

CS 161 Final Exam Review Objectives

General Computing Background

Draw and label a diagram of the basic computer model.

Describe the function of each component of the model.

Explain why information is represented in binary for computing.

Discuss the different types of programming languages, and compare them based on portability and execution speed.

Data Types, Variables & Literals, and Assignment

Understand the different data types (integer, float, string, and Boolean) available for basic computing.

Recognize the difference between literal and variable data.

Identify whether a given piece of information is integer, float, string, or Boolean.

Interpret code that includes a series of assignment statements.

Identify errors in code that includes assignments.

Selection

Identify syntax errors in **if** structures.

Formulate **if** structures to represent simple selection in a problem solution.

Recognize and evaluate Boolean expressions.

Interpret code that includes **if** structures.

Iteration

Identify syntax and logic errors in **for** and **while** structures.

Recognize situations that call for iteration, and state whether **for** or **while** is needed.

Formulate **for** and **while** headers to control iteration.

Interpret code that includes **for** and **while** structures.

Functions

Describe the role of functions (subroutines) in programming.

Recognize syntax errors in function headers.

Given a statement that includes a function call, state whether the function must include a return statement, and describe its parameter list.

Given a function header, write a statement that could call the function.

Interpret code that includes functions.

Describe the difference between the way array variables and scalar variables are handled as parameters.

Arrays (Lists)

Recognize problem situations that require (or benefit from) the use of arrays.

Initialize arrays for specific problems.

Formulate **for** structures to manipulate data stored in arrays.

Interpret code that includes array variables (including function calls).

Documentation

Describe the purpose of documentation in programs.

Recognize program comments.

Describe the elements of program documentation (comments, good identifiers, and white space), and give examples of how to use them.

Some Problems

Write a function named **countOcc** that is given a list and a value, and returns the number of times that value occurs in the list.

Write a function called **doubleMyMoney** that is given an annual interest rate and determines how many months it will take to double an investment, then returns that number. This requires repeatedly compounding the interest for a value and counting how many times it takes to reach the doubled amount.

Write a function **powerList** that is given two integers n and c , and returns a list of the first c powers of n . (In other words, **powerList** (3, 4) would return the list [1, 3, 9, 27]).

Other Study Suggestions

Review the sample programs in the book and the sample solutions posted on the class web site. Don't try to memorize them—understand how they work.

General Remarks

The exam will be approximately one half multiple choice and one half short-answer and problem solving. You will trace at least one small program to determine its output and the values of variables at various points in its execution. You will not have to write a complete program of any size, but you may be required to write a function of just a few lines. Expect to look for syntax errors, but don't dread this—Python syntax is after all pretty simple. Just remember where colons appear (at the end of any header, be it **if**, **for**, **while**, or a **def** for a function), and remember variable scope and when to use “(“ and “)” versus “[“ and “]” and you should do fine.

When you trace code, be thorough and accurate; don't jump to conclusions!