Introduction to Computational Thinking – ISC1057

Goals for this lecture:

- Go over the syllabus for the course

- Look at some characteristics of computational thinking

- To understand the difference in computer hardware and computer software

- Introduce the concept of algorithm

- Compare a "brute-force" algorithm and a "smarter" algorithm

- Get to know each other

ISC1057 - Computational Thinking Syllabus[1]

Instructors: Drs. Janet Peterson & John Burkardt
                          email: jpeterson@fsu.edu, jburkardt@fsu.edu
                          Offices: Peterson: 444 DSL   Burkardt: 445 DSL
                          Office Hours: Peterson: Tues 11-12, Thurs 1-2
                                            Burkardt: Mon 11-12, Wed 1-2

Teaching Assistant: Mike Conry
                          email:mc12s@my.fsu.edu
                          Office: 465A DSL
                          Office Hours: Mon, Tues 1-2

Class: TR 2-3:15 p.m.
Location: 217 HCB
Blackboard Site: ISC 1057

Texts: *9 Algorithms That Changed The Future: The Ingenious Ideas That Drive Today's Compute*
                          John MacCormick, Princeton University Press, 2012;
                          Paperback ISBN 978-0-691-15819-8; cost $< \$15$

                          Essays by Brian Hayes, downloadable from the
                          website http://www.americanscientist.org/authors/detail/brian-hayes

---

[1]Full syllabus found on Blackboard

Course Description. It is clear that computers can almost imitate human-like intelligence. The evidence of this is everywhere around us: movie, book and music recommendation systems; programs that allow us to experiment on models of the earth; medical imaging software that can detect tumors that humans can't see. This course asks how computers have gained this ability. The answer includes our detecting patterns in nature, but also patterns in the very way we think. This course will present popular computational methods shaping our lives, and try to explain the ideas that make them work. Students will practice logical thinking by working with versions of these computational methods that affect society and science. Knowledge of a computer programming language is not required nor will it be taught.

Final Grade Determination. Your grade for the course will be determined by class participation, homework and group projects. The distribution of grading for the course is:

- Class Participation - 10%
- Quizzes - 25%
- Homework - 40%
- Projects - 25%

- The in-class quizzes will be taken using the application **Socrative** which you will access through your smart phone. **Quizzes** will be given almost daily and are typically given at the end of each lecture. Only quizzes taken in class count; there will be no make-up quizzes. However, you are allowed to drop some quizzes because we will only keep your top twenty scores.
- The class participation portion of your final grade will be determined by the percent of these quizzes that you attempt. Your score on the quiz does not affect the class participation grade; essentially we are using the quiz to count for attendance.
- Also during many of the lectures you will be asked to take **Practice Quizzes** which help you (and the instructor) assess whether you understand the material; these do NOT count towards your grade but are for your benefit.

- Assigned homework must be done individually (unless otherwise indicated) and submitted by the due date unless you are using your late homework days. **You will have a total of 7 late homework days throughout the semester** unless extenuating circumstances arise. Homework assignments consist of problems similar to the examples discussed in the lecture and assist the student in understanding the material.

- Projects are much more involved than homework assignments and serve to enhance or expand on the material covered in class. Consequently, students will be expected to work on the project over a longer span of time. Two group projects will be assigned during the semester. The projects will require a substantial amount of internet research and writing. For each project, several choices of topics will be suggested to accommodate the varied interests of students.

This course is approved to satisfy FSU's Liberal Studies Quantitative/Logical Thinking requirement.

In order to fulfill this requirement the student must earn a "C" or better in the course.

Some quotes about Computational Thinking

Computational Thinking is the new literacy of the 21st century. The goal is for it to be a fundamental skill used by everyone in the world by the middle of the 21st century. Just like reading, writing and arithmetic.

Even people who aren't going into computer science or engineering programs, should be exposed to computer science. Programming teaches you how to take problems, break them down into smaller problems, and solve them. Any kind of job that involves problem solving can benefit from computing.

Computers are extremely talented at performing menial tasks quickly; Computational Thinkers can leverage that power to find solutions to previously impossible problems.

I think everyone should get a little exposure to computer science because it really forces you to think in a slightly different way, and it's a skill that you can apply in life in general.
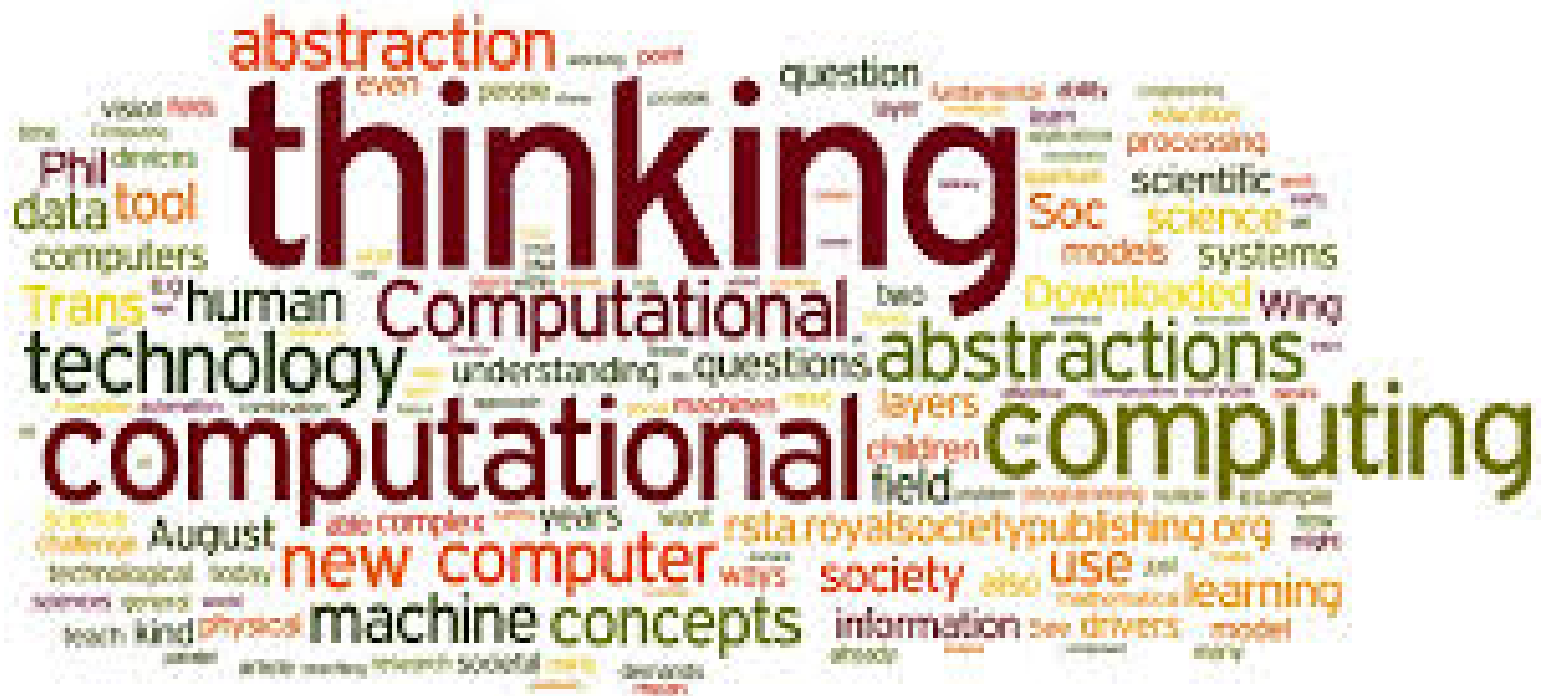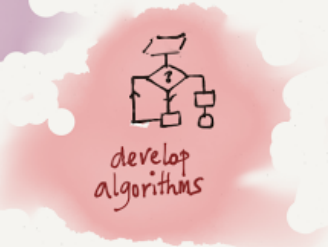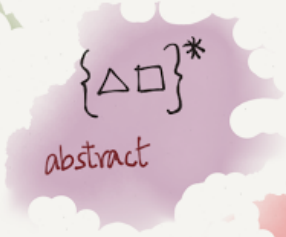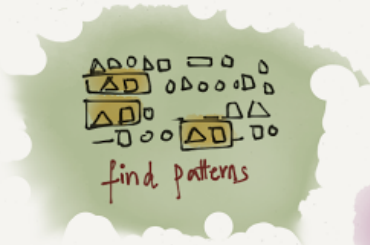
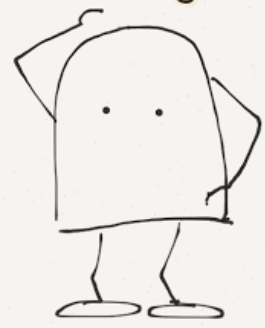# What is computational thinking?

Who needs it?

Why?
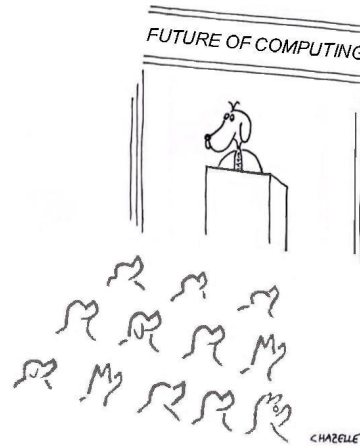
"Soon, my friends, you will look at a child's homework - and see nothing to eat."
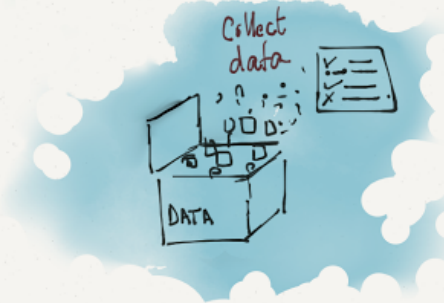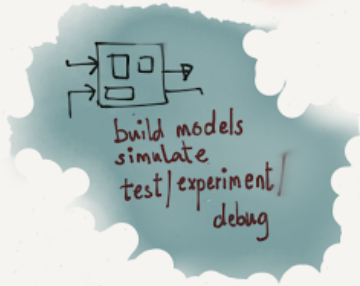
Computational Thinking is a problem-solving process that includes the following characteristics.
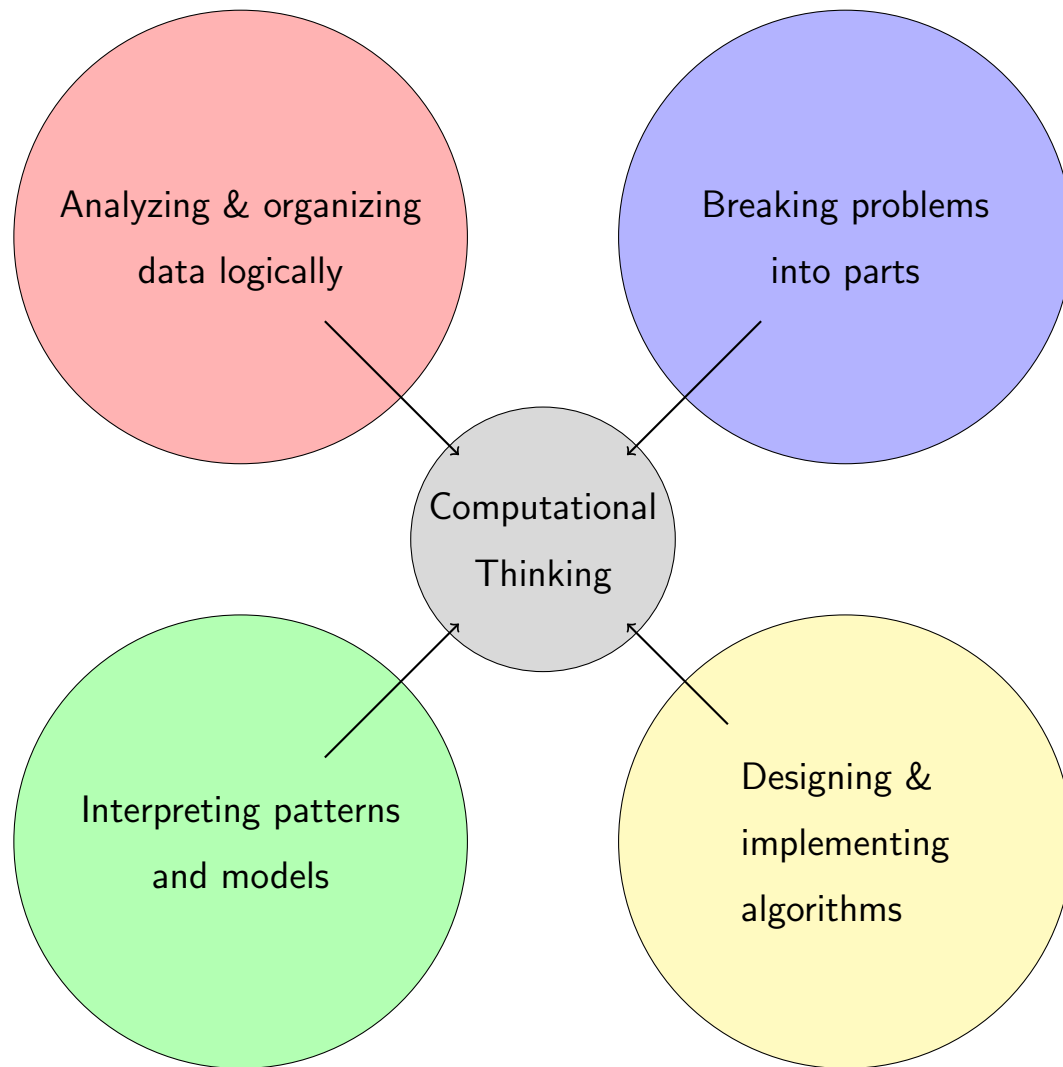
Breaking problems into simple parts
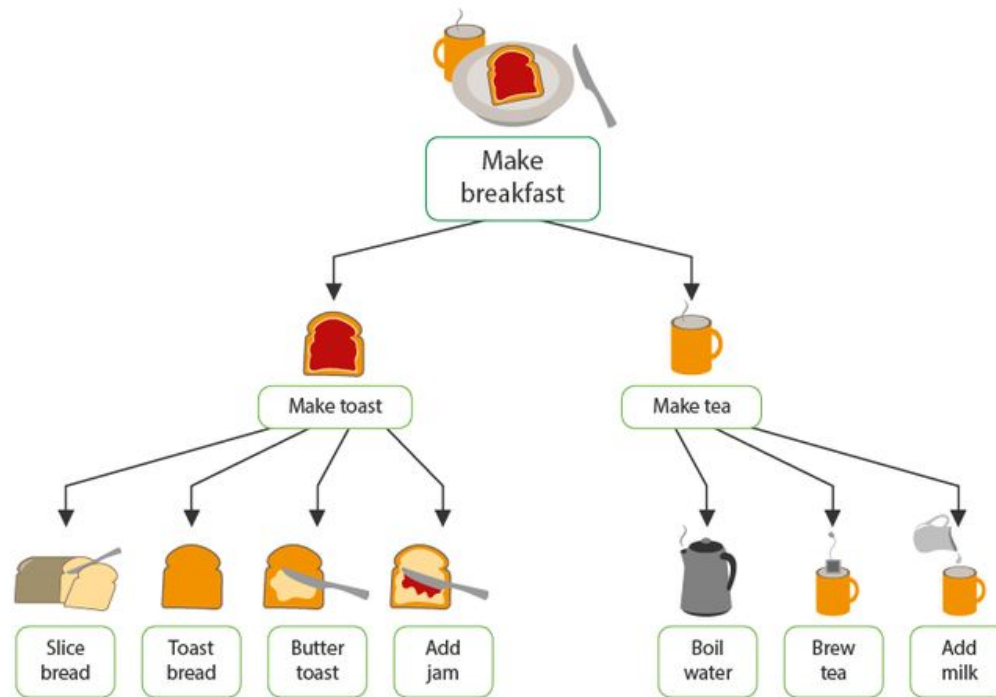
I have two orange fish.
I have three orange cats.
I have two orange chairs.

I have ＿＿＿ orange ＿＿＿.

Analyzing outcomes
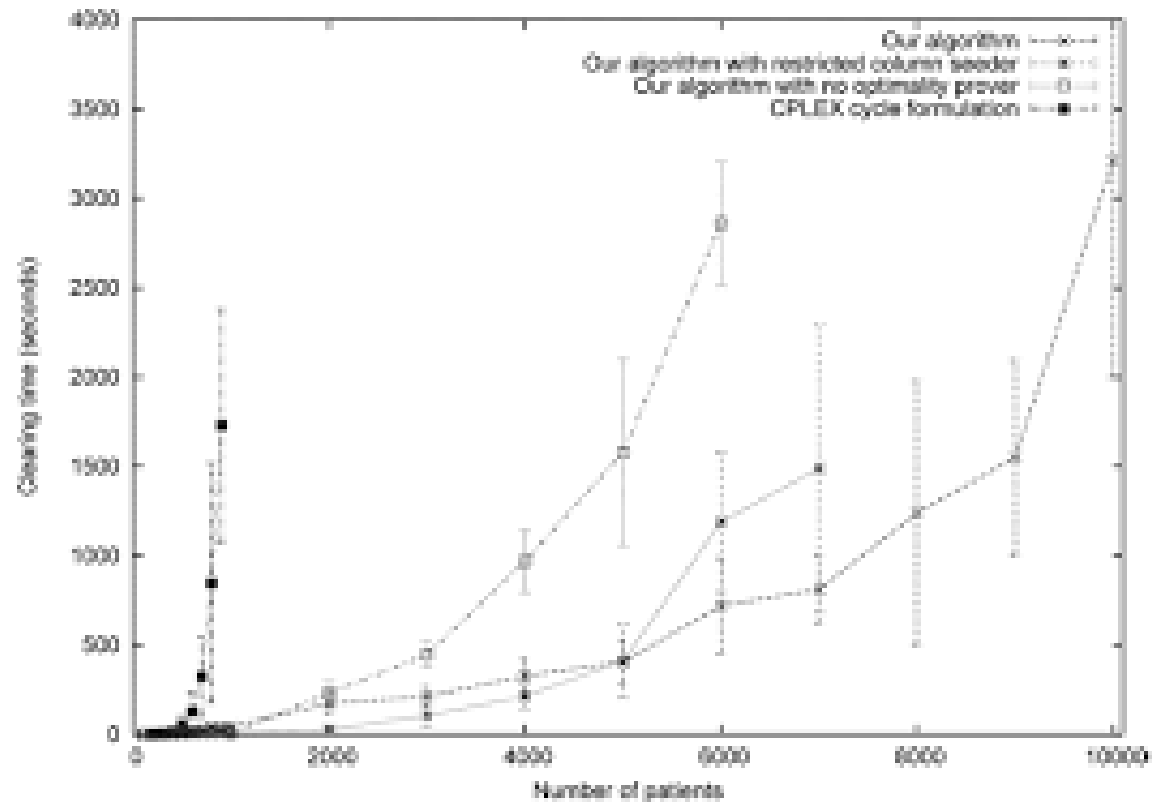
Which beaver is not dressed like the dress code?

big tail

yes — glasses
no — big tooth

glasses:
yes — blue trousers
no — green shirt

big tooth:
yes — blue trousers
no — blue hat

blue trousers (left):
yes — yellow hat
no — red hat

green shirt:
yes — brown trousers
no — blue hat

blue trousers (right):
yes — red hat
no — yellow hat

brown trousers:
yes — blue hat
no — yellow hat

Answer:

A   B   C   D
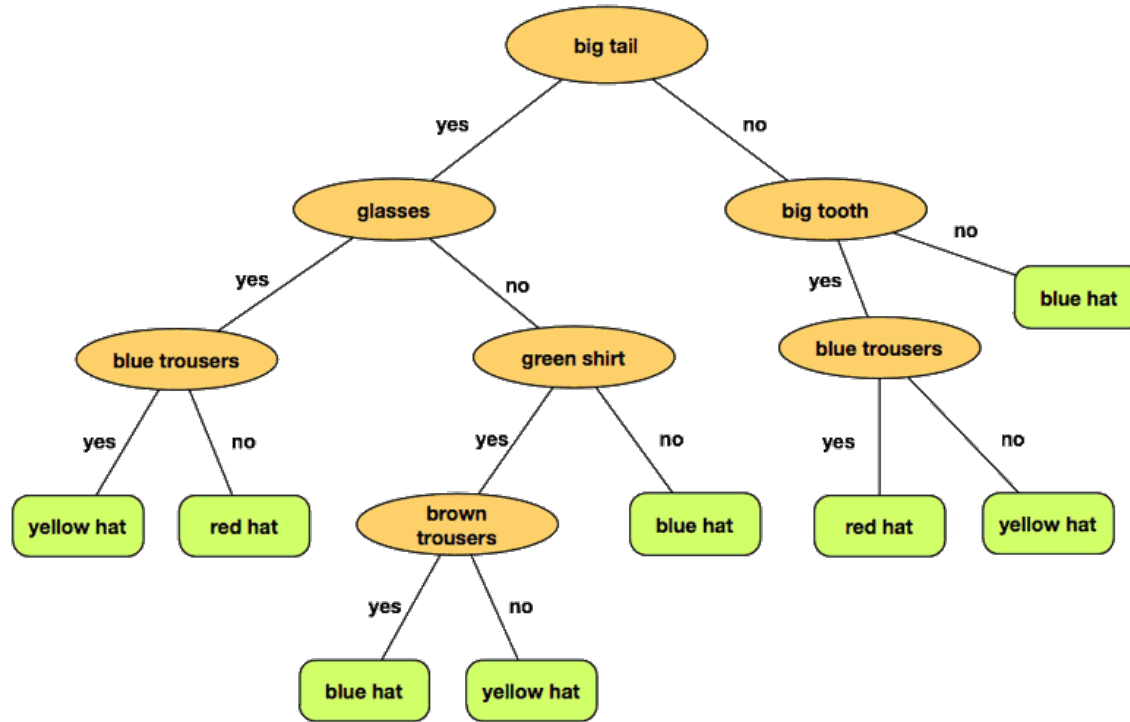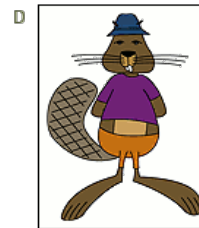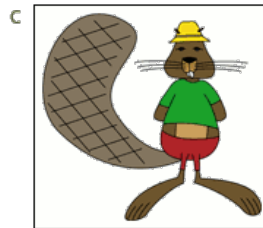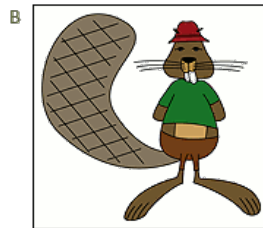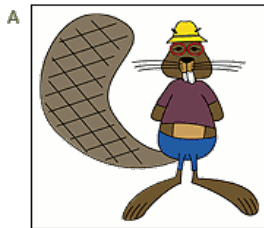
Designing algorithms

The Steps To Long Division

- <u>Divide</u>- the number inside the house with the number outside of the house
- <u>Multiply</u>- the number outside of the house by the number on top of the house
- <u>Subtract</u>- the number inside the house from the number below the inside number
- <u>Check</u>- to make sure the difference (the answer to a subtraction problem) is smaller that the number outside the house
- <u>Bring Down</u>- the next number in the dividend to add to the ones place in your sum
- <u>Repeat</u>- all the steps as many times as needed
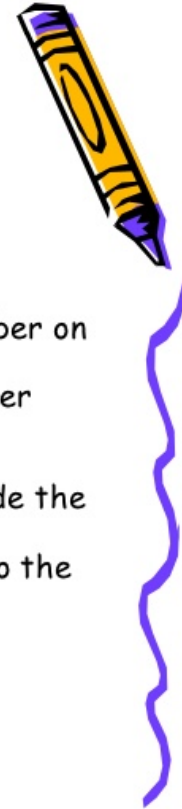
Computer Hardware vs Computer Software

Computer hardware is any physical device used in or with your computer. This includes your monitor, mouse, keyboard, hard drive, sound card, video card, etc.

Computer software is a general term used to describe a collection of computer programs that perform some task on a computer system. Computer software is generally divided into two classes – system software and application software. Operating systems, utilities, device drivers, programming languages, interpreters are examples of system software. Application software are a group of computer software that are created to help the user complete tasks such as creating documents, performing calculations, storing and managing data, playing videos, etc. This includes programs like Microsoft Word, Adobe Acrobat, etc.

Computer hardware operates under the control of software.

We are interested in the algorithms that are contained within the software.

"WHICH CAME FIRST:
THE HARDWARE OR THE SOFTWARE?"

©Marty Bucella   www.martybucella.com

"Before you take me, settle a bet. Are you
considered hardware or software?"

What is an algorithm?

"An algorithm is like a recipe, it must be followed exactly, it must be unambigious, and it must have an end."

This is a really tough case.
Raise your hand if you know what an "algorithm" is.
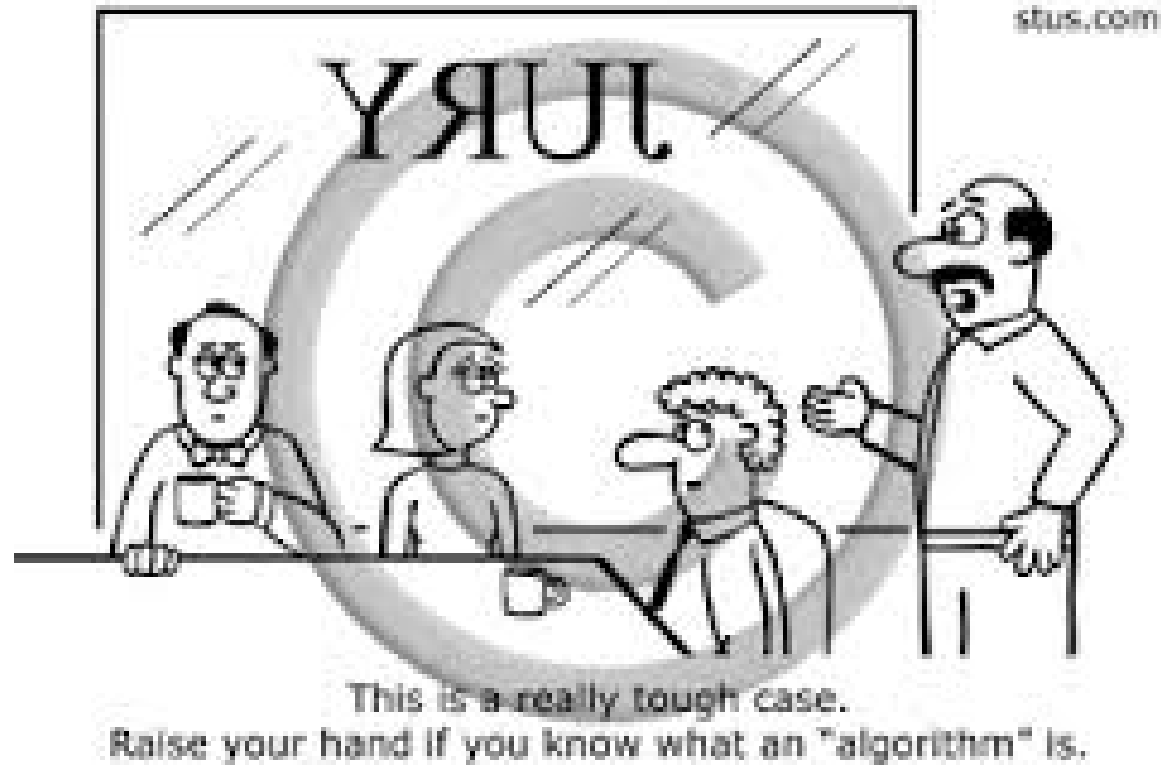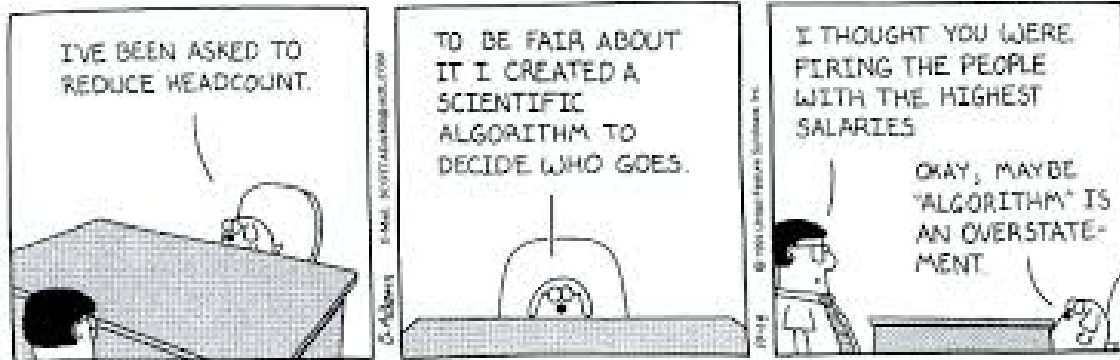
Dilbert by Scott Adams From the ClariNet electronic newspaper Redistribution prohibited info@clarinet.com

"On my summer vacation I wrote a computer algorithm. I sold it, for thirty million dollars, to a social networking website."

In an interesting 2006 article entitled The Algorithm: Idiom of Modern Science the author Bernard Chazelle, a Princeton professor, makes the following statement:

The Algorithm's coming-of-age as the new language of science promises to be the most disruptive scientific development since quantum mechanics.

and further in the article

Algorithms are often compared to recipes. As clichés go, a little shopworn perhaps, but remember: no metaphor that appeals to one's stomach can be truly bad. Furthermore, the literary analogy is spot-on. Algorithms should be understood as works of literature. The simplest ones are short vignettes that loop through a trivial algebraic calculation to paint fractals, those complex, pointillistic pictures much in vogue in the sci-fi movie industry. Just a few lines long, these computing zingers will print the transcendental digits of π, sort huge sets of numbers, model dynamical systems, or tell you on which day of the week your 150th birthday will fall . . . For the rest, we have, one notch up on the sophistication scale, the sonnets, ballads, and novellas of the algorithmic world. These marvels of ingenuity are the engines driving the

algorithmic revolution currently underway. (And, yes, you may be forgiven for thinking that a computer geek's idea of culinary heaven is a nice big bowl of alphabet soup.) At the rarefied end of the literary range, we find the lush, complex, multilayered novels. The Algorithmistas' pride and joy, they are the big, glorious tomes on the coffee table that everyone talks about but only the fearless read.

For us, we assume the following scenario.

INPUT → Algorithm → OUTPUT

No Man's Sky: A Game Crafted by Algorithms

The virtual universe in this program is generated by an algorithm, not by an artist's pen. The generated universe is, of course, not infinite but "if you were to visit one virtual planet every second then our own sun will have died before you have seen them all."

The most common example of this type of game is Minecraft which creates a unique world for each of its players, randomly arranging rocks and lakes from a limited palette of bricks whenever someone begins a new game. But No Man's Sky is far more complex and sophisticated because the algorithm contains rules for the physics, biology, etc. of each virtually generated planet. The tens of millions of planets that comprise the universe are all unique. Each is generated when a player discovers it, and is subject to the laws of its respective solar systems and vulnerable to natural erosion. The multitude of creatures that inhabit the universe dynamically breed and genetically mutate as time progresses.

A simple, but outdated, example of an algorithm

Consider the problem of looking up a name in the phone book. (I know that you have probably never had to do this in your life!)

If you are looking up, e.g., Mike Shephard
in the phone book then the following would be an algorithm for finding the phone number.

Step 1 Open the phone book to the first page of names, i.e., the ones beginning with "A";

Step 2 Check to see if the name Mike Shephard is on the page;

Step 3 If the desired name is not on the page then turn to the next page and go back to Step 2;

Step 4 If the desired name is on the page, write down the phone number and stop because you are done.

The way we have written this algorithm is called pseudo-code because it describes the necessary steps in an ordered and unambigious way without the use of any programming language.

Notice that we have a conditional, i.e., an if-then statement which allows us to determine if we are done or if we have to go to the next page of the phone book.

Note that we also have a repeated loop, i.e., we keep doing steps 2 & 3 until we are done.

Now let's look at how much work it takes to find the name Mike Shephard using this algorithm. Assume the phone book has 500 white pages and names beginning with "Sh" appear 81% into the book, i.e., on page 405. That means we have to look at 405 pages before we find the correct phone number! For obvious reasons, this is called a Brute-Force algorithm or the Exhaustive search algorithm.

If the phone book doubles in size the next year to 1000 pages and "Sh" still appears 81% into the book, we have to look at 810 pages before we find the phone number. If the book doubles again to 2000 pages then we have to look at 1,620 pages.

This means that our work (i.e., the number of pages we must scrutinize) doubles as the size of the book doubles. If we plot this, we get the straight line

$$y = 810 + .81(x - 1000)$$

where x is the number of pages in the phone book and y represents the number of pages we have to look at.

If we are looking for the phone number for Karen Allen then it will take less work using this algorithm but if we are looking for Eric Yeager then it will take more work.

In general, the amount of work to find a phone number is proportional to the number of pages in the phone book which means it is a constant times N.
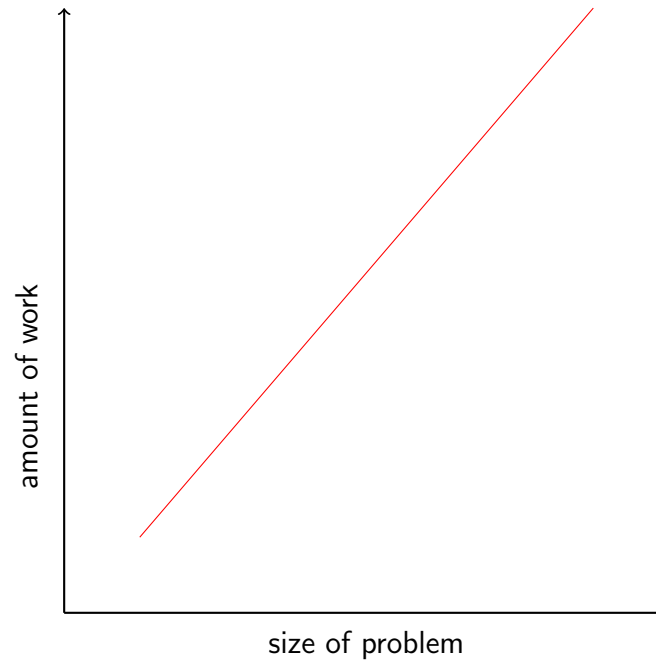
Brute-Force Algorithm

What can we do to reduce the amount of work?

Consider the following algorithm which intuitively seems to cut the work in half.

Step 1 Open the phone book to the first page of names, i.e., the ones beginning with "A";

Step 2 Check to see if the name Mike Shephard is on the page;

Step 3 If the desired name is not on the page then skip next page, go to next page and then go back to Step 2;

Step 4 If the desired name is on the page, write down the phone number and stop because you are done.

Does it always work?

Divide and Conquer Approach

Step 1 Open the phone book to the middle;

Step 2 Check to see if the name Mike Shephard is on the page;

Step 3 If the desired name is not on the page then

- if the names on the page begin with a letter combination before "Sh" then throw the first half of the book away and go back to Step 1;
- if the names on the page begin with a letter combination after "Sh" then throw the second half of the book away and go back to Step 1;

Step 4 If the desired name is on the page, write down the phone number and stop because you are done.

How much work does this algorithm take?

Assume again that Shepard appears 81% of the way into the book. If the book is 500 pages, then assume Shepard appears on page 405.

Step 1   discard → pages 1-250   pages 251-500 ← keep

Step 2   discard → pages 251-375   pages 376-500 ← keep

Step 3   keep → pages 376-438   pages 439-500 ← discard

Step 4   keep → pages 376-407   pages 408-438 ← discard
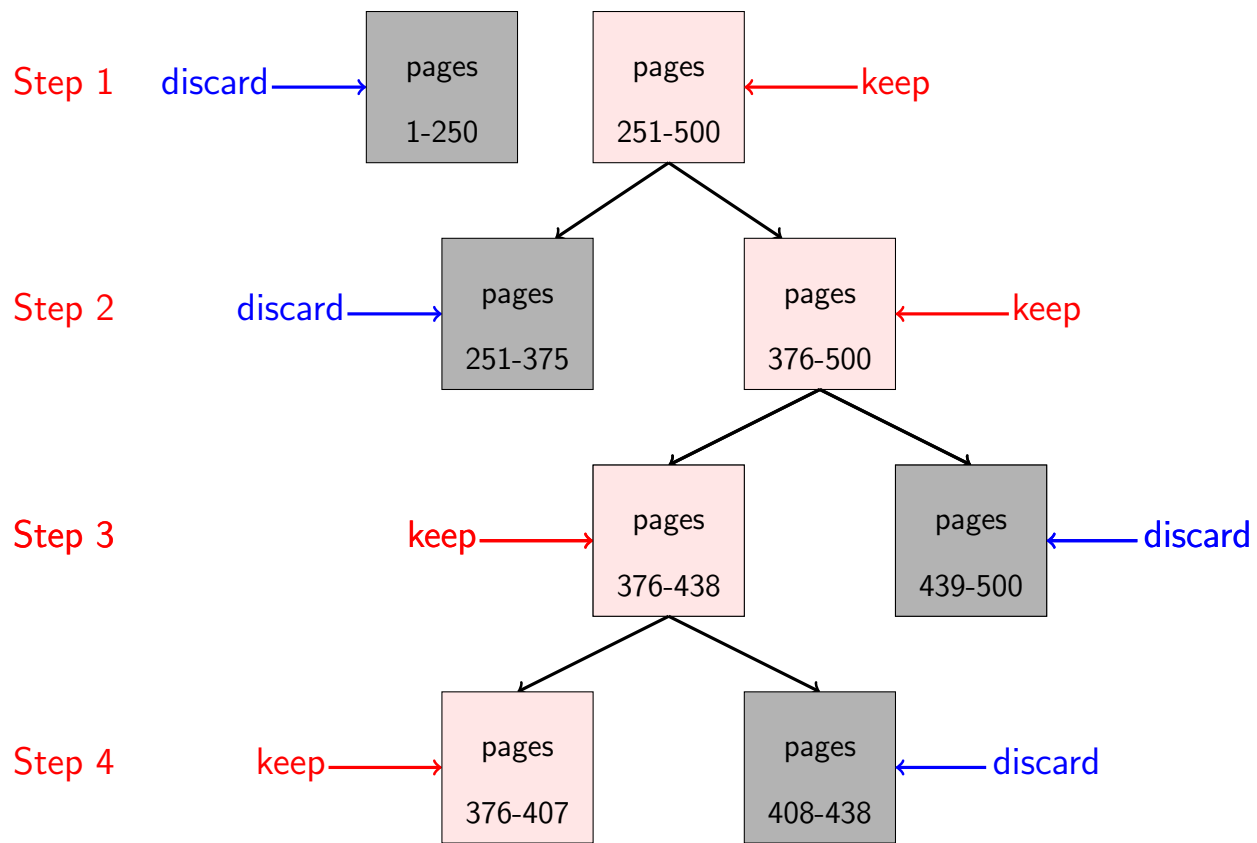
Continuing in this manner, we narrow the search to the following number of pages

<div align="center">

Step 5   376-391   392-407

Step 6   392-399   400-407

Step 7   400-403   404-407

Step 8   404-405   406-407

Step 9   404   405

</div>

Now what happens if the phone book doubles in size to 1000 pages and the phone number is on page 810? Remember before that the work doubled with the brute-force approach.

Step 1	1-500	501-1000
Step 2	501-750	751-1000
Step 3	751-876	877-1000
Step 4	751-813	814-876
Step 5	751- 782	783-813
Step 6	783- 798	799-813
Step 7	799-806	807-813
Step 8	807-810	811-813
Step 9	807- 808	809-810
Step 10	809	810

Step #	# number of pages to search		
	500 pages	1000 pages	2000 pages
1	250	500	1000
2	125	250	500
3	62	125	250
4	31	62	125
5	15	31	62
6	7	15	31
7	3	7	15
8	1	3	7
9		1	3
10			1

It turns out that the work required depends upon $\log N$ where N is the size of the phone book. Now it's not important whether you remember what a logarithm is, but it is important that you understand the following plot where we plot the amount of work as the phone book size increases for both the Brute-Force approach and the Divide and Conquer approach.
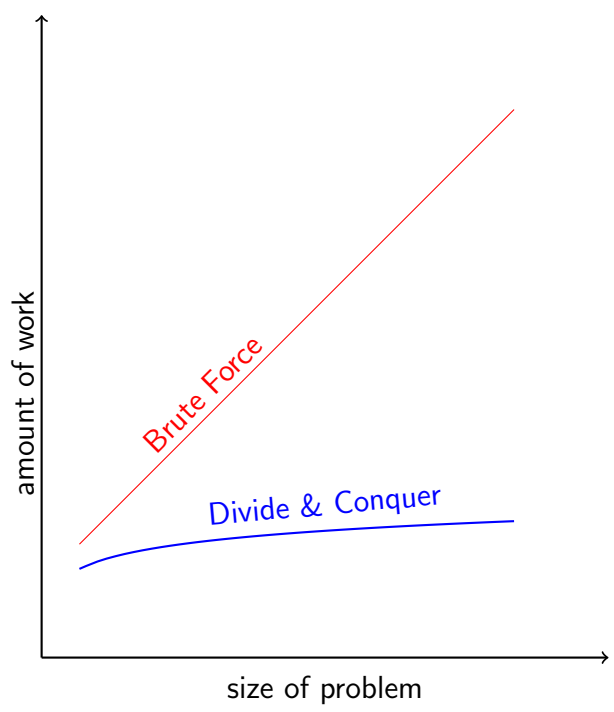
1. *Computational Thinking*, Jeannette Wing[2], 2006

2. *The Algorithm: Idiom of Modern Science*, Bernard Chazelle, introduction and section "Thinking algorithmically"

Both articles are posted on our Blackboard site.

Next time we will have a Socrative quiz so be sure and bring your phones!

Let's adjourn for some cookies and get to know each other!

---

[2]Former Chairman of Computer Science at Carnegie Mellon University and currently Corporate Vice President of Microsoft Research

Introduction to Computational Thinking - Continued

Goals for this lecture:

1. To look at different ways to design algorithms through a specific example

2. To see that it's not so easy to describe even a simple algorithm precisely

3. Introduction to Socrative through quizzes

4. Topics we plan to cover this semester

Last time we saw an example of a brute-force algorithm and a Divide & Conquer algorithm for finding a phone number in a telephone book.

As another example, consider an algorithm for counting; specifically we will determine the number of people in a room.

The brute-force approach is for me to point at each person and count 1,2,3,4,.... This approach requires me to go through all the numbers from 1 to N, where N is the number of people in the room.

A second, but slightly more efficient, brute-force approach is to count individuals by two, i.e., 2, 4, 6, 8, ... This approach takes half of the work of counting each person individually.

A smarter approach is the following algorithm.

Step 1 Each person in the room stands up and is assigned the number 1.

Step 2 Each standing person must pair up with another person standing in the room and add your respective numbers together.

Step 3 One of the two of you must sit down.

Step 4 If more than one person is standing, go to Step 2 and repeat. If only one person is standing then their number is the total number of people in the room.

Again, our algorithm has a conditional (an IF-THEN statement) and a repeated loop.

This algorithm does not require the room to have an even number of people in it as the following schematic illustrates.

The table below compares the number of people remaining after Step 3 is completed (when one member of each pair sits down) for different size classrooms. Note that as in the Divide & Conquer algorithm the work increases by one as the number of people is doubled.

Loop #	# people standing after each loop			
	class: 21	class: 42	class: 84	class: 168
1	11	21	42	84
2	6	11	21	42
3	3	6	11	21
4	2	3	6	11
5	1	2	3	6
6		1	2	3
7			1	2
8				1

Based upon the trend we see from this table we can predict how many steps it will

take for a classroom of size 336, 672, etc. and when we plot the points we get a curve that looks like the $\log$ plot we had before.

Class Size	Number of loops required
21	5
42	6
84	7
168	8
336	9
672	10
1344	11
2688	12

ADD PLOT

This example demonstrates that we may need to think in a different way to design more efficient algorithms.

In these algorithms, as well as the ones from last time, we had a conditional statement, i.e., an if-then statement. Here is Bill Gates (Microsoft) explaining these.

YouTube video   www.youtube.com/watch?v=m2Ux2Pnje6E

In these algorithms, as well as the ones from last time, we had a repeat loop. Here is Mark Zuckerberg (Facebook) explaining the importance of these.

YouTube video   `www.youtube.com/watch?v=mgooqyWMTxk`

Another issue that comes up after we have designed an algorithm is that we have to describe the algorithm in an orderly and unambiguous way. This is not as easy as it may seem.

As an example, consider the following video from an introductory computer science course at Harvard where the class is attempting to give three volunteers an algorithm for making a peanut butter and jelly sandwich. Note that the center participant has a different type of jelly container than the other two. This is to illustrate that we often have to take care of special cases when designing an algorithm.

16:09 / 1:03:59

Go to `b.socrative.com` and you will see a login page.

Under `Student Login` enter the Room Name IUZGAZ34E.

The next page has you enter YOUR NAME (not a student ID) in the given format
Last, First

The first question on the correct quiz should appear. If the question is True/False then you simply click the correct box.

1. The term "algorithm" is only used to refer to computer programs.

2. Your computer's trackpad is an example of software.

3. An example of a brute-force algorithm for counting the number of people in a room is counting the individuals by twos.

4. If a phone book has 1000 pages then the Divide & Conquer algorithm for finding a number in this phone book takes twice as much work as for a phone book with 500 pages.

5. Jeannette Wing believes that computational thinking should only be taught in college.

6. Using pseudo-code to describe an algorithm means that you write the algorithm using syntax from a programming language.

Suppose you have a recipe for hush pup-
pies and you want to tell a person in China
how to make this dish but they don't
speak English. The only want to achieve
this is to write the recipe in English and
have an interpreter translate it into Chi-
nese. In this analogy the recipe is the algorithm, you are the programmer and the
person in China plays the role of the computer. You might be able to write the recipe
but the "computer" can't understand it!

This analogy is like the programmer/computer situation. The programmer uses a
programming language such as C++, Java, Fortran, etc.. to write the software.
Unfortunately, the computer doesn't understand these languages because all it under-
stands is machine language. To complicate matters, programmers don't understand
machine language.

To solve this problem we need the equivalent of an interpreter which is what the
compiler does. It converts our computer program written in a programming language

that we can understand into one that a computer understands, basically into binary which is a language that only consists of ones and zeros (we will talk about this later).

Why can't we write our computer programs in English? Our language is too imprecise for computers!

We will NOT learn a programming language in this course but rather concentrate on the ideas behind the algorithms which a programmer implements.

Software Bugs

Oftentimes one hears the term bug when referring to software.

A software bug is an error or flaw in a computer program that causes the result to be incorrect or to behave in unintended ways. These errors typically occur due to mistakes made by humans either in the actual program or in the algorithm design.

A study commissioned by the US Department of Commerce's National Institute of Standards and Technology in 2002 concluded that "software bugs, or errors, are so prevalent and so detrimental that they cost the US economy an estimated $59 billion annually, or about 0.6 percent of the gross domestic product".

Where did the term "bug" come from?

Grace Hopper, a computer scientist and Navy Rear Admiral, joined the Harvard Faculty at the Computation Laboratory after WWII to continue her work on the Mark II computer. Operators traced an error in the Mark II to a moth trapped in a relay, coining the term bug. This bug was carefully removed and taped to the log book. Stemming from the first bug, today we call errors or glitches in a program a bug.

Typos in a program are usually caught by a compiler . Computer bugs often occur when the programmer makes a logic mistake or fails to take into account special cases (as in the PB&J video with the differences in the jam containers).

Topics to be covered this semester

Part I – Google Ngram Viewer

- One week, Burkardt

- In 2004, Google announced a plan to systematically scan a copy of every book ever published, estimated to number 130 million. The result would be a huge database called Google Books.

- The idea was not to just copy all the publications but to make a searchable data base. To allow everyone easy access to this data base, Google developed a viewer which allows the user to compare the number of occurrences of words or phrases in print through the years.

- We will be looking at some of the history of digitizing text, the problems Google encountered with this project, the type of computer algorithms needed to convert text into a searchable document and how to glean information from the Ngram viewer.
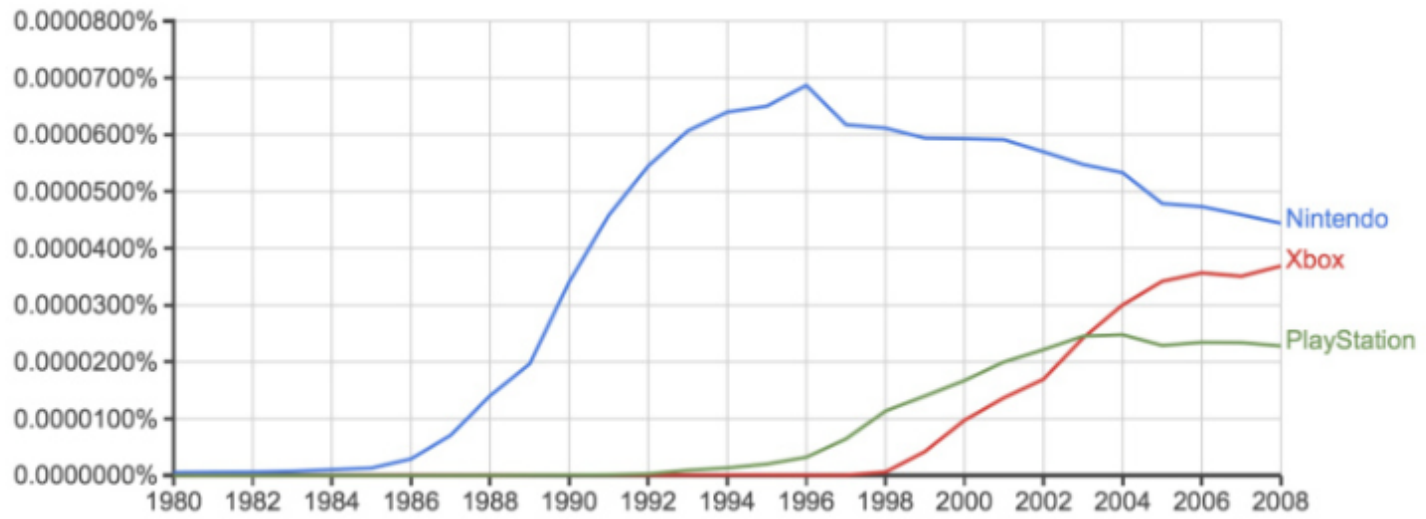
Part II – Page Match and Page Rank Algorithms

- Two weeks, Burkardt

- Algorithm # 2 (Page Match) and Algorithm # 3 (Page Rank) in text

- When you do a search on the Web then not only does the search engine find information that matches your search words but it also ranks them, i.e., it puts the most relevant matches on the first page. Moreover, it does this in seconds not hours.

- Just finding matches seems like an impossible task because it is estimated that there are over a billion websites. What we will see is that when you do a search, the search engine doesn't really search the Web but rather an index of the Web which it has created earlier.

- Ranking the sites which match the search words seems like another impossible task because a computer doesn't really understand what is written on the website. How can it distinguish between spam websites and ones which have meaningful information? How can it decide which are the most relevant sites? What can webmasters do to help their sites appear on the first page of results? These are some of the questions that we will look at.

Newton's - Kepler's E-mail to Alfred Nobel Prize winner Physicists

There are no physicists after us only "University" Idiots like you

Space time - Relativistic - Quantum - Strings mechanics is not physics

Real time Universal Mechanics

Newton's - Kepler's equations solved wrong for 350 years

1001 new real time physics formulas

Changing physics and the History of physics

Annexing quantum mechanics to classical mechanics and deleting relativity and strings

It is the math formulas that matches a physics experiment results

Real time Newton's - Kepler's mechanics

Professor Joe Nahhas July 4th 1973

joenahhas1958@yahoo.com

Read my Lips: I Joe Nahhas (lucid) will end Alfred Nobel Mafia of Physics and physicists' stupidity.

There is one and only one Mechanics

Real time universal mechanics

Real time mechanics is the natural law of past present and future of mechanics.

For 400 years Newton's - Kepler's were formulated and solved wrong.

The new real time solution of Newton's Kepler's equations deletes modern physics which is based on relativistic quantum string time travel mechanics and matches experiments with unprecedented accuracy to change physics and the history of physics in its entirety.

Modern Physics is based on relativistic quantum string time travel mechanics. Time is caveman and modern man scale of convenience and is not Alfred Nobel dimension for time travel regardless of what all Alfred Nobel time travel "Physicists" have to say about it. Time travel is not physics and accepting time travel as physics in classrooms and using it in scientific

Part III – Cryptography, Public Key Encryption & Digital Signatures

- 3 $\frac{1}{2}$ weeks, Peterson

- Algorithm # 4 (Public Key Encryption) and Algorithm # 9 (Digital Signatures) in text

- We will begin by looking at cryptography from a historical perspective. We will see how it has been used to send secret messages as early as Julius Caesar (who has a cipher named for him) up through World War II. You will learn how to encrypt/decrypt a message given a key and how to decrypt certain types of ciphers without a key by looking at letter and word frequency plots.

- Cryptography became critical for everyone, not just military leaders, when computers became prevalent. Now we enter our credit card information in a website to make purchases with confidence that the information is secure. How does this work? Public Key Encryption is the algorithm that makes this possible. We will strive to understand this algorithm by looking at simple examples.

- Another application of encryption is when you sign a document electronically. How can the individual receiving your signed document know that it was really you who signed it?

GHIHQG WJH HDVW ZDOO RI WKH FDVWOH

G H I H Q G   W J H   H D V W

↓ ↓ ↓ ↓ ↓ ↓   ↓ ↓ ↓   ↓ ↓ ↓ ↓

D E F E N D   T H E   E A S T

DEFEND THE EAST WALL OF THE CASTLE

Part IV – Doublets or Can a Computer Solve a Word Puzzle?

- One week, Burkardt

- Lewis Carroll (Alice in Wonderland) developed a word game called Doublets. For example, can you take the word MAN and turn it into APE by changing only one letter at a time. Moreover, we would like to do this in the shortest number of moves possible.

- We will look at strategies for solving this puzzle and see how they can be implemented in an algorithm for a computer to solve the puzzle.

- This will give you an insight into algorithms for game solving such as chess. In 1997 IBM's chess computer beat the top-rated chess player in the world and over the next few years humans and computers traded blows until eventually by 2005-6, computer chess programs were in the lead.
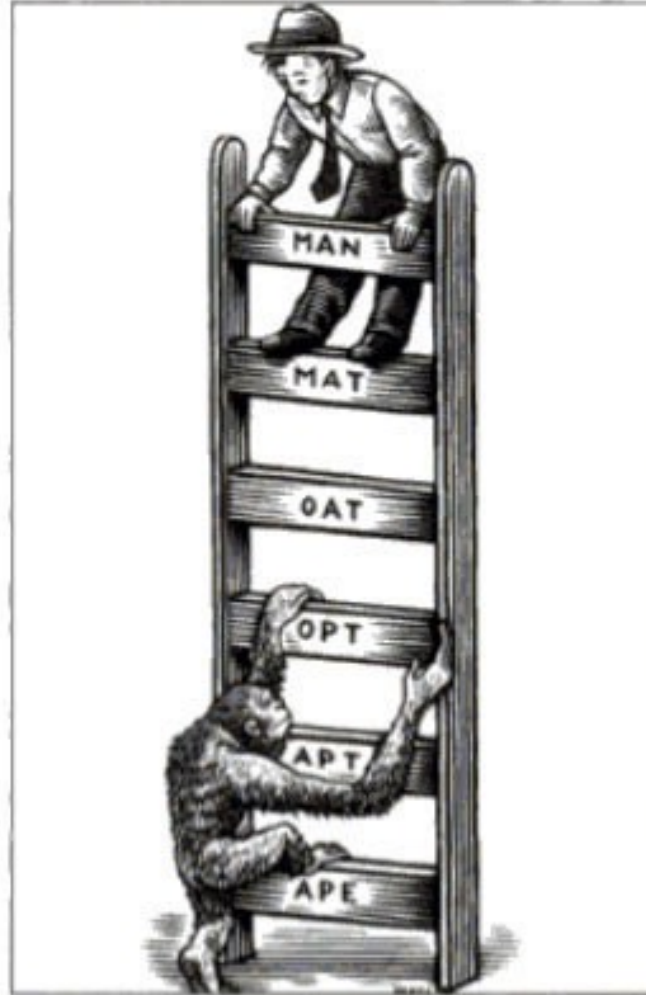
Illustration by Gregory Nemec

Part V – Machine Learning & Pattern Recognition

- $2\frac{1}{2}$ weeks, Peterson

- Algorithm $\#$ 6 in text

- Since the advent of computers, people have considered whether computers can become the dominant form of intelligence on Earth; this has been a theme in many science fiction stories/movies through the decades. There is actually a test called the Turing test (1950) which is used to measure a machine's ability to exhibit intelligent behavior.

- Machine learning algorithms are used in many applications from reading postal codes to autonomous land vehicles. Unlike other algorithms we have considered, in machine learning we typically "train" the algorithm with a set of data and then, after it has "learned", we input new data for it to identify. For example, when reading hand written postal codes we could train the algorithm with a set of handwritten numbers and then give it a new code to identify.

- Recognizing patterns is something that humans do better than computers. Typically a human can identify an image of a person after having seen another picture of the person but this ability is difficult to incorporate into an algorithm.

- We will look at various types of Machine learning algorithms and see the types of applications they are best suited for. An underlying theme is there is not one algorithm which works for all problems.
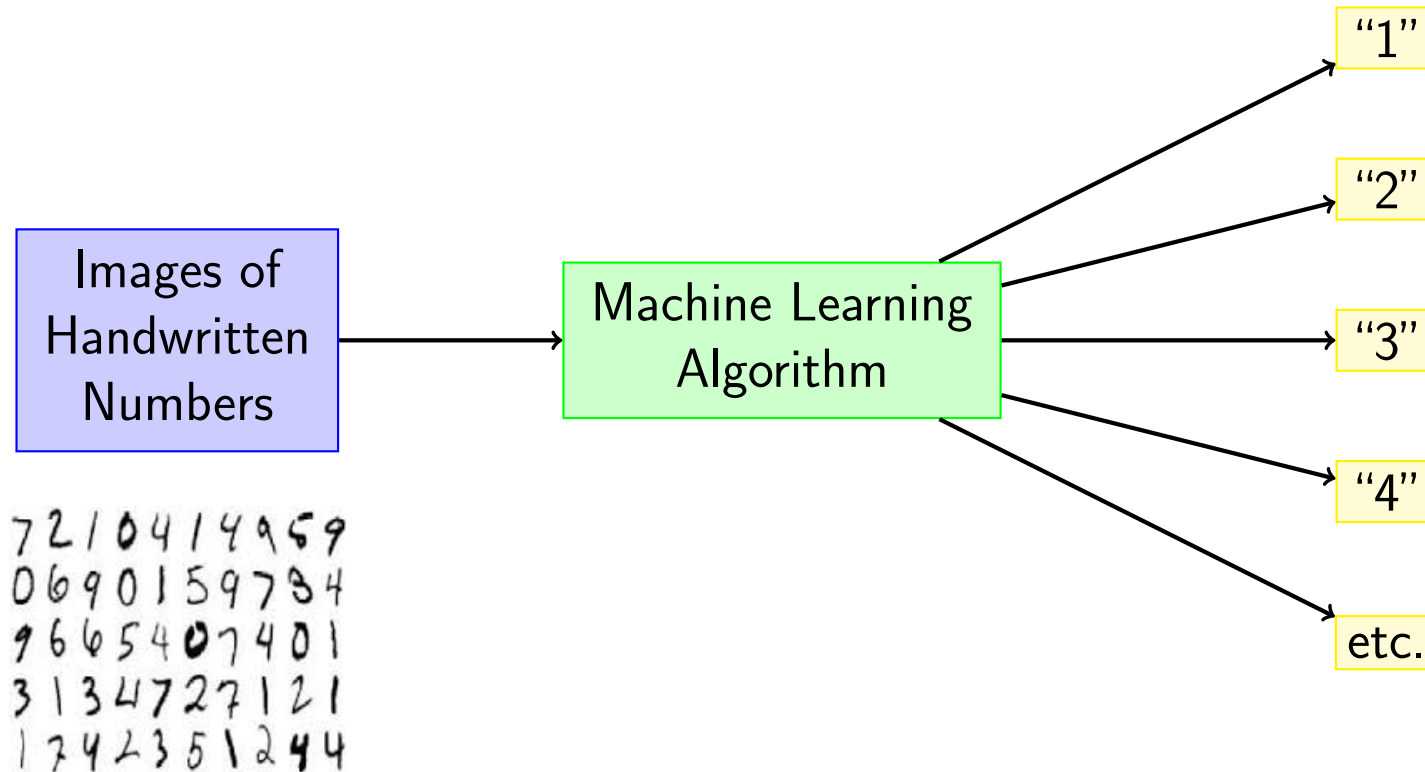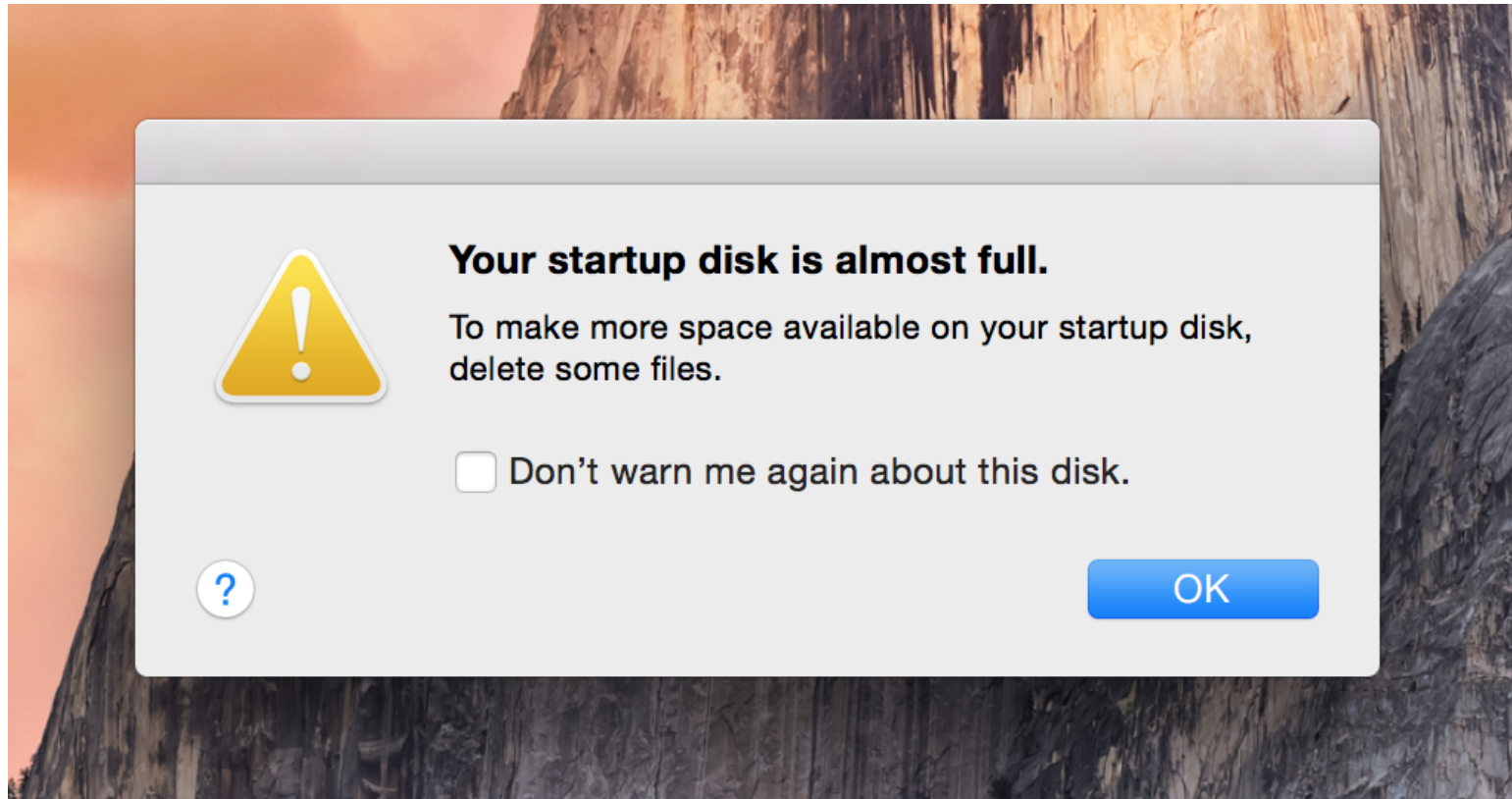

Training an algorithm to read a zip code

Part VI – Clustering & Data Compression

- $1\frac{1}{2}$ weeks, Burkardt

- Algorithm # 7 in text

- Clustering is the process of organizing objects or people into groups. You could say that your friends on Facebook is a cluster and my friends are another cluster. Your neighbors form one cluster and my neighbors form another. Clustering is used in many applications without us even realizing it. For example, if we compare a photo on our computer with one that we print, we see that they are not exactly the same because the monitor has many more colors available than most printers. How did the printer assign a color to a pixel?

- Of course to develop an algorithm to cluster objects requires having criteria for forming the clusters such as distance, color, etc. We will look at various types of clustering and applications.

- Big data is the new buzz word for this century. Jobs requiring big data expertise are predicted to be numerous and carry high salaries.

- Amazon gets 35 orders per second or 1.1 billion per year. That is a lot of data! In order to handle this much data it must be compressed in such a way that it can

be decompressed at a later date. We want to look at some of the ideas behind data compression.

Your startup disk is almost full.

To make more space available on your startup disk, delete some files.

Don't warn me again about this disk.

OK

In the remaining two weeks of class we plan to do two different things.

1. Pencil is an online editor that lets you get a feel for programming by working in blocks and texts. It is not a programming language but a fun tool to try out some graphics, games, etc.

2. Brian Hayes writes a column for the *American Scientist* publication and his articles are accessible to the general population. We will explore a couple of these essays.
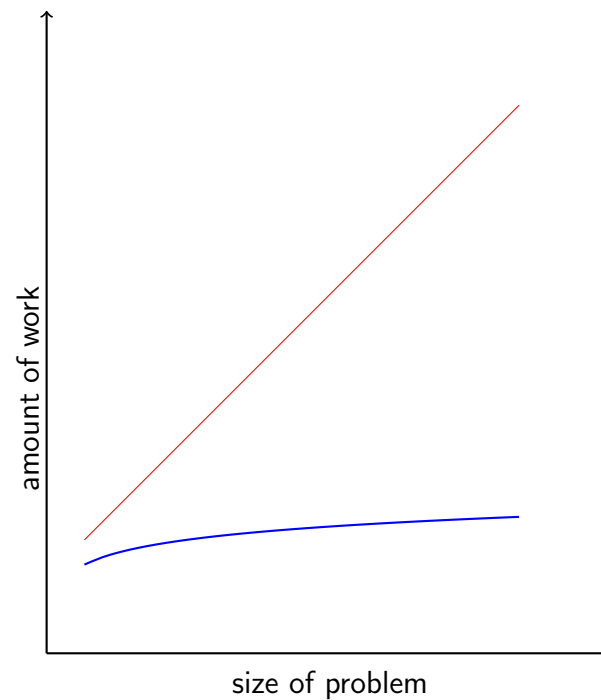
Reading Assignment

Computational thinking, 10 years later, Jeannette Wing, Microsoft Research Blog, 2016

Answer T for true and F for false.

1. In the graph of the work required to find a name in a phone book, the red line is for the Divide and Conquer algorithm and the blue curve is for the brute-force

algorithm.

2. In the graph of the work required to find a name in a phone book, the algorithm described by the blue curve requires less work for a give phone book than the one described by the red curve.

3. Describing an algorithm using pseudo-code means to write it in plain English, in an orderly and unambigious way.

4. At each step of a Divide & Conquer algorithm you break the problem into smaller pieces.

5. In Jeannette Wing's article on Computational Thinking she advocates teaching computational thinking in K-12 grades.

6. In programming lingo a conditional statement means that you have an If - then statement.

7. The two most common components of an algorithm are repeated loops and conditionals.

8. Computer algorithms are better at pattern recognition than humans.

9. Computer algorithms are better at repeated tasks than humans.

10. Microsoft Word is an example of computer hardware.