What else do you need to know about this class?

What is the goal of this course?

This is a beginning programming course. There is no programming prerequisite for the class, but it is assumed that you can read the textbook and technical instructions and that you are experienced with basic algebra. Upon "successful" completion you will have attained "basic" programming skills, and be ready to continue in further programming classes (e.g. CIS22B).

What happens in the class?

Four lectures per week, 50 minutes each. Each lecture will contain new material. If you miss a lecture, you will miss required topics. The lectures will contain concepts and sample code, written on the board, web page examples, explained, and live demonstrations of designing, writing, compiling, and testing code. There will be practice problems written on the board. At the end of the Tuesday lecture, there will be an assigned lab exercise. This is meant to be a quick review of some of the topics covered in the lecture. Sometimes assignment details and hints will be discussed in class. There is a midterm and a final. And there is time to ask questions. If you ask questions, then, most likely, you are involved and you have some basic understanding of what is going on. If you do not ask questions, then either you are not involved, you don't care, you are not "getting it", or you are too shy to ask a question - try to get over that.

Your "compiler"

In this course you will be writing a lot of code. You will be compiling all the code you write**. It is mandatory that you acquire a compiler immediately**. They are free and you should download and install one on your computer right away. The instructor can assist with that process. You can use any "standard", "not-too-old" C++ compiler. You can work on any computer type, PC, Linux, Unix, Chrome Book, desktop, laptop, Windows (7/8/10). Tablets and smart phones won't work for this. You do not have to spend any money for this. The ATC computer lab works great and there are several compiler options available there. A laptop is good, and if you bring it to class, then you can download code and/or try out the same code that you are seeing live. Compiler suggestions: Code::Blocks, NetBeans, MS Visual Studio/C++ (after 2010), the gnu compiler, and Eclipse. Mac compilers (Xcode, Code::Blocks, Eclipse) are not recommended.

If you do not know which compiler you want to use, then use Code::Blocks on a PC. It is easy to use and does a very good job of enforcing the "standard".

Online Time

Online time will be held on Tuesdays, 7:00-8:15 pm. This time will be used to review topics covered in class, to answer questions, to discuss assignments, to work exercises and practice problems, and to explore related topics in more detail. The online time will be held online using the web-based, interactive software, **ConferZoom**. The instructor will be logged on, speaking through a webcam and sharing his desktop to answer questions and provide demonstrations. You may ask questions using your microphone or a chat box. You will need a ConferZoom account to access the software.

Programming Assignments

- Email source code. Check your code before sending it. Once received, it will be graded.
- In the email, use ***Ass***# as the email subject (where # is the assignment number)
- Add the source code as an attachment. Name it ***ass#.cpp*** (where # is the assignment number).
- **Add comments to the top of your source code including your name, CIS22A, the assignment #, the operating system and the compiler used.** If you are using Code::Blocks, indicate whether you are using it on a PC or a Mac. **You will be penalized one point if you are missing this comment.**
- If you want to ask a question about the assignment, make sure it is clear that you are asking a question, otherwise, it will be graded. D**o not ask to have your assignment checked before you submit it**.
- The code will be compiled and tested using Code::Blocks on Windows. If your code does not compile and run with the instructor's compiler, you will be penalized for not writing "standard" code.
- Assignments are due at the beginning of the class lecture (8:30 am) on the due date specified. **Assignments will be accepted late with a 5 point penalty if they are received within 24 hours of the due date. After that time, they will not be accepted**.
- You will receive an email response for every assignment submitted with feedback on you assignment.

Lab Exercises

- Start a new email for each lab exercise.
- In the email, use ***Ex***# the email subject (where # is the exercise number).
- Add the source code as an attachment. Name it ***ex#.cpp*** (where # is the exercise number).
- Check your code before sending it. Once received, it will be graded.
- If you are asking a question about the exercise, make sure it is clear that you are asking a question. Otherwise, it is assumed that you are submitting it for grading. Do not ask to have your exercise checked before you submit it.
- Add comments to your source code including your name, the assignment #, operating system, and compiler used.
- The code will be compiled and tested on Code::Blocks running on Windows.
- Lab exercises are due at the beginning of the Wednesday lecture (8:30). **Lab exercises are not accepted late**.
- You will receive an email response for every lab exercise submitted with feedback on you exercise.

CodeLab

- The CodeLab exercises are not required. You will be given extra credit points for the exercises that you complete.
- These are web page exercises. They are free if you are enrolled in the class.
- You must set up your account
- The web page address for CodeLab is https://codelab3.turingscraft.com/codelab/jsp/login1.jsp
- Access code**: DEAN-27288-JPPP-42**
- All exercises will have due dates (by chapter).
- You may ask for help on any CodeLab exercise. You can use on-line time, office hours, or email to ask for help
- Your CodeLab extra credit points (10 maximum) will be determined by the percent of exercises completed.

If you want an A, …

- sit in front of the class
- ask questions
- pay attention to the entire lecture
- learn to debug your code
- test your code as you proceed
- ask yourself, "what if ..."
- learn to write functions to solve a specific task
- learn to write reusable code, tight code, efficient code, self-documenting code, neat code
- learn to read code
- use a second compiler
- **And get involved in the class. Make sure Joe learns your name (soon)**

Textbook

The textbook is a good one. The 8th edition of the textbook is acceptable for this class. You are advised to read the chapters following the schedule listed in the syllabus. Some class examples may be taken from the textbook.

The class web pages

- There will be updates to the class web pages. Refresh your browser if you are viewing a web page a second time.
- Lecture notes web pages will be updated on the weekend before the lectures.

The Midterm and Final

These are timed. You will be permitted one page of notes (both sides of the paper). Preparing this page of notes is a good way to study for the test.