1) Define the following terms:

(a) Computer Program:
Instructions for solving a problem to be performed by a computer.

(b) Algorithm:
A set of step-by-step instructions for solving a problem. Also ok: A set of step-by-step instructions for transforming input to output.

(c) Documentation (as it relates to a computer program):
Written text and comments that make a program easier to understand, use and modify.

(d) Machine Language:
The language made up of binary coded instructions that is run directly on a computer.

(e) Assembler:
A utility program that translates assembly language code into machine language code. It is one part of the "compilation" process with a C++ program.

(f) Compiler:
A rather complex utility program that translates a high level language (such as C++) into assembly code. This is always going to be machine dependent since the assembly code is specific to each machine architecture.

2) Describe the difference between the Problem Solving Phase and the Implementation Phase. Explain why it is important not to be tempted to bypass the Problem Solving Phase and go straight to the Implementation Phase.
The problem solving phase is where we solve the actual problem. We understand the problem requirements and generate a generic (pseudo-code) solution. This is an idea. The implementation phase has us translate our idea into a specific programming language. It is important not to bypass the problem solving phase because it is easy to conflate the two. When learning a new language, there is cognitive load in coding the problem that can detract from cognitive capacity of thinking about the problem. Plus when jumping straight into coding we often short-change the deep thinking about a problem and arrive at a suboptimal or incorrect solution.

3) Scan the section 1.3 on What's Inside a Computer. There are no questions for this topic, but I want you to get a cursory understanding of the main parts. This will be the central topic in CS281, but it is good to have a basic understanding for CS173.

4) What is the difference between how a C/C++ program is run versus a python program?
The main difference is that python is interpreted while C++ is compiled. This means that python programs run inside a "virtual machine" which is the python interpreter program. C++ programs

are translated directly into machine code and then they run directly on the hardware without that extra layer of abstraction (the interpreter).

5) What is software maintenance?
Software maintenance is the natural ending process of the software development cycle where software is fixed (bugs), and adapted and modified (changing requirements).

6) Define Software Engineering.
The application of traditional engineering methodologies and techniques to the development of software.

7) Why is software engineering an important aspect of computer programming? What can go wrong if we don't embrace the principles of software engineering?
The primary reason to embrace software engineering is to develop code that has fewer errors and bugs. Bugs in software can have serious consequences for injury to people and loss of material goods. Software engineering can also streamline the development of large projects where the responsibility is spread across a large number of people.

8) Describe the Divide-and-Conquer approach to problem solving.
Divide and conquer is an algorithm design technique where we break a big problem up into smaller pieces that are usually smaller instances of the exact same problem, solve those smaller pieces (often recursively) and then combine the small solutions into one big solution. Do not confuse this with "functional decomposition" which is very similar. Functional decomposition is really more of a software engineering approach to avoid large, complex functions and blocks of code. And the subproblems by functional decomposition are usually not the same and require the construction of different functions. Whereas divide and conquer is an algorithmic technique (design idea, not a coding methodology) where the subproblems are usually smaller instances of the big problem. Usually we handle divide and conquer through recursive calls instead of by creating blocks of other functions.

9) Who invented the C programming language and when? Who invented the C++ programming language and when?
Dennis Ritchie invented C in the late 1960s. Bjarne Stroustrup invented C++ in 1985.

1) Define syntax and semantics.
Syntax is the formal set of rules indicating how instructions are written in a programming language. Semantics is the set of rules that translate syntax (statements) into meaningful actions. You can think of syntax as the grammatical rules and semantics as the meaningful actions taken by a program.

2) Explain why there must always be a main() function in every C/C++ program. Explain the purpose of the return 0 statement at the end of main().
Every c++ program has a main() function because the operating system calls this function when the program is executed. It is expected that the function will return an integer code indicating the status of the completed program; a return value of 0 tells the operating system that everything executed correctly.

3) What are the rules for valid C identifiers?
C++ identifiers serve as names (variable names, function names, etc) in a program.

- Each identifier must be composed of letters (a-z, A-Z), digits (0-9) and underscores.
- They must begin with a letter or underscore.
- Reserved words may not be used as identifiers.

4) What is a C variable declaration? What purpose does it serve? Why don't we have type declarations for variables in python?
A declaration is a C++ statement that defines an identifier and assigns a datatype, function, or data object with that identifier. These are necessary so that the compiler can check our program to be sure our identifier is being used correctly (syntax). Python does not include declarations because it is a dynamically typed language (as opposed to a statically typed language).

5) Give an example or two of some data that might be stored in each datatype:

(a) int 3, 0, -5

(b) float 2.6, -5.0, 3.14e12

(c) char 'a', '9', 32 (an integer in a specified range)

(d) string "Hello World!", "a"

6) What does the C++ preprocessor do?
The preprocessor scans our c++ programs before the compiler and does simple translation tasks which are usually designated by # symbols in our code. The #define is a straight find-replace substitution operation while the #include copies in code from other modules (often function and variable declarations).

7) How do we signify "special characters" for output? Give three examples of special characters.

Special characters in output strings are designated by the \ character. For example "\n" prints a newline, "\t" prints a tab, and "\\" prints a slash.

8) What is the difference between a reserved word and a constant in C++?

A reserved word is part of the language used by c++ for special purposes. A constant is an identifier that we create in our program which doesn't change value in the program.

9) What is the purpose of the `#include <iostream>` statement?

This preprocessor statement finds a file called iostream.h on our computer and copies in its contents to our program at this point. These #include statements allow us to incorporate other modules (predefined functions) for use in our programs in much the same way that import statements did with python.

10) What is the purpose of a namespace?

A namespace is a protected "scope" of our program where we can establish identifiers so that they do not conflict with other identifiers of the same name in other parts of the program.

11) Convert the following binary values to decimal representation:

(a) 10101_2
21 in base 10

(b) 10101101_2
173 in base 10

12) Convert the following decimal numbers to binary representation:

(a) 777_{10}
$11\ 0000\ 1001_2$

(b) 101_{10}
$110\ 0101_2$

List the linux commands below which perform the indicated task:

(a) List the contents of the current working directory.

(B) Print the current working directory path.

(C) Change back one directory (to the parent of the current directory).

(D) Create a directory named cs173.

Explain why every c++ program must have a function named main().

What is the purpose of the following statement in c++? Be specific about the purpose of both #include and <iostream>.

Convert the following decimal number to binary and binary number to decimal.

(a) 567_{10} (b) 10111_2

Linux Commands

pwd – print working directory, tells us where we are
ls – lists everything in the working directory (or a specified directory)
cd [path to a directory] – change directory, move where we are
mkdir [path to a directory] – make a new directory
touch [path + filename] – makes a new file
python [path + filename] - runs a .py file
g++ [path + filename] -- compiles a .cpp file

- Use -o [ofilename] to specify an output file name
- If no output file is specified, a.out is the produced file
- Use –g to compile so the executable can be run with a debugger like gdb

./[ofilename]

- Runs an executable file

mv [source_path + filename] [destination_path + filename] -- moves a file (possibly renames it)
rm [path + filename] -- removes a file
cat [path + filename] -- view the contents of a file
less [path + filename] -- view contents of a file in a scrollable window

- Press q to exit back to the terminal!

make

- Runs a makefile in the current directory

include <iostream>

- Preprocessor directive, similar to import
- C standard library split into header files
- iostream header file lets us use input and output
- Contains istream and ostream objects
    - Ostream object: cout
        - Insertion operator: << puts output to an ostream
        - endl manipulator: moves to the next line (inserting a newline character

Don't need def keyword to define functions
Before the name of a function, we need to specify the type it returns

- int
- void (doesn't return anything)
- float/doubles
- char
- bool
- string

Syntax differences:

- Instead of whitespace and colons, use {} to denote code blocks
- Statements end with ;
- Comments denoted with //

main() functions in C++ are required for a program to run, entry point for the operating system
- must return an int

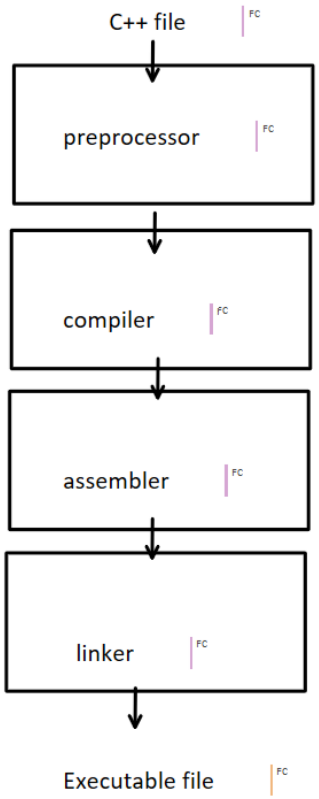
- Returning 0 indicates success
- Get called when the program runs

Python is compiled in little chunks and run on a virtual machine, interpreted so it runs line by line until an issue is found

- Portable

C++ gets compiled/translated into machine language, produces an output file which can be run on a specific machine

- Compiler can help us find a lot of issues
- Faster!
- Output file produced on one machine can't be run on other machines



More detailed breakdown of the steps of the compilation process

|

V

Handles preprocessor directives, copies over content from included header files

Translates into assembly language
- More in 281!

Translates into machine code (.o files)
- More in 281!
- But we'll talk a bit about machine code
- Binary

Combines multiple .o files

Representing Values with Binary

Monday, January 27, 2025 1:57 PM

1 byte = 8 bits

Range of values that can be stored in one byte:

00000000_2 to 11111111_2

2^7 ... 2^0 2^7 ... 2^0
128 128

0_{10} $255_{10} = 2^8 - 1$

Values that can be stored in *n* bits:
0 ... $2^n - 1$

Types in C++ are represented with a specific number of bytes:
chars (1 byte)
ints (often 4 bytes) vs. shorts (often 2 bytes) vs. longs (at least 4 bytes, can be more)

Types can be signed (default for numerical types) or unsigned:
To represent signed types:
If leftmost bit is 1 -- value is negative (-2^{n-1} to -1)
If leftmost bit is 0 -- value is nonnegative (0 to $2^{n-1} - 1$)