CSC 120 and 120L (Section A)
Principles of Computer Science I
Fall 2004

Syllabus and General Information

Class Meetings

Lectures: Monday / Wednesday / Friday, 10:00 AM – 10:50 AM, Room SCI 107
Labs: Thursday, 2:45 PM – 4:45 PM, Room SCI 228

Instructor

Prof. Nadeem Abdul Hamid *Phone:* (706) 368-5632 (office)
Office: SCI 354B *Email:* nadeem@acm.org
Office Hours: Mon 11-12:30, 2:30-4 • Tue 9-11, 2:30-4 • Wed 11-12:30 • Thu 9-11 • *(or by appt)*

Course Catalog Description

CSC 120 Principles of Computer Science I **3-2-4**

An introduction to the fundamental principles of computer science. Emphasis on algorithms and computational problem solving, fundamental programming constructs, data representation and storage, language translation, software development methodologies, operating systems, networks, and social contexts. *Prerequisites: None.*

Overview

This is the first course in the Computer Science (CS) academic program. The fundamental question permeating all of Computer Science is, "What can be (efficiently) automated?" Throughout this course we will introduce the tools and techniques used in the discipline of computing. Aspects of design, theory, and experimentation will be addressed. An overview of central issues and concepts of the field will be offered, including software development, specifications, testing, data organization, machine architecture, history of computing, professional organizations, computer security, and computing ethics.

One of the primary goals of this class will be to learn programming. For this we will be using the Java programming language. As the best way to learn programming is to write programs, much of the student's time will be spent on this particular activity. During lectures, we will also spend a significant amount of time reviewing and studying Java source code examples.

Course Objectives

- To explain the purpose of the basic parts of a computer system, and to demonstrate the operation of the fetch-execute cycle using those parts;
- Enumerate the primitive data types and to convert such data between their everyday, "human" representation and their machine representations;
- Demonstrate fundamental algorithms for the manipulation of primitive data types;
- Apply Boolean logic to construct and evaluate simple combinational circuits;

- Relate the fundamental tasks of an operating system;
- Explain the fundamental structure and operation of computer networks;
- Apply basic design methodologies to realize solutions to computational problems;
- Explain the program translation process, and to distinguish aspects such as: compilers *vs.* interpreters, machine language *vs.* high level language, object code *vs.* executable code, and loading *vs.* linking;
- To obtain a solid understanding of the basic features of the Java programming language: object-oriented programming (classes, methods, objects), control structures (conditionals, looping, recursion), data types (arrays, strings, numbers), basic algorithms, and some advanced ideas;
- To use an integrated development environment (IDE) to translate and execute Java programs;
- To gain fundamental problem solving skills, applicable to all disciplines;
- To obtain a general understanding of the various topics in the broad context of computing;
- To develop an appreciation of the history of computing, as well as legal and ethical dimensions.

Keys to Success:

- Start early; work steadily; don't fall behind.
- You can't cram, unlike most other courses.
- If you get stuck, take a break and then go back to think about it.
- Don't hesitate to contact the instructor if you have any problems.

Expected Outcomes

The student will meet the objectives with at least 70% success, based on performance on exams, labs, and assignments.

Methods of Instruction

Three lectures and one lab session per week. Please see *Materials & Resources* below for website and computer hardware and software information.

Materials & Resources

Required Textbooks:

- Nell Dale, Chip Weems, and Mark Headington, *Programming and Problem Solving with Java*, Jones and Bartlett Publishers, 2003. ISBN# 0-7637-0490-3. (Book support website: *http://computerscience.jbpub.com/ppsjava*)
- (Lab manual) Nell Dale, *A Laboratory Course for Programming with Java*, Jones and Bartlett Publishers, 2003. ISBN# 0-7637-2463-7.

Online course website:

- *https://vikingweb.berry.edu* - It is your responsibility to check the Viking Web site for this course regularly (*i.e.* daily) throughout the semester, as it will be regularly updated with announcements, lecture notes, assignments, *etc.* Laboratory documents and announcements will be posted here also. Instructions for accessing this material will be provided.

Other suggested readings:

- Official Java API Documentation – *http://java.sun.com/docs/*

- The Java Tutorial – *http://java.sun.com/docs/books/tutorial/* – A practical guide for programmers, also available in print: Mary Campione, Kathy Walrath, and Alison Huml, *The Java(TM) Tutorial: A Short Course on the Basics*, 3rd Edition, Addison-Wesley Pub Co, 2000. ISBN# 0201703939.

Computer hardware and compilers: The machines in Berry's computer labs will have some Java development tools installed on them. We will become familiar with the development software within the first few lab sessions of this course. Students who wish to set up a Java programming environment on their own computers may consider downloading and installing the following:

- Sun Java SDK – *http://java.sun.com/j2se* – the main **s**oftware **d**evelopment **k**it for compiling and running Java programs.
- SciTE (**Sci**ntilla **T**ext **E**ditor) – *http://www.scintilla.org/SciTE.html* – a no-nonsense editor for writing programs with simple facilities for compiling and running them.

There are many other options available for Java development environments. Please consult with the instructor if you have any questions in this regard or need help in any way.

Assignments and Grading

Student grades will be determined on a standard 10% grade scale: 90% - 100% earns an A, 80% - 89% earns a B, *etc.*, with the instructor reserving the right to apply + / - grades at his discretion. Grades will be based on the weighted average of the following course work:

Participation (10%) – Attendance and participation in class will be taken into consideration as well as in-class exercises and / or occasional (possibly unannounced) quizzes. (See *Attendance Policy* below.)

Assignments (40%) – There will be a regular series of weekly or biweekly assignments throughout the course. For the most part these will consist of developing computer programs in Java, but they may occasionally be in the form of short written questions reinforcing the material in the lectures, readings, and labs.

Labs (15%) – Each 2-hour laboratory session will involve a variety of problems, including paper-and-pencil exercises, writing computer program code, modifying program code, and / or running code and observing behavior. Some part of the lab exercises, which are to be completed during the assigned lab session, will be turned in for credit.

Exams (35%) – There will be 3 exams, **tentatively** scheduled as follows:

- First Exam (10%), Wednesday, September 15, 2004
- Second Exam (10%), Friday, October 22, 2004
- Final Exam (15%), Monday, December 6, 2004 (10:30AM – 12:30PM)

Tentative Syllabus and Schedule of Classes

(We will spend about 3-4 lectures per chapter, with scattered lectures on broader topics of computer science.)

Date	Topics	Reading
Aug 23-30	Introduction to computer science and computing hardware and software	(Lecture notes, to be posted and / or handed out)
Sep 1-8	Programming and introduction to object-oriented design	Chapter 1
Sep 10-17	Java syntax, classes and objects	Chapter 2
Sep 20-27	Simple types, arithmetic operators and functions	Chapter 3

Sep 29-Oct 6	Control flow (conditionals)	Chapter 4, Sections 1-4
Oct 8-18	Abstraction and encapsulation	Chapter 4, Sections 5-7
Oct 20-29	More control flow (looping) and file IO	Chapter 5
Nov 1-5	OO design and implementation concepts	Chapter 6
Nov 8-15	Inheritance and overloading, object IO	Chapter 7
Nov 17-22	Error-handling and exceptions	Chapter 9
Nov 29-Dec 3	Arrays	Chapter 10

Course Policies

- *Attendance Policy:* Please see the Berry College Viking Code for "Class Attendance Policies" (pp 10-11, 2004-2005 edition). Missing three (3) or more classes without justifiable reason (and appropriate documentation) will be considered excessive absences.

  Attendance records will be kept by the instructor. Sign-in sheets will be circulated every class period and attendance records will be kept from the sign-in sheet. **If your name is not readable on the list, you will be marked absent**. Signing for someone else will be considered a serious breach of academic integrity.

- *Academic Integrity:* Students are expected to have read carefully and understood the rules governing breaches of academic integrity that are to be found in the Viking Code (pp 16-17) and the Course Catalog (pp 27-28, 2003-2005 edition). Be aware that, unless otherwise specified, all assignments, labs, and examinations in this course are expected to be done on an individual basis. When it comes to learning and understanding the *general material* covered in class or *practice problems*, you may certainly use other references and / or have discussion with other students or people outside this class. However, when it comes to work that is submitted for evaluation in this course, all such work must be entirely your own. The only exception to this is that **you are very welcome to consult the instructor for assistance**.

  Furthermore, exercise extreme care that discussion of programming techniques does not result in collaborative efforts. If you discuss general programming techniques for solving a problem with other students or individuals, do not start writing any code of your own until you have engaged in at least half an hour of totally unrelated, non-computer work, to ensure that the code you write is entirely your own, and that you comprehend it completely. **Program source code that is submitted for assignments and projects will be actively checked for being "too similar" to other students' submissions and to any other existing code. Copying programs and code from other sources and trying to just make minor changes therein will be detected and can result in severe penalties, up to and including an F in the course.**

  Here are some examples of acceptable collaboration:

  - Clarifying ambiguities or vague points in class handouts, textbooks, or lectures.
  - Discussing or explaining the general class material.
  - Providing assistance with Java (or other programming language), in terms of using the development environment facilities, or with editing and debugging tools.
  - Discussing the code that is given out for the assignment.
  - Discussing the assignments to better understand them.

- Getting help from anyone concerning programming issues which are clearly more general than the specific project (*e.g.*, what does a particular error message mean?).

As a general rule, if you do not understand what you are handing in, you are probably cheating. If you have given somebody some code simply so that it can be used in that person's project, you are probably cheating. In order to help you draw the line, here are some examples of clear cases of cheating:

- Copying program or assignment files from another person or source, including retyping their files, changing variable names, copying code without explicit citation from previously published works (except the textbook), *etc.*
- Allowing someone else to copy your code or written assignment, either in draft or final form.
- Getting help that you do not fully understand, and from someone whom you do not acknowledge on your solution.
- Writing, using, or submitting a program that attempts to alter or erase grading information or otherwise compromise security.
- Copying someone else's files containing draft solutions, even if the file permissions are incorrectly set to allow it.
- Lying to course staff.
- Reading the current solution (handed out) if you will be handing in the current assignment late.

When in doubt, ask the instructor.

- *Late Work:* All exams must be taken at the announced date and time. Similarly, labs are to be completed and submitted at the scheduled time in order to receive credit. Homework or other assignments that are submitted late will be assessed a penalty of 5% per day late up to 5 days. These policies will be waived only in an "emergency situation" with appropriate documentation and/or prior arrangement with the instructor.

- *Disabilities:* Students with disabilities who believe that they may need accommodations in this course are encouraged to contact the Academic Support Center in Krannert Room 326 (Ext. 4080) as soon as possible to ensure that such accommodations are implemented in a timely fashion. Failure to contact the Academic Support Center will constitute acknowledgement that no disability exists and that no accommodations are needed.