**CSC 112: Fundamentals of Computer Science**  
**Syllabus: Spring 2017**

**Instructor:** Pete Santiago  
**Office:** Manchester 236  
**Phone:** Office 336-758-4190  
**Email:** ps@wfu.edu  
**Class; Lab:** Tuesday & Thursday 2:00 – 3:15; Thursday 3:30 – 4:45. Manchester 229  
**Office hours:**  
- Scheduled TBA (but usually around)  
- Drop in I will make time if possible.  
- By appointment I try to be available as needed, so email.  
- Recitation/review As needed or as requested by student groups (evenings are fine).  
- Via email Simple and convenient.  
- Via Sakai Post questions or to start discussions. Classmates can be a big help.  
**TA:** Alec White whitak13@wfu.edu  
**Office hours:** TBA - CSC Center, Manchester 229.  
**CS Center:** Assistance with labs, assignments, programs, and the concepts is available with the Teaching Assistants in the CS Center. Check the Department Calendar on the website for CS Center hours.  
**CS Department staff resources:** Paul Whitener (server issues); Steven Wicker (Sakai and desktop).

**Course Description**  
This course, which includes a lab component (accounting for the 4th hour of credit), begins by introducing students to the C++ programming language and reinforcing the fundamental programming ideas introduced in CSC 111 - Introduction to Computer Science. The course extends CSC 111, particularly with respect to: building correct and efficient programs to solve real-world problems; refining problem solving and algorithm development skills; providing experience in developing and debugging larger programs; efficiently using techniques for dynamically requesting and structuring memory. A detailed list of topics is given at the end of the syllabus.

CSC 112 lays the groundwork for advanced courses in computer science, such as CSC 221 (Data Structures and Algorithms), where the focus is almost exclusively on designing, developing, and using different memory structures to manage large data sets.

**Learning Outcomes**  
By the end of this course, students should be able to:  
- use logical problem-solving, algorithm development, and programming basics  
- design and implement increasingly complex programs using top-down design, functional decomposition, procedural programming, and object-oriented programming. Develop programs using the core procedural and object-oriented aspects of the C++ programming language.  
- efficiently and effectively debug code.  
- employ commonly available software development tools for C++ program development.  
- use basic ideas of algorithmic complexity to analyze simple searching and sorting algorithms.
• define, implement, and employ the linear data structures of arrays, linked lists, stacks, and queues.
• define the algorithmic concept of recursion and employ recursion to solve problems.
• define the notion of dynamic memory management, choose when dynamic memory management is appropriate, and employ dynamic memory management to solve problems.

Text: The primary text for this class is the online text from zyBooks: Programming in C++. Instructions for access are:
1. Sign up at zyBooks.com
2. Enter the zyBook code: WFUCSC112Spring2017
3. Click subscribe (There are two sections of the class. Be sure to sign up for instructor Santiago.)

The cost to subscribe should be $67. The subscription will be valid through 06/08/2017.

References: There are many online resources that you will likely find useful. One can be found at the link below. The zyBook also points to various online references as appropriate.

http://www.cplusplus.com/

I often find it useful to have a hardcopy reference. I like the book C++ How to Program by Deitel & Deitel. The current edition is the 10th edition, but earlier editions are fine and a lot cheaper. And there are other good texts.

Format
• **Students prepare before class** by reading assigned material and working through the exercises and problems in the zyBook. Students should prepare questions for the next class as needed.
• Pre-reading and zyBook exercises in the text are part of the grade and are due by the beginning of class. The zyBook keeps track of how much students have done and how well. The instructor will download this information at the start of class, and incorporate appropriately into the final grade. The instructor will answer questions, emphasize important points, and work through exercises.
• Students (individually or in small groups) will work on exercises during the remaining time. The instructor will work with students as a class, in smaller groups, or individually to assist in learning.
• Students or groups may be called upon to explain their solutions and to discuss problems and alternatives.

Assignments/Labs
• Generally, assignments are a combination of lab and take-home and usually given each week. Assignments are both learning exercises and opportunities to demonstrate that you have indeed learned. Assignments take time and effort and, while careful problem analysis and planning are critical, are often a trial and error exercise. Learning by making mistakes is important. You will struggle at times. Learning to recognize, trace, and correct problems is a key outcome of this course. This is called debugging. It is a skill (and an art at times) that is best, perhaps only, learned by doing – especially trial and error. Work through problems thoughtfully and persistently. The instructor and TA are there to help by guiding you to the solution, not to immediately find one. **Learn by doing – through success and failure.**
• Students will use a server (williams) for programming using clion. You must be on the WFU network to access williams. If not on campus, you will have to use virtual private network (vpn) to access the network.
• Students should use MobaXterm for access to the server. Download instructions are in Sakai.
• **CAVEAT!** When programming using clion on the server (williams), you MUST save your work and close clion BEFORE closing your laptop. If you do not, your work will be LOST!
- **CAVEAT!** Back up your work. Copy, whatever it takes. Make versions of your projects. Whatever it takes.
- Labs and other programming assignments will follow the coding guidelines at the following link: [http://csweb.cs.wfu.edu/~turketwh/CSC112/style/styleGuide.html](http://csweb.cs.wfu.edu/~turketwh/CSC112/style/styleGuide.html)
- Tests (2) will cover cumulative learning and will typically require a full class period to complete.
- The final exam will be cumulative, will be given during the assigned exam period for the class, Tuesday, May 2nd, at 2:00pm and may take the full three hours to complete. Please take note of this date before you make airline reservations or similar to leave campus for the holidays. If you have a conflict, you should know what it is on the first day of class. If you have a real conflict that may require rescheduling the exam for this class, you **must** discuss with the instructor no later than the end of the third class meeting.

### Assessment and Grading

Assessment allows students, and the instructor, to assessment progress and understanding. The zyBook is good for formative assessment as students have ample opportunity to rework exercises prior to final submission. Grades are simply a fact of life but should not be the goal of the student. Work hard, learn, have fun – the grade you earn will take care of itself. Quizzes are based on the lab with which they are associated. Tests and exam are cumulative but normally are weighted on material since the previous test.

- ✓ Assignments/Labs 24%
- ✓ Assignments/Labs Quizzes 6%
- ✓ Textbook activities and Unix Tutorial 5%
- ✓ Tests: 2 @ 20% each 40%
- ✓ Exam 25%

Final grade assignment will adhere minimally to table below. That is, achieving the number grade will ensure that the associated letter grade will be at least the one shown with the following possible exceptions: In order to pass, students must: Complete and submit all assignments/labs showing a reasonable effort. As defined by the instructor, a reasonable effort ordinarily means a minimum grade of 50% on each assignment. Failure to meet this requirement, may result in an F or significant grade reduction for the course.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>93 or above: A</td>
<td>73 thru 76.99: C</td>
</tr>
<tr>
<td>90 thru 92.99: A–</td>
<td>70 thru 72.99: C–</td>
</tr>
<tr>
<td>87 thru 89.99: B+</td>
<td>67 thru 69.99: D+</td>
</tr>
<tr>
<td>83 thru 86.99: B</td>
<td>63 thru 66.99: D</td>
</tr>
<tr>
<td>80 thru 82.99: B–</td>
<td>60 thru 62.99: D–</td>
</tr>
<tr>
<td>77 thru 79.99: C+</td>
<td>below 60: F</td>
</tr>
</tbody>
</table>

### Course Policies

- ✓ Familiarity with the syllabus and the Sakai site is required.
- ✓ Preparation prior to class is required.
- ✓ Attendance for and participation in all class periods is required. If work is completed prior to the end of the period, students may request to be excused if the instructor is satisfied with the submission and other progress in the course.
- ✓ Students must bring laptops to each class meeting to be used when specified as part of the class. Other electronic devices (phones, calculators, cameras, recording devices, etc.) are not to be used.
- ✓ No devices other than pencils are allowed during tests. Paper is provided.
✓ Announcements and other course communications will be made frequently using Sakai and email. Students are responsible for monitoring these forums regularly.

✓ Assignments are due on the due date (duh – that’s called the “due” date). If students are unable to turn in an assignment when it is due, please make arrangements to submit it before the due date. Late submissions have consequences (usually negative ones). Turn in what you have, even if only partially complete.

✓ Medically related absences are excusable, but if possible please email the instructor if unable to attend class. The instructor retains the right to require documentation justifying absences.

✓ Other absences may be requested, but please provide a schedule as soon as possible if you have activities that will require you to miss class or due dates. Examples include clubs, sports, etc.

✓ Make-up tests/exam may be arranged in the case of unavoidable circumstances such as illness or official university events/activities, etc.

✓ Unexcused absence from a test or exam will result in a zero for that grade. Examples of unexcused absence from past classes include, “I slept through 5 alarm clocks.” Kidnapped by extra-terrestrials may be allowed with proper documentation.

✓ Students with limitations or other conditions that may require an accommodation should contact the Learning Assistance Center as soon as possible within the first two weeks of the semester and bring this to the attention of the instructor as appropriate.

✓ Wake Forest University subscribes to an honor system. The instructor assumes student familiarity with and adherence to the honor system.

✓ Emergency preparedness policy: In the unlikely event of a major disruption of normal university activities (such as might result from a health emergency or other disaster), a course continuation contingency plan will be enacted in order to allow completion of the course. During this time, students should continue with the reading and other assignments listed on the syllabus and monitor email, Sakai, and the WFU website for information. If students have questions or are in doubt about how to proceed, they should contact the instructor by email if available, otherwise they should contact by phone. Carrier pigeon and Pony Express services have been discontinued.
Course Topics
The following course topics, which include review from CSC 111, are prescribed by the faculty of the Computer Science Department. The order of these topics covered in CSC 112 may not match the order presented below.

- **Problem Solving**
  - Review from CSC 111
    - The concept of an algorithm
    - Algorithmic problem-solving and problem solving strategies
    - Structured decomposition and top-down design
    - Debugging strategies
    - Unit testing
  - New in CSC 112
    - Searching and sorting algorithms

- **Fundamental Constructs**
  - Review from CSC 111
    - Syntax and semantics of a higher-level language, C++
    - Variables, types, expressions, and assignment statements
    - Simple input and output
    - Conditional and iterative control structures
    - Recursion
    - Functions and parameter passing
    - Call by value vs. call by reference
  - New in CSC 112
    - Operator overloading

- **Data Representation, Data Types, and Data Structures**
  - Review from CSC 111
    - Binary and hexadecimal base systems
    - Representation of primitive types such as character, integer, and floating point numbers
    - Arrays (including multi-dimensional ones), array lists, or vectors, as appropriate
    - Strings and string processing
    - References
  - New in CSC 112
    - Type conversion and coercion
    - Scope
    - Namespaces
    - Template classes
    - Pointers
    - C-strings
    - Linked lists
    - Introduction to stacks, and queues
    - Bit operations

- **Procedural and Object Oriented Programming**
  - Review from CSC 111
    - Procedural program layout
    - Object-oriented design
    - Classes, objects, and inheritance
- Encapsulation, data protection, and information-hiding
- Separation of interface and implementation
  - New in CSC 112
    - Friends
    - Constructors and copy constructors
    - Polymorphism

- **Other Topics**
  - Review from CSC 111
    - Types of errors: syntax vs logical vs run-time
    - Exception handling
    - Memory management (e.g., garbage collection, run-time storage management)
  - New in CSC 112
    - File input/output
    - Command-line arguments
    - Compiler directives
    - Multiple file compilation
    - Introduction to computational complexity
    - Run-time performance of a program

- **Tools and Skills**
  - Review from CSC 111
    - How to use an IDE effectively, including the debugger
    - How to use libraries and APIs, including their documentation
    - File structure fundamentals (directories and subdirectories)
  - New in CSC 112
    - Working in the Linux environment, including basic Linux commands such as mkdir, cd, ls, cp, mv, grep, chmod, etc.
    - Compiling, running, and debugging a program from the Linux command line
    - Makefiles
    - Introduction to network communication, file transfer, and shells in Linux (e.g., ftp and ssh)