CS A250 (CRN 20690-25399)

C++ Programming Language 2

Syllabus – Fall 2015

Instructor: Gabriela Ernsberger

E-mail: gernsberger@occ.cccd.edu

Course Web site: www.thisclass.info

Office: MB Computing Center 116
Office Hours: **MW** 1:45-2:20 pm / **MW** 4:55-5:15 pm / **TTh** 4:55-6:00 pm

Classroom: MB Computing Center 123

Hours:

CRN 25399	**MW** 11:10am - 01:45pm
CRN 20690	**TR** 06:00pm – 08:35pm

Course Description

Course Description. Second course in ANSI/ISO Standard C++ programming language. Topics include sorting and searching, data structures, operator overloading, memory management, exception handling, name scope management, polymorphism, templates, STL containers, STL algorithm and iterators, and functional programming.

Prerequisites. Before enrolling in this class, you need to **complete CS A150**, C++ Programming Language 1, with a grade of C or better.

When taking this course, **you are EXPECTED to already know how to create C++ applications that include the following**:

- Control structures
- Functions that pass parameters by value and by reference
- Classes
- OOP and separate compilation
- Dynamic variables and arrays
- Pointers
- Recursion
- Inheritance

Transfer Credit. CSU; UC.

Course Objectives. The course objectives for this course are as follows:

1. Compare performance of various sort and search techniques.
2. Create class templates and function templates that show the extensibility of the language.
3. Create programs that use the container classes in the Standard Template Library (STL).
4. Apply memory management techniques to solve problems.
5. Differentiate between the assignment operator and the copy constructor.
6. Produce programs that use the concept of overloaded functions and operators with respect to classes.
7. Implement and manipulate singly- and doubly-linked lists.
8. Produce programs that show the understanding of polymorphism as it applies to C++.
9. Apply the concept of abstract classes to enhance their understanding of OOP concepts.
10. Produce programs that use the exception handling features of the language.
11. Produce programs that show an understanding and appreciation for iterators and algorithms in STL.
12. Produce programs that show an introductory understanding of functional programming.

Student Learning Outcomes. Upon successful completion of this course, the student should be able to:

- Use C++ to solve problems that incorporate the use of class and function templates, linked lists, and overloaded operators.
- Use a functional programming language to solve problems by implementing lambda expressions.

Textbooks and Software

Textbook. There is no specific textbook for this course. Lectures, slides, exercises, and projects make up the material you need to prepare for the course. If you feel you need to also have a textbook, then there are a few suggestions for you listed below. (Note that the edition does not really matter).

- **Absolute C++**
  Savitch
- **C++ How to Program**
  Deitel
- **Big C++**
  Horstmann
- **Objects, Abstraction, Data Structures and Design: Using C++**
  Koffman, Wolfgang

Keep in mind that some authors use different versions of the algorithms and code syntax. You will be responsible for material presented in class, NOT for material learnt from any of the suggested books.

Software. We will be using **Visual Studio 2013** to complete all programming exercises and exams. The software is installed on most of the computers in the MB Computing Center. If you would like to own the software, you can do the following:

- **Visual Studio 2013** – This is the full edition. Ask the front desk at the Computing Center how to get a copy.
- **Visual Studio Express 2013 for Windows Desktop** – This is a smaller version of Visual Studio. You will need to register to obtain a registration key (registration is free).

Student Evaluation

To be **successful** in this course you need to study the assigned material from the slides and code examples, work on the exercises and projects, attend class, and do well on the exams and final assessment.

IMPORTANT: No matter what your grade is at the end of the semester,

if you do NOT take the final exam, you will NOT pass the class.

Exercises. Some of the exercises will be assigned as **homework** and some will be assigned as **class work**. You will get credit for turning in **80%** of the exercises; the 20% margin will take care of a few exercises you might miss.

- **In-class exercises**
  - These will be randomly assigned during class and must be completed by the end of the class period, unless stated otherwise. Some of these exercises will be on paper; therefore, **make sure you bring paper and pencil to class.**
- **At-home exercises**
  - All at-home exercises need to be turned in on the due date before class starts.

Projects. There will be **two (2)** programming projects that you can complete by yourself or as a group with other students in the class—a **maximum of three (3) students in a group** is allowed.

Programming Exams. You will be required to write C++ code. The number of the exams will be determined during the semester, but the total scores for all exams will remain fixed to 70 pts.

Midterm and Final. Both midterm and final exams are on paper (make sure you bring a pen/pencil). Type of questions will be: multiple-choice, true/false, short answer, find the output, find the error, and write missing statements for a given code segment. The **final exam will be comprehensive**. If you miss the midterm exam, you will be assigned the grade you received on the final exam. If your final exam score is greater than your midterm exam score, the final exam score will replace the midterm score.

Assessment. This is a short quiz that will be given on the last day of the semester, which will include ***only*** questions related to the SLO's.

IMPORTANT: There is **NO make-up** for **any** of the exams

(programming exams, midterm, final, assessment).

Programming Exercises. These programming exercises will help you prepare for the programming exams. Although solutions to some of these exercises will be provided, it is important that you try and do them yourself because all programming exams will be based on these exercises.

IMPORTANT: It is **YOUR responsibility** to look at the solutions given and compare them to your implementation and ask me questions if you are in doubt.

Grading. All grades will be **posted on the class Web site**. On the third week of class, you will be getting instructions on how to access your grades. You should check your grades regularly and notify me immediately if there are any inconsistencies.

Grades will be calculated as follows:

	Points
80% of exercises turned in	10
Projects	20
Programming Exams	70
Midterm	40
Final	40
Assessment	10
Total	**190**

A >= 171

B >= 152

C >= 133

D >= 95

The CCFaculty Q Drive

The **CCFaculty Q** drive is where you can find material presented during the week and where you can turn your projects.

You can access the **Q drive** only from the **Computing Center** using any computer. To access the **Q drive**:

1. Go to **Start | Computer**
2. Select **ccfaculty on 'occsdfs' (Q:)**
3. Select **faculty2 | gernsberger**

Submission folder. This is the folder where you can drop your project. Drop the files in the folder that has your course name.

IMPORTANT: Before dropping a **folder**, you need to compress it as a **zip** (**NOT** .rar); otherwise the files inside your folder will not transfer.

A **simple file** does **NOT** need to be compressed.

NOTE: You **may not be able to** retrieve and/or modify files that are stored in the **submission folder**. If you need to make changes to your project, simply re-submit by naming your file as indicated in the homework instructions and add the **suffix _S2**. For each subsequent submission, increase the submission number to **_S3**, **_S4**, etc.

Files folder. This is where you can find the **week's course material**. Files will be stored in there until the beginning of the following week, *after* they have been posted on the class Web site.

Saving Your Work

As a programmer, you should already know how important it is to back up your work regularly. You should always have at least two copies of your work, saved in two different places. Accidentally deleting or losing your work is **not** an excuse for not turning in your assignments.

These are a few options on **how to save your work**:

- Bring a flash drive
- Upload it to Google Drive via your student account
- Send an email to yourself with your work as an attachment, **BUT** do not include the .exe file, because your email server will recognize it as an application and possibly discard the file.

Programming Standards

Name Headers. You are required to write a name header on ***all*** projects and programming exams. The header is placed at the beginning of the main file, where the **main()** function is implemented, unless otherwise specified. Failure to do so in any of the programming exams will cost you **1 point**.

The format required for the name header is shown below:

```
/*
  Lastname, Firstname    // If you are working as a group, write all names
  CS A250
  Month (day), (year)    // Month must be written in letters
  (leave one line)
  Project #/Exam #
*/
```

IMPORTANT: This is about **paying attention to detail**, which means that you are expected to write your name header as shown here to avoid losing 1 point.

Naming Standards. You will be testing other students' files, and for this reason it is imperative that you use the same standards for **identifiers**.

Use **camelCase** for:

- Variables `var, myVariable`
- Objects `obj, myObject`
- Functions `func, myFunction`

Use all **CAPITAL_LETTERS** separated by an **underscore** (_) for:

- Constants `CONSTANT, MY_CONSTANT`

Use **PascalCase** (this is also called **UpperCamelCase**) for:

- Classes `Class, MyClass`

Academic Honesty

Although I encourage you to discuss your work with other students and look at their code as well, you are **NOT** allowed to copy code from someone else's implementation **AND/OR** to exchange files. At the end of the semester, I will be **testing your code** with **MOSS** (**Measure of Software Similarity**), which is a software used to detect plagiarism. If I find two or more files that are the same, **you will automatically get an F in the class and you will be reported to the Dean of Students**.

RULES for all exams:

- No cell phones
- No flash drives (or any type of external memory)
- No notes, no books
- No calculators
- No watches
- No hats
- No ear phones
- No talking to other students
- Seating will be assigned
- Bring a pen/pencil (I will provide paper)

The **exercises** are designed to prepare you for the programming exams. It is to your advantage to work on them on your own. Nevertheless, I suggest you talk to other students and compare ideas with one another. Again, this does NOT meaning exchanging files.

Attendance

Attendance is mandatory. Class starts at the time shown on the schedule, but **lecture starts 10 minutes after**. This is to give you time to log in to a computer and be ready for lecture. I will take attendance 10 minutes after class starts either by calling your names or by relying on the classroom's network software—when you log in to any of the class computers, your ID shows on the network and marks you as present.

Note: Do **NOT** disconnect yourself from the network, or you will be **marked absent**.

A few **rules** about attendance:

- If you **miss any class meetings** on the **first two weeks OR** if you **miss more than three (3) class meetings** in the **following weeks** you **may be dropped without notice**; do **not** assume, however, that you will be automatically dropped if you fail to come to class. It is your responsibility to withdraw from class and to check the deadlines on the OCC Web site.
- If you arrive to class **AFTER I took roll, you will be considered absent**.
- If you **fall asleep** during class hours, you **will be asked to leave the room**.
- Note that students **cannot** be dropped *after* the stated deadline for the last day to drop with a W.

Communication

The best way to communicate with me is through **email** AND/OR **coming to my office during office hours**. **I read and reply to emails from Monday through Friday early in the morning**.

Email. You need to set up your **OCC student email account** to receive my emails. You can access your email account through **MyOCC**. You may send me emails from your personal account **as long as you write in the subject "CS A250"**. Replies will be sent to the sending address.

Announcements. I will email you any last-minute announcements to your **OCC student email address** (***not*** your personal email address), so do **check your email account daily**.

Forwarding Your Student Email. Instructions on how to forward your student email to your personal account can be found on the OCC Web site.

Other Course Information

Incomplete grade. I do **not** give incomplete grades to anyone. If you are not able to take the final exam, you will not pass the class, but you can retake the class at a later time and overwrite your grade with a passing grade.

Disruptive Behavior. I expect all of you to be polite, respectful, and helpful to your fellow students**, and to leave your sitting station neat and clean**. If, in my judgment, your behavior negatively impacts the rest of the class, you may be subject to disciplinary action.

Disabilities. If you have a disability that may impede your ability to successfully complete this course, you should contact the Disabled Student Center (432-5807 or 432-5604 TDD) no later than the first week of the course. Their staff will assist you in arranging accommodations that can help you meet course requirements.

Reservation of Rights. I reserve the right to change this syllabus without limitation and without prior notice. If I do modify any item or policy, I will notify you by sending you an email to your OCC email account.