Welcome!

# Lesson Notes

## Intros

* Introduce teachers.
* Personal: In high school, computer science sparked my imagination, and I’ve been hooked ever since. I was excited about teaching this class because it is a chance to help others find that “spark”. Also, I was excited because I think this topic can actually be fun.
* Discuss role of each teacher.

## Why did you sign up for this class?

* **Students**: Raise hand if this is one of the reasons you signed up for the class.
* This class is valuable even if you don’t go into computer science.
  + Learn logic
  + Get practice breaking down problems
  + Know instinctively what would be easy for a computer to do
  + Be less mystified as a user of computers
* This class is valuable if you do go into computer science.
  + We are covering many concepts that appear in intro CS classes
  + We are giving you programming experience
* Personal: What excites me is that I can make a computer do something that matters to **me**.
* Personal: When I was in 10th grade I created a “lunar lander” game – I was hooked!
* Mention survey at the end of the class.

## What is computer science?

* There are many definitions of computer science.
* Background: The ACM is an educational and scientific computing community. <http://www.acm.org/>
* In order to understand this definition, we first need to understand data structures and algorithms.

## What is an Algorithm?

* Pieces of data:
  + Not just 4 inputs and 2 outputs (example).
  + Inputs and outputs are any digital data, including:
    - Data you use and store on your computer: images, graphics, tables, documents, graphics, sound, movies, etc.
    - Data that comes as input into your computer: keyboard, mouse, touch, camera/video, microphone, etc.
    - Data that is output by your computer: monitors, speakers, printers, etc.
  + This can be anything you can dream up!
* Example algorithm
  + Input: X
  + Output: a number
  + Algorithm: add 1 to X
* Algorithms can be defined in many ways:
  + As English sentences
  + As mathematical formulas
  + As computer code
  + (etc)

## A shopping algorithm…

* Your family is headed camping, but needs to stop at the grocery store on the way to the campground. You have a grocery list in hand, and you want to get done as quickly as possible so you can be spending the day in the outdoors instead of in a grocery store. What do you do? How do you get your shopping done as quickly as possible?
* Inputs:
  + The location of shopping items in the grocery store
  + The layout of the grocery store
  + We’ll assume we know a good way to visit all the areas of the store
  + We’ll assume we know how to walk to a particular item in the store
* Outputs:
  + The path(s) you take through the store
* **Students:**
  + Break into groups and come up with ideas on what algorithm to use.
  + Representative from each group presents approaches to the class after time period.
* Some approaches for discussion:
  + For each item on the grocery list, walk to the item, put it in the cart.
  + For each area of the grocery store, see if there are any items on the list. We’ll assume that scanning the list takes less time than walking. Walk in a good order among the areas of the grocery store.
  + Tear the list into parts and give each part to a member of your family.
  + Group the list by areas in the grocery store, given each group to a member of your family.
  + Alphabetize the grocery list so that it is easier to find things on it.
  + Cross things off the grocery list once you find them, so it is easier to find things.
* What do we need to do to turn these ideas into algorithms?
  + We need to write down the steps of our algorithm.
* What do we judge these algorithms?
  + Amount of time it takes to do the shopping (length of the path)
  + Computer scientists often compare algorithms based on how long they take to finish
  + We had a small shopping list, but when the number of inputs gets big, the algorithm we choose really matters. For example, when you type a search, the search engine does not go through the whole internet looking for the answer – it has algorithms that work well for large amounts of data.
* There is more than one right answer!
  + In my experience, you don’t have the perfect answer every time, you have to have one that will work well for the situation.

## What is a Data Structure?

* Part of defining an algorithm is to define the data structures that will be used in the algorithm.
* The word data structure refers to what data is needed, and how it is organized.
* For example, is it a name? A number? A table of values? A location on a map? A picture?
* **Students**:
  + Break into groups and determine the data structures needed by the algorithms you just came up with.
  + Be specific! For example, a shopping list is not specific enough. What do you want to know about each item on the list? Is the list in any particular order?
  + What additional data will your algorithm use to keep track of what it is doing?
  + Representative reports back with findings after time period.

## So what is Computer Science?

* The “formal properties” refer to the mathematical properties of algorithms and data structures. This part of computer science is theory oriented, and is typically independent of any particular computer.
* The “mechanical realizations” refer to how data structures are stored, and how algorithms are executed within a computer.
* The “linguistic realizations” refer to how we write down data structures and algorithms using some language. This process of “telling a computer what to do” is called **programming** a computer. The result is a **program**, or **software**, that a computer knows how to run, or **execute**. The language we use for this called a **programming language**. There are many programming languages.
* The “applications” refer to what we use data structures and algorithms for. The things you download on your phone are applications (“apps”), but so is all the other software you use: facebook.com, Microsoft Word, an XBOX game, etc.
* Although this is a good definition of computer science, and is quite broad, it is not the only definition. For example, this does not include the impact that computers have had on society as part of the definition.

## Units

* In order to learn Computer Science, we have to learn to program a computer.
  + There are many ways to program a computer, and many types of applications you can build, and each comes with a bunch of details.
  + For example, creating an XBOX game is going to be different than creating facebook.com.
  + But each of these have a bunch of things in common, and that is really what we are trying to learn in this class.
  + The goal is to learn the common concepts so that when you need to create something new, you can learn the details much quicker because you understand the concepts.
* We have four units and we are going to spend the first three developing **web applications** using a programming language called **Javascript**.
  + Example of a web applications are: facebook, gmail, youtube, etc.
  + In the first two units you will be creating small web applications that work on your own computer, and in the third unit you will break into teams and create a larger web application that is part of the internet.
  + When we are ready, we will give the link to the application to the rest of the school so they can try it out.
* We are going to spend the last unit developing **desktop applications** using a programming language called **Java**. A desktop application are installed on your local computer. Examples are: your web browser (Chrome, Firefox, IE, Opera), Microsoft Word.
* In **Unit 1**, we will learn a series of building blocks and will start to combine them. Problems will typically have most of the solution provided to you, and you will need to fill in the rest.
* In **Unit 2**, we will focus on techniques for taking problems without a solution and figuring out how we can design algorithms and model data structures to fit the solution. This unit will be more like what we did with the shopping algorithm exercise.
* In **Unit 3**, we will learn a bunch of details about how to develop web applications so that we can build something bigger and something that matters to you. We will also learn to work in groups, which is very common when developing software.
* In **Unit 4**, we will look back on the larger application we created in unit 3, and take a look at some techniques we can use to better organize the code in our applications.

## Expectations