Independent Study in Idaho

CS 120
Computer Science I

Providing independent study opportunities for more than 40 years.

The University of Idaho in statewide cooperation with
Boise State University — Idaho State University — Lewis-Clark State College

Course Guide

Independent Study in Idaho

Self-paced study. Anytime. Anywhere!

Computer Science 120

Computer Science I

University of Idaho
4 Semester-Hour Credits

Prepared by:
Terence Soule
Professor
University of Idaho

RV: 12/2015
2 – CS 120 (new revision #, 8-30-16; same content)

Table of Contents

CS 120 Computer Science I 4 Semester-Hour Credits: UI

Welcome!

Whether you are a new or returning student, welcome to the Independent Study in Idaho (ISI) program. Below, you will find information pertinent to your course including the course description, course materials, course objectives, as well as information about assignments and grading. If you have any questions or concerns, please contact the ISI office for clarification before beginning your course.

Policies and Procedures

Refer to the ISI website at **www.uidaho.edu/isi** and select *Students* for the most current policies and procedures, including information on setting up accounts, student confidentiality, exams, proctors, transcripts, course exchanges, refunds, academic integrity, library resources, and disability support and other services.

Course Description

Fundamental programming constructs, algorithms and problem-solving, fundamental data structures, overview of programming languages, virtual machines, introduction to language translation, declarations and types, abstraction mechanisms, object-oriented programming.

Prereq: Math 143 with a grade of 'C' or higher or CS 112 with a grade of 'C' or higher; or sufficiently high ACT, SAT, or Math Placement Test score to qualify for Math 170

15 graded assignments, 15 graded quizzes (auto-graded in Canvas), 3 proctored exams

Students may submit up to 1 assignment at a time/2 per week. Before taking exams, students MUST wait for grades and feedback on assignments, which may take up to three weeks after date of receipt by the instructor.

ALL assignments and exams must be submitted to receive a final grade for the course.

Course Materials

Required Course Materials:

Soule, Terence, A Project Based Introduction to C++, First Edition, KendallHunt, 2014. ISBN-13: 9781465260468, ISBN-10: 1465260463.

Supplementary Materials: Student will need access to a computer with a C++ compiler. Several free options exist. They can use free software to log into the CS department's computer and use the software available there. Alternatively, Windows users can install either Cygwin or minGW. Apple computers come with a compiler pre-installed.

Course Delivery

All ISI courses are delivered through Canvas, an online management system that hosts the course lessons and assignments and other items that are essential to the course. Upon registration, the student will receive a *Registration Confirmation Email* with information on how to access ISI courses online.

Course Introduction

This course is an introduction to programming using the C++ programming language. It is suitable for students with little or no programming experience. The course focuses on hands-on activities that involve reading, analyzing, modifying, and writing code. Because programming is a skill requiring practice the course does require fairly extensive time programming at a computer.

Course Objectives

The objective of this course is to give the student a foundation in programming and programming concepts and a working knowledge of the C and C++ programming languages. At the end of the course students should be able to write fairly sophisticated programs, including defining and using classes and objects, using arrays, and using pointers.

The major topics covered in the course are:

1. Intro to computers and programming
2. Basic program structure, variables, I/O
3. Types: internal representation, base conversion
4. Arithmetic expressions, operators, computer arithmetic
5. Relational operators, conditional statements (if, if-else, switch)
6. Iteration, looping, loop techniques
7. Functions, parameter passing, return values
8. Files, file I/O
9. Arrays, 1-D, multidimensional
10. Character strings (C-style and C++ string class)
11. Classes
12. Social and Ethical Issues
13. Pointers with Arrays, dynamic memory
14. Recursion

Course Structure

The course is divided into 15 “lectures”. There are two lectures for each chapter, except the first, introductory, chapter, which only has one lecture associated with it. There is one assignment and one short quiz for each lecture. There are two midterms and one final exam. All of the exams are comprehensive, covering all of the material in the course up to the point of the exam.

Assignments typically consist of one or more programming projects. Many of the assignments require starting with sample programs from text and making multiple additions and/or changes. For these assignments the idea is to make a better program. You should make one change at a time. It’s a good idea to save the program with a new name after completing each successful change so that if you make a mistake and “break” the program you can go back to an older version.

Some assignments will include a programming problem from a previous chapter as a review.

Exams containing a mixture of questions types, including: short answers, matching, multiple choice, true/false, analyzing code, and writing short pieces of code.

Grading

There are 15 assignments, 15 short quizzes, and three exams in the course. The percentage of the overall grade assigned to each of these is:
Assignments: 20%
Quizzes: 15%
Exam 1: 20%
Exam 2: 20%
Final exam: 25%

About the Instructor

Dr. Terence Soule is a Professor in the Computer Science Department at the University of Idaho. He has been teaching computer science courses for over 15 years, ranging from introductory programming courses to graduate level courses in machine learning and artificial intelligence. His own passion for programming started in high school and he works on a range of programming projects including artificial life simulations and evolutionary computation based learning algorithms.

Instructor Contact Information

Dr. Terence Soule can be reached via email at tsoule@cs.uidaho.edu
He can also be reached via phone at 208-885-7789 during his office hours which are on-line at www2.cs.uiaho.edu/~tsoule

Lesson 1
Introduction to Programming

Objectives

Become familiar with the programming environment that the student will be using.

Understand the basics actions of a compiler and the difference between a high level programming language and an executable.

Be able to create, compile, and run a simple C++ program.

Students using the CS department's computer system should become familiar with the use of a secure shell program and the basics of the Unix operating system.

Reading Assignment

Chapter 1

Important Terms

Compiler, Unix, high level language, executable, library, main(), cout, <<, {}

Lecture

C++ (and C) is a high-level programming language, meaning that it is fairly easy for a human to understand it and to write code using it. However, computers actually run machine executable code (in an "executable" file), which is very difficult for humans to read and write. A compiler bridges this gap, turning (correct) high-level code into an executable file that the computer can run.

Thus, the first step is to become familiar with the use of a C++ compiler. If you have already done some programming with a C++ compiler it is fine to continue to use it. If not there are several options:

1) (Windows users) Download putty from the University of Idaho's ITS department (http://www.uidaho.edu/its/software) and use it to log into the CS departments Unix computer at wormulon.cs.uidaho.edu. This computer has a C++ compiler (called gcc) that you can use.
2) (Apple users) Open a terminal window and use the command:
   ssh username@wormulon.cs.uidaho.edu to log into the CS departments Unix computer at wormulon.cs.uidaho.edu. This computer has a C++ compiler (called gcc) that you can use.
3) Download an integrated development environment (IDE) such as Netbeans (https://netbeans.org/).

If you are using the CS department's computer (i.e. you logged into wormulon) then familiarize yourself with the basic of Unix and nano using the simple manual available here:
http://www2.cs.uidaho.edu/~cs120/f15/soule/Manual.pdf
Note that nano is a very basic text editor program that can be used to write C++ programs (or any other kind of text file).

Once you have access to a compiler enter, compile, and run the following "Hello, world" program. Begin by creating a new program. If you are using nano, simply type: *nano helloworld.cpp*

This will start nano and create a new file called helloworld.cpp. The .cpp extension indicates that this is a C++ program.

If you are using a different IDE (such as netbeans) go ahead and start a new project.
One you have started nano (or a new project in another IDE) enter the following program:

```
#include<iostream>
using namespace std;
int main(){
    cout << "Hello, world." << endl;
}
```

Pay careful attention to all of the "odd" symbols.
To compile this program on wormulon, type control^o to save the program from nano, and type control^x to exit nano. Now type:

```
g++ helloworld.cpp
```

to compile the program. If you get any errors you will need to use nano to fix the program (type nano helloworld.cpp again to reedit the file). If there are no errors the compiler will create a file called a.out.
Type:

```
./a.out
```

to run the program. You should see the "Hello, world." message.

This is the basic procedure to create, compile, and run a program:

1) Use an editor (such as nano) or an IDE (such as Netbeans) to create a C++ program.
2) Use a compiler (such as g++) to compile the program into an executable file.
    a. Use the editor to fix and errors in the program if necessary (possibly several times)
3) Run the executable file.

Written Assignment

For this assignment you will be learning the basics of the Unix operating system environment and how to compile and run a simple C++ program.

1) Follow the directions from lecture to log into wormulon.
2) Use the mkdir to make a new directory called *assignments*.
3) Use the cd command to switch to the new directory.
4) Follow the directions from the lecture to create, compile, and run the Hello, World program.
5) Once the program works, change the message so that the program says Hello, *Yourname*.
6) Add another line of output to the program and compile and run it.
7) Use the script command (described in the Unix tutorial) to capture the output of the new program.
8) Email the program and the output to the course instructor.

Exam 1 Information

Make arrangements with your proctor to schedule Exam 1.

Prior to taking this exam:

- You must submit assignments 1-7 to your instructor before taking this exam.
- Please do not take this exam until you have received graded assignments 1-7 back from your instructor.
- Do not submit any subsequent assignments until you have taken this exam.

Exam Components:

- This exam covers lessons 1-7.
- This is a closed-book, closed-notes exam.
- This exam is worth 100 points.
- Time limit: 1 hour

Items to take to the Exam:

- Photo identification
- V number
- Pen, pencil, eraser

Additional Exam information:

- Graded exams will not be returned to you.