1 Basic Course Information

1.1 When and Where

This course is online and has no regularly scheduled lectures.

There will be an optional Orientation session conducted via network conferencing at the start of the semester. Additional similar sessions may be announced later in the semester.

1.2 Objectives

This course will explore the techniques of object-oriented programming, analysis, and design. The emphasis will be upon the development of clean interfaces that permit easy modification and reuse of software components. Other techniques, drawn from outside the object-oriented approach, that significantly contribute to this goal will also be discussed.

Students will gain facility in an object-oriented programming language and will learn the constructs that differentiate such languages from others. This course will explore the idioms and styles of object-oriented programming in C++ and
Java, with emphasis upon how these contribute to reusable software components.

Students will learn how to use object-oriented techniques in support of programming. In particular, students will be introduced to the process of object-oriented analysis as a means of understanding an unfamiliar problem domain. Students will learn to build and use models, expressed in the Unified Modeling Language (UML) to codify and evaluate that understanding and to evolve system requirements. Students will learn how to use those models to facilitate a smooth transition from analysis to design and from there to implementation.

1.3 Textbooks

Most assigned reading will be from books and other sources available on-line.

Required:

- Any reasonable C++ text that covers classes and inheritance (e.g., your CS250 or 333 textbook)

2 Communications

2.1 Instructor

<table>
<thead>
<tr>
<th>Steven Zeil</th>
<th>ECSB 3208</th>
</tr>
</thead>
<tbody>
<tr>
<td>(757) 683-4928</td>
<td><a href="mailto:zeil@cs.odu.edu">zeil@cs.odu.edu</a></td>
</tr>
</tbody>
</table>

Important: All email related to this course should have the phrase “cs330” somewhere in the subject line. This flags your message in my mailbox for faster attention. Omit this, and your message may get lost amid the ton of daily spam and ODU administrative messages I get each day.

I try to respond to all (properly marked) messages within 24 hours M-F, within 48 on weekends & holidays.

2.1.1 Office Hours

Students may meet with me in person, by telephone, or via internet-conferencing. A week-by-week schedule of available meeting times can be found by going to the my home page https://www.cs.odu.edu/~zeil/ and clicking on “Office Hours and Appointments”.

General questions about course content and reports of website problems can be asked in the Forums on Blackboard. Questions about grades, how to solve assignments and other graded activities should be send to zeil@cs.odu.edu.

For more discussion on course communications, please refer to the Communications policy.

3 Course Prerequisites

The prerequisites for this course are:

- [CS 250], Problem Solving and Programming II, or CS 333 Problem Solving and Programming in C++
- CS 252, Introduction to Unix for Programmers

Note that, if it has been some time since you took CS 250 or 333, or if you received weak grades in them, then you may need to do some review work to prepare for this class.

If you are a transfer student who took equivalent courses elsewhere, you would do well to review the material on those course websites and look for topics that may not have been covered in your prior courses, because course “equivalence”
is often a very rough approximation. Pay particular attention to the material on design, testing, and debugging, as these are often given short shrift at other institutions.

Either way, if you need to review any of the prerequisite topics described below, the time to do so is early in the semester, before you need it to understand the lectures or are called to use it in assignments.

Students who have successfully completed those prerequisites should have acquired the following knowledge and skills:

### 3.1 General Programming Knowledge

Students should be familiar with certain basic programming techniques that are largely independent of any specific programming language:

- using editors, compilers and other basic software development tools.
- basic software design (i.e., stepwise refinement and top-down design)
- software testing, including the use of scaffolding code (stubs and drivers), selection of test cases for black-box testing, and head to head testing
- debugging, including the use of debugging output, the use of automatic debuggers to set breakpoints and trace program execution, and the general process of reasoning backwards from failure locations to the faulty code responsible for the failure.

### 3.2 C++

I will assume that you are familiar with the basics of C++, including

- the various C++ statements and control-flow constructs,
- the built-in data types,
- the use of arrays, pointers, pointers to arrays, and linked lists,
- the use and writing of functions, and
- the basic use of structs and classes for implementing abstract data types.

### 3.3 Java

No prior knowledge of Java is assumed.

- In general, CS students at the 300 level should be able to pick up new programming languages with only moderate effort. Because Java is closely related to C++, this is particularly true of students moving from C++ to Java.

- If you believe that you need a more structured aid to learning Java, the 1-credit course CS382 presents the basics of the language in a self-paced form. In fact, much of the course material for CS382 will be listed as required reading for this course.

  - When the CS 330 Outline page lists a section of 382 for readings, students are expected to both read the 382 lecture notes and to do the associated 382 labs. You will not be expected to do the 382 assignments or to take the 382 exam.

You may, if you wish, choose to register for CS382 and earn credit for it. In that case, you will be doing largely the same set of readings, but will then be obligated to do the assignments and exam for that course.

- CS 330 will pick up with Java topics more advanced than are currently covered in 382, including multi-threading and the development of GUIs and in general will explore how to use Java in a truly Object-Oriented style.

### 3.4 Unix
All students in the course will receive accounts on the CS Dept. Unix network, and knowledge of how to work with the Linux servers is part of the course prerequisites. This course does not require familiarity with shell scripting. All other topics in CS 252 are required.

Some assignments may require the use of software available only on the Linux servers. Others may require (or, at least, be simplified by) use of the X windowing system.

### 3.5 General Computer Literacy

You will be studying techniques in this course for preparing professional-quality software documentation. The key embedded word in “software documentation” is “document”. Students taking this course should be able to use word processors and other common tools to produce good quality documents, including mixing text and graphics in a natural and professional manner.

### 4 Assignments

Assignments for this course will include *programming assignments* (in C++ and Java), which must be done on an individual basis, and *design assignments*, which may be done in small teams.

#### 4.1 Computer Accounts

Students will need an account on the CS Dept. Unix network to participate in this class. This account is unrelated to any University-wide account you may have from the ODU’s Information Technology Services (ITS).

If you have had a CS Unix account in the recent past, you should find it still active with your login name, password, and files unchanged. If you have had an account and it has not been restored, contact the CS Dept systems staff at root@cs.odu.edu requesting that it be restored.

If you do not yet have such an account, go to the CS Dept. home page and look for “Account Creation” under “Online Services”. *All students in this course are responsible for making sure they have a working CS Unix account prior to the first assignment.*

#### 4.2 Compilers

The “official” environment in which students’ programming assignments will be evaluated is defined by our Dept. Linux servers. It is the student’s responsibility to be sure that their code compiles and executes using the compilers and run-time environment provided there. As of this writing, the compiler versions used are

- C++: g++ 5.5
- Java: 1.8.0_91

[Help is available](#) for students wishing to install compatible compilers and IDEs on their own PCs.

### 5 Course Policies

#### 5.1 Academic Honesty:

Everything turned in for grading in this course must be your own work. If an assignment is explicitly described as a team assignment, it must be the work of the team members only.
The instructor reserves the right to question a student orally or in writing and to use his evaluation of the student’s understanding of the assignment and of the submitted solution as evidence of cheating. Violations will be reported to the Office of Student Conduct & Academic Integrity for consideration for possible punitive action.

Students who contribute to violations by sharing their code/designs with others may be subject to the same penalties.

- Students are expected to use standard Unix protection mechanisms (chmod) to keep their assignments from being read by their classmates. Failure to do so will result in grade penalties, at the very least.

This policy is not intended to prevent students from providing legitimate assistance to one another. Students are encouraged to seek/provide one another aid in learning to use the operating system, in issues pertaining to the programming language, or to general issues relating to the course subject matter.

Students should avoid, however, explicit discussion of approaches to solving a particular programming assignment, and under no circumstances should students show one another their code for an ongoing assignment, nor discuss such code in detail.

Use of Online Resources

You may not post details of course assignments, projects, or tests at online Forums, Bulletin Boards, Homework sites, etc., soliciting help.

You may use information that you have not solicited but have located, subject to the following restrictions:

- Just as when writing a paper, if you use someone else’s ideas, you must cite your sources appropriately. Within code, such citations appear in comments.

Example:

```
double x = 23.0;
double xsqrt = sqrt(x);
// Search algorithm based upon code by S Zeil at
// https://www.cs.odu.edu/~zeil/cs361/latest/Public/functionAnalysis/index.html#orderedsequentialsearch
int loc = 0;
while (loc < arraySize && numbers[loc] < xsqrt) 
```

- Just as when writing a paper, if you use someone else’s words (code), you must cite your sources appropriately and mark the quoted text. Within code, such citations appear in comments.

Example:

```
double x = 23.0;
double xsqrt = sqrt(x);
// Begin quoted code from S Zeil at
// https://www.cs.odu.edu/~zeil/cs361/latest/Public/functionAnalysis/index.html#orderedsequentialsearch
int loc = 0;
while (loc < listLength && list[loc] < searchItem)
{
    ++loc;
}
// End quoted code
```

- Failure to appropriately cite any such “found code” will be taken as evidence of plagiarism.
The overall principle stated in the first sentence of this section remains in effect. “Everything turned in for grading in this course must be your own work.” If the bulk of your assignment, project, test answer, etc., are copied, even with appropriate citation, to the degree that, in the judgment of the instructor, you have not demonstrated your own knowledge of the course material, you will receive a zero for that submission.

5.2 Due Dates:

Assignments are marked with an explicit due date, and are due at the end of that day (11:59:59PM, ET). You will find these dates on the outline page and on the course Announcements page on Blackboard.

Late submissions for programming assignments (one in which the principal thing being submitted is source code) will be accepted, at a 5% per day penalty, up until the start of the final exam. Late submissions will not be accepted once the scheduled starting time of the final exam has begun. Late submissions of design or non-programming assignments, and exams are not accepted.

5.3 Grading:

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignments</td>
<td>40%</td>
</tr>
<tr>
<td>Midterm Exam</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>35%</td>
</tr>
</tbody>
</table>

I will drop one assignment grade or the midterm exam grade, whichever improves a student’s overall grade more.

For the truly curious, some further information on my approach to grading is available.

The midterm and final exam will be administered online via Blackboard.

5.4 Extensions, Exceptions, and Incomplete (I) Grades

Exceptions to these dates or to the grading policies of the course will be made only in situations of unusual and unforeseeable circumstances beyond the student’s control.

Reasons that are most likely to justify an exception include extended illness, military deployments, or job transfer/relocation, but you should be prepared to document these if requested.

The following are usually not valid reasons for an extension:

- “I forgot that I was signed up for this course.” or “I didn’t know what the Grading policy was.”
  
  This was not beyond your control.

- “I have a part-time (full-time) job.”
  
  This is not exceptional. Most of your classmates work, many of them full-time.

- “I have a heavy course-load this semester.”
  
  Neither exceptional nor outside your control.

- “I’ve worked so hard on this course.”
  
  I’m sure that your fellow students would not appreciate the implication that you believe they have been slacking off.
Fundamentally, though, this argument misses the whole point of grades in a course. They are not a reward for putting in time and effort. They are given, instead, to certify that you have demonstrated a certain level of knowledge and, in the case of a course like this that is a prerequisite to many other courses, to make sure that you are sufficiently prepared to succeed in those later courses.

- “I got stuck on assignment X and was never able to catch up.”

Actually, this might qualify, but only if you made good use of email and/or my office hours to resolve your problems with that assignment in a timely fashion. Your chances of getting an exception in this case will also depend upon just how far behind you are and on how long before you waited to discuss the matter with me. You are far more likely to get a short period of time to complete one assignment than to get any extra time at all to complete 50% of the course.

- "I had trouble completing some assignments and haven’t sent you an email or attended your office hours because I’m not the kind of person who likes to ask for help.

Then you shouldn’t be the kind of person who asks for exceptions either. A significant part of a college-level education is learning to exploit the information resources available to you. Deliberately refusing to do so is not a behavior that I’m inclined to reward.

Requests for an “I” grade or extended time to complete the course should be made before the actual end of the semester, whenever possible. Requests made after grades have been submitted will need to include an explanation of why the request was delayed.

6 Educational Accessibility

Old Dominion University is committed to ensuring equal access to all qualified students with disabilities in accordance with the Americans with Disabilities Act. The Office of Educational Accessibility (OEA) is the campus office that works with students who have disabilities to provide and/or arrange reasonable accommodations.

- If you experience a disability which will impact your ability to access any aspect of my class, please present me with an accommodation letter from OEA so that we can work together to ensure that appropriate accommodations are available to you.

- If you feel that you will experience barriers to your ability to learn and/or testing in my class but do not have an accommodation letter, please consider scheduling an appointment with OEA to determine if academic accommodations are necessary.

The Office of Educational Accessibility is located at 1021 Student Success Center and their phone number is (757) 683-4655. Additional information is available at the OEA website [http://www.odu.edu/educationalaccessibility/](http://www.odu.edu/educationalaccessibility/)

© 2015-2019, Old Dominion Univ.