
  e.g.   `(0 input to number.cpp)`

Logic error - code runs but doesn't return
  the correct answer
  e.g.   `addition.cpp`
Syntax

Syntax is a fancy word for the “grammar” of programming languages.

The basic English syntax is: (subject) (verb) (noun)
“I eat bananas” not “Bananas I eat”
Types of errors

To remove your program of bugs, you should try to test your program on a wide range of inputs.

Typically it is useful to start with a small piece of code that works and build up rather than trying to program everything and then debug for hours.
Comments

Comments are ignored pieces of code

// denotes a single line that is commented
// (everything before hitting enter)

/* denotes the beginning of a comment
and the end of a comment is denoted by */
“White space” is ignored (this is why we need a semi-colon)

Braces denote a block of code \{ \} (belonging to a method, class, etc.)