Course Introduction
CMSC 202 - Computer Science II

Instructor & Lecture Section
• Dr. Christopher Marron
  o Section 01: MoWe, 2:30 — 3:45 pm, LH1
  o Office: ITE 359
  o Email: cmarron@umbc.edu
• Office Hours:
  o Mo 1300 – 1400
  o Tu 1430 – 1530
  o We 1300 – 1400

What is CMSC 202?
• An introduction to
  – Object-oriented programming (OOP) and object-oriented design (OOD)
  – Basic software engineering techniques
• Emphasis on proper program design and maintainability
• Tools
  • C++ programming language, GCC (Gnu Compiler)
  • Linux (GL system)
Course Web Site and Blackboard

Links to syllabus, schedule, projects, and labs:
http://www.csee.umbc.edu/courses/undergraduate/210/Fall15_marron/

All grades will be posted on Blackboard.

Review of the Syllabus

Procedural vs. OO Programming

**Procedural**
- Modular units: functions
- Program structure: hierarchical
- Data and operations are not bound to each other
- Examples:
  - C, Pascal, Basic, Python

**Object-Oriented (OO)**
- Modular units: objects
- Program structure: a graph
- Data and operations are bound to each other
- Examples:
  - C++, Java, Python (huh??)
What’s an Object?

• Must first define a class
  – A data type containing:
    • Attributes – make up the object’s state
    • Operations – define the object’s behaviors

<table>
<thead>
<tr>
<th>Bank Account</th>
<th>String</th>
</tr>
</thead>
<tbody>
<tr>
<td>account number</td>
<td>sequence of characters</td>
</tr>
<tr>
<td>owner’s name</td>
<td></td>
</tr>
<tr>
<td>balance</td>
<td></td>
</tr>
<tr>
<td>interest rate</td>
<td>compute length</td>
</tr>
<tr>
<td>more</td>
<td>more</td>
</tr>
<tr>
<td>deposit money</td>
<td>concatenate</td>
</tr>
<tr>
<td>withdraw money</td>
<td>test for equality</td>
</tr>
<tr>
<td>check balance</td>
<td>more</td>
</tr>
<tr>
<td>transfer money</td>
<td>more</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Type</th>
<th>Attributes</th>
<th>Operations</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>(state)</td>
<td>(behaviors)</td>
</tr>
</tbody>
</table>

So, an Object is...

• A particular instance of a class

<table>
<thead>
<tr>
<th>Marron’s Account</th>
<th>Kukla’s Account</th>
<th>Park’s Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>12-345-6</td>
<td>05-432-1</td>
<td>43-261-5</td>
</tr>
<tr>
<td>Chris Marron</td>
<td>James Kukla</td>
<td>John Park</td>
</tr>
<tr>
<td>$1,250.86</td>
<td>$5.50</td>
<td>$825.50</td>
</tr>
<tr>
<td>1.5%</td>
<td>2.7%</td>
<td>2.5%</td>
</tr>
</tbody>
</table>

For any of these accounts, one can...
• Deposit money
• Withdraw money
• Check the balance
• Transfer money

Why C++ for 202?

• Popular modern OO language
• Wide industry usage
• Used in many types of applications
• Desirable features
  – Object-oriented
  – Portable (not as much as Java, but fairly so)
  – Efficient
  – Retains much of its C origins
Some C++ Background

- Created in 1979 by Bjarne Stroustrup of Bell Labs (home of UNIX and C).
- Added object-oriented features to C.
- Renamed to C++ in honor of auto-increment operator.
- Later standardized with several International Organization for Standards (ISO) specifications.

Interpreters, Compilers, and Hybrids

Interpreted Languages (e.g., JavaScript, Perl, Ruby)
- Interpreter translates source into binary and executes it
- Small, easy to write
- Interpreter is unique to each platform (operating system)

Compiled Languages (e.g., C, C++)
- Compiler compiles code into binary
- Compiler is platform dependent
- JVM is an interpreter that is platform independent

Many other models: e.g., Java (Python is stranger still):
- Source code
- Bytecode
- Java Virtual Machine (JVM)

C++ Compilation & Linkage

Any text editor

C++ source code

Linux C++ compiler

Linux C++ code library

Linux C++ binary

Linux linker

Linux executable code

Windows C++ compiler

Windows C++ code library

Windows C++ binary

Windows linker

Windows executable code
Python vs. C++ Syntax

**Python**

```python
print "Hello, world"
quotient = 3 / 4
if quotient == 0:
    print "3/4 == 0",
else:
    print "3/4 != 0"
```

**C++**

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

int main() {
    int quotient;
    cout << "Hello, world";
    quotient = 3 / 4;
    if (quotient == 0) {
        cout << "3/4 == 0";
    } else {
        cout << "3/4 != 0";
    }
    return 0;
}
```

Elements of C++

- Procedural and OOP elements
- Must have a "main()" function
- Statements end with ";"
- Variables must be declared
- "Python" syntax different
- Statement blocks demarcated by "{...}"
- Much that is similar

Development Environment

- You will use the GL Linux systems and GCC (GNU Compiler Collection) suite for development.
- You will learn to be semi-literate in Linux and shell usage.
- You will learn to use a text editor — Emacs is recommended.
- You may use IDEs such as Eclipse or XCode, but support will not be provided, and...

  *Your programs must compile and function correctly on the GL Linux systems.*

Challenges

- Knowing and following the schedule and course policies.
- Getting used to the Linux environment (tends to hit transfer students hardest).
- Starting projects early.
- Thinking all that matters is the projects.
- Waiting to late to seek help.

[https://youtu.be/WVvK9qSXT-g](https://youtu.be/WVvK9qSXT-g)