Welcome!
I and the entire teaching staff for CSE8A are incredibly privileged to welcome you to this class. For most of you, we also extend a warm welcome to the computing profession. We and the entire CSE department look forward to working with you in the days and years to come to help you develop the skills and abilities to transform and improve society through the understanding and application of computation.

What does this course offer to me?
This course provides an introduction to the basics of computation. This includes understanding and being able to employ (in a programming language) basic computing concepts which allow us to control a computer and to design programs for it so it can solve problems for us. No prior programming experience is expected for this class. In this class, our goal is to help you experience the thrill of getting a computer to solve a problem of your choosing – by expressing that solution in the Java programming language.

Why would I want to learn a programming language rather than just a computer application (like PhotoShop)? First, applications like PhotoShop are, themselves, computer programs. Someone wrote them as solutions to a set of problems they wanted to be able to solve. But, what if you have ideas for solving a problem that PhotoShop can’t solve? How can you get a computer to follow your directions? You have to write a program. Programs are one of the most direct ways one can give instructions to a computer. They give you a great deal of control over the computer – the kind of control needed to develop novel software to solve problems of your choosing. The kind of control that will allow you to explore and design solutions for society’s ever changing computing needs.

Particularly in recent years, computation has come to play a critical role in the advancement of many fields – including biology, medicine, chemistry, film, archeology, etc. Our world is full of data – and being able to manipulate data or analyze it in an efficient or novel way can lead to dramatic new findings in many fields. In this class, you will learn the computational basics of flow of control and data representation which will allow you to express solutions to many problems – and to learn other programming languages (besides Java) to allow you to accomplish your goals.

How, exactly, am I going to accomplish this?
The class has two components – “lecture” and lab. They are designed to fit together. Both seek to give you introductory experience in the critical aspects of software development including understanding of new computational concepts through being able to read, understand, modify, and debug code in addition to being able to write code. Notably, most modern software is developed in teams. We have designed pair programming experiences, lecture team discussions, and programming homework interviews to start giving you practice to be a successful software developer. Specifically, you can expect the following cycle to repeat itself throughout the weeks of the course:

- Each week, before lecture, you’ll read a set of pages in the textbook to give you your first experience with the new computational concepts of the week.
Then, in lecture, we’ll work together to test out your understanding of the material and discuss confusion and misconceptions that you may have about it. Note – this makes lecture a very interactive place – you are expected to come prepared to engage and help yourself and others gain a much deeper understanding of the concepts.

Lab comes in as the next step. In lab you will work in teams to get your first “hands on experience” with putting computational concepts into practice. Labs are short, and directed at likely mistakes you might make. They will give you the time and support to understand those mistakes and ask questions about them so you won’t make them again.

In discussion section, you and the TA will discuss problems closely related to the homework including previous week quiz and homework solutions. Sample test questions will be provided and solved as a group. In addition, discussion section is another forum for questions.

Finally, both lectures and labs are a time to help you get prepared for the programming homework assignments – where you can experience for yourself the thrill of getting a computer to follow your directions and create something of your choosing. This is the part where you can really test out your knowledge and understanding of computational concepts. Every line of code you write should be understandable to you – and you will be expected to be able to explain how your code works and why to someone else.

How will I (and the professor) know if I am making progress in my learning?
Each aspect of this course is designed to help you develop deeper understanding of the computational concepts that will allow you to produce computer programs to solve problems of interest. For each new concept, you’ll follow the reading, lecture, lab, homework process to give you repeated experience with that concept, at increasingly complex and less constrained levels. Along the way we’ll also have weekly quizzes and exams to help you assess for yourself how well you are mastering the material. You should reflect on each of these experiences each week to determine for yourself if you are making adequate progress – and plan specific changes if you are not. (See checklist on web site for more details on assessing your progress).

The key goal for you (and me) in this course is that, by the end, you can employ the computational concepts we’ve learned to solve problems by writing a computer program. In a very short period of time (maybe next summer!) you may be paid to help people get the computer to solve their problems – through writing a computer program. Whether or not you fully understand a particular concept at the period of time we work on it in class only is important in terms of helping you stay on track with the pace of the course. The critical thing is that – by the end of our ten weeks together – you can understand and employ those computing concepts to effectively solve problems. This means that you can read, understand, explain, modify, debug, and write code using these concepts. These are the first skills to be developed in becoming a software developer and a computer scientist.