

Programming 1

Lecture 1 - Introduction
COP 3014 Spring 2022

January 7, 2022

Some More Course Policy

1. How to find your section number?

- ▶ Your “section” depends upon your discussion session time. It is the attached to COP 3014 on your course schedule on myFSU, NOT the number on Canvas. We cross-list multiple section son Canvas and it shows only 1 of those numbers.
- ▶ You can also find your section number on the syllabus. Match your Discussion Session’s Time with your section.

2. How to find instructional staff for Help Sessions?

- ▶ The instructor, TA and LA’s have offices in the JJ Love building which is currently under renovation.
- ▶ We will list temporary office locations and times on the course website once we have a full list of instructional staff, and send out a course announcement.
- ▶ We will also announce in class and send out course announcements as people slowly move into their permanent offices upon completion of renovation.

Some More Course Policy

1. General Email Policy

- ▶ The instructor, TA's and LA's are available through email.
- ▶ When you send us email, please make sure you use an appropriate subject line.
- ▶ If you need us to look at your code, please attach the source code file. No screenshots please.
- ▶ Emails received before 5PM on business days will generally get a same day response. Emails received after 5PM will get a response (if required) on the next business day.
- ▶ Most of us receive canvas Messages as a digest (all of them in one lot, once a week). So, Canvas messages might not get a very quick response. Please email us directly (from outlook).

2. Special Note for emailing the instructor

- ▶ Please use Sharanya's CS email (jayarama@cs.fsu.edu) - preferred, or FSU work email (sjayaraman@fsu.edu). Simple looking it up on outlook might result in the email going to other people with similar names (there are at least 2 of them).
- ▶ If your email goes to the wrong person, or Sharanya's student email, the response may be delayed.

Some More Course Policy

- ▶ The final grade is calculated on a weighted average based on percentages, as shown in the syllabus. All assessments are not the same in terms of points.
- ▶ The grade totals are hidden on Canvas, since Canvas overestimates ungraded assessments. If you wish for an accurate picture of your current standing, please request an Excel formula/file from the instructional staff.
- ▶ **Please read the course announcements sent out through Canvas.** They usually contain explanations regarding grades or assignments.
- ▶ **If we need to contact you individually, we will send email to your myFSU email.** Please check that regularly.

Some More Course Policy

This course is partnered with the Proactive Referral Engagement Program (PRE)

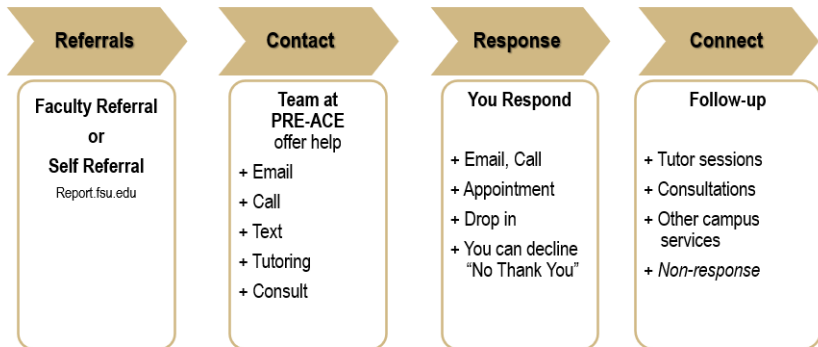
1. What is PRE?

- ▶ Early support for students to achieve their academic goals
- ▶ Provide supplemental course-based tutoring and other services
- ▶ Prevent course failure, repeating course, or unnecessary drops/withdrawals

2. Why?

- ▶ We want you to succeed in this course!
- ▶ You have the capacity to succeed in this course
- ▶ Freely available services for you
- ▶ Quality academic support by vetted tutors on campus and online
- ▶ Private consultations about strategies to address personal challenges

How does PRE work?



ace.fsu.edu

Some More Course Policy

Attendance Policy

- ▶ Attendance and Class Participation is required to do well in the course. **However, we're living through an exceptionally difficult time**
- ▶ **If you're ill, please stay at home, and let the instructor know.** We can make arrangements to make sure you're caught up with material.
- ▶ Please follow FSU guidelines and recommendations with respect to absences and safety protocols.
- ▶ This is a fairly large class. Please work with us in creating a safe and positive learning experience for everyone!

Programming is a Ubiquitous Discipline

- ▶ Why study Computer Programming?
- ▶ Computers are used in nearly every field. A basic understanding of how to effectively use a computer is becoming essential in several disciplines.



What makes Computer Programming Different?

- ▶ Almost everyone uses a computer.
- ▶ The most common uses of computers include end-user-applications (Apps) that do not involve any modifications of the programs. These are “shipped” to the user as a finished product.
- ▶ Some applications are “tools”. These allow some freedom to put the processing power of the computer to use data and some direction from the user to accomplish a task.
- ▶ For example, we can use Microsoft Excel or an ERP application to generate Reports. This involves some knowledge of what the underlying system/data is capable of doing.

What makes Computer Programming Different

- ▶ Computer Programming can be done on two levels.
- ▶ **Application Development** using mostly existing software products and libraries to put together a product for a specific area of use. This requires domain knowledge as well as a general understanding of Computer Science.
- ▶ **Software Development** using basic programming language constructs to develop the software and libraries used in Application Development. This requires deeper knowledge of Computer Science, including theory and hardware.
- ▶ Both of the above can be considered **Software Engineering**.

Main Components of a computer

- ▶ CPU - Central Processing Unit: The “brain” of the computer.
 - ▶ ISA - Instruction Set Architecture: the specific set of low-level instructions available to a CPU. Differs for various CPU types (Intel Pentium, Mac G4, etc).
- ▶ ALU - Arithmetic & Logic Unit responsible for performing arithmetic calculations, as well as logical operations (comparisons for equality, inequality, for instance).
- ▶ Main Memory (RAM - Random Access Memory).
 - ▶ storage close to CPU
 - ▶ Faster to access than hard disk
 - ▶ stores executing programs and data being currently worked on
- ▶ Secondary Memory
 - ▶ SSD, hard disk, DVD, etc.

Main Components of a computer

- ▶ Input devices
 - ▶ mouse, keyboard, scanner, network card, etc.
- ▶ Output devices
 - ▶ screen/console, printer, network card, etc.
- ▶ Operating System
 - ▶ Examples: Mac OS, Windows 10, Linux
 - ▶ Controls computer operations
 - ▶ Manages allocation of resources for currently running applications

Memory Concepts

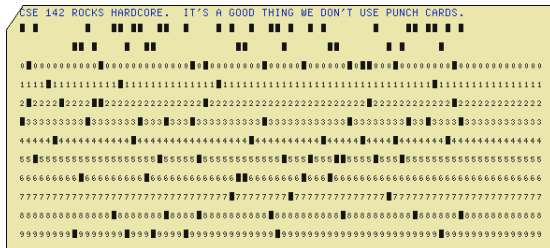
- ▶ bit: a binary digit
 - ▶ Stores the value 0 or 1
 - ▶ Smallest unit of storage in a computer
- ▶ byte: 8 bits
 - ▶ Smallest addressable unit of storage in a computer
 - ▶ Storage units (variables) in a program are 1 or more bytes
 - ▶ Each byte in memory has an address (a number that identifies the location)

Programming, and Programming Languages

Program - a set of instructions for a computer to execute

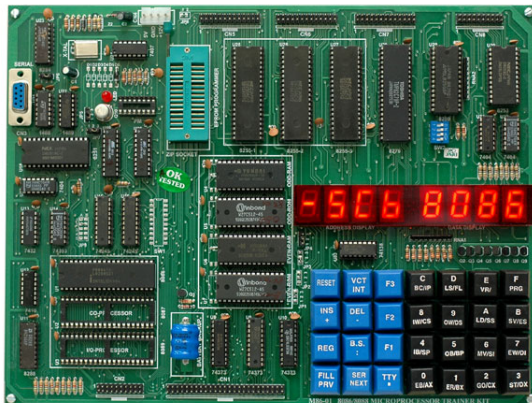
Evolution of Programming languages

- ▶ Machine Language
 - ▶ Based on machine's core instruction set
 - ▶ Needed by computer, hard for humans to read (1's and 0's)
 - ▶ Example: 1110110101010110001101010



Programming, and Programming Languages

- ▶ Assembly Language
 - ▶ translation of machine instructions to symbols, slightly easier for humans to read
 - ▶ Example: `ADD $R1, $R2, $R3`



Programming, and Programming Languages

- ▶ High-level procedural languages
 - ▶ Abstraction of concepts into more human-readable terms
 - ▶ Closer to "natural language" (i.e. what we speak)
 - ▶ Easy to write and design, but must be translated for computer
 - ▶ Examples include C, Pascal, Fortran
- ▶ Object-oriented languages
 - ▶ Abstraction taken farther than procedural languages
 - ▶ Objects model real-world objects, not only storing data (attributes), but having inherent behaviors (operations, functions)
 - ▶ Easier to design and write good, portable, maintainable code
 - ▶ Examples include Smalltalk, C++, Java

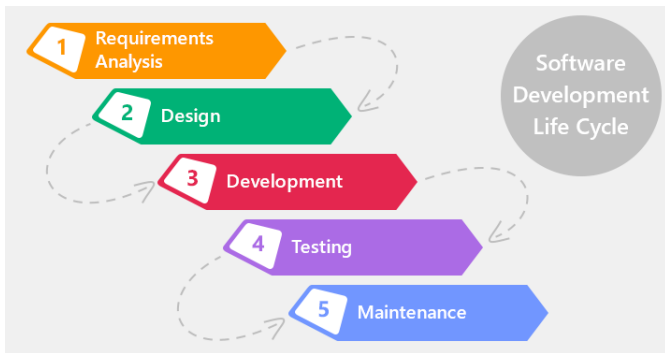
Code Translation

Bridging the gap between high-level code and machine code

- ▶ Interpreted languages – source code is directly run on an interpreter, a program that runs the code statements
- ▶ Compiled Languages
 - ▶ A compiler program translates source code (what the programmer writes) to machine language (object code)
 - ▶ A linker program puts various object code files together into an executable program (or other target type, like a DLL)
 - ▶ C and C++ are compiled languages

Software Development

Involves more than just writing code



Software Development

- ▶ Analysis and problem definition
- ▶ Design - includes design of program or system structure, algorithms, user-interfaces, and more
- ▶ Implementation (coding)
- ▶ Testing - can be done during design, during implementation, and after implementation
- ▶ Maintenance - usually the major cost of a software system. Not part of "development", but definitely part of the software life cycle

Programming is about Problem Solving



Programming is about Problem Solving

- ▶ Algorithm - a finite sequence of steps to perform a specific task
 - ▶ To solve a problem, you have to come up with the necessary step-by-step process before you can code it
 - ▶ This is often the trickiest part of programming
- ▶ Some useful tools and techniques for formulating an algorithm
 - ▶ Top-down Refinement: Decomposing a task into smaller and simpler steps, then breaking down each step into smaller steps, etc
 - ▶ Pseudocode: Writing algorithms informally in a mixture of natural language and general types of code statements
 - ▶ Flowcharting: If you can visualize it, it's often easier to follow and understand!

Programming is about Problem Solving

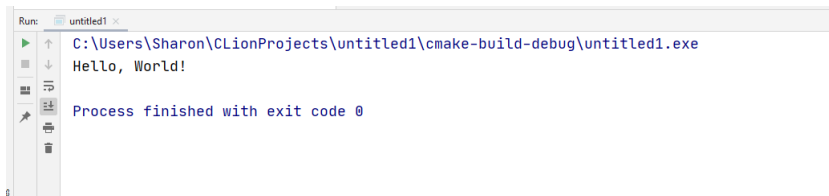
- ▶ Testing - algorithms must also be tested!
 - ▶ Does it do what is required?
 - ▶ Does it handle all possible situations?
- ▶ Syntax vs. Semantics
 - ▶ Syntax – the grammar of a language.
A syntax error: "I is a programmer."
 - ▶ Semantics – the meaning of language constructs
Correct syntax, but a semantic error: "The headphones ate the tree."

Basic Creation and Execution of a C++ program

- ▶ Create **source code** with a text editor, store to disk.
 - ▶ Source code is just a plain text file, usually given a filename extension to identify the programming language (like .c for C, or .cpp for C++)
- ▶ **Preprocessor** – Part of compiler process, performs any pre-processing tasks on source code.
- ▶ **Compilation** – syntax checking, creation of *object code*.
 - ▶ Object code is the machine code translation of the source code.
- ▶ **Linking** – Final stage of the creation of an executable program. Linking of object code files together with any necessary libraries (also already compiled).
- ▶ Execution of program
 - ▶ Program loaded into memory, usually RAM
 - ▶ CPU executes code instructions

Our First C++ Program

- ▶ Opening a New Project on CLion will create a default “Hello World” program.
- ▶ Most C++ programs start out with the same set of basic elements. We will cover these in the next lecture.
- ▶ Right now, click on the green “Run” button on the top right. Now look at the Window on the bottom. It should say “Hello, World!”



- ▶ All the phases mentioned in the previous slide were triggered by the “Run” button.