


ANNOUNCEMENTS

- Lab 6 started in lab this week
- Next week: No new assignment – time to finish up any remaining issues with Labs 4, 5, 6.
 - All three are due at the same time, but starting them all the day before they are due is NOT recommended
 - They are due November 28th
 - This is the Sunday at the end of Thanksgiving break – don't count on anyone being available for help Wednesday through Sunday.

ANNOUNCEMENTS

- Exam 3 – November 15th (in lecture)
 - We will discuss more about the exam and go over some topics on Friday in lecture.


 - Monday, November 22nd – lectures will not meet. Instead, I will be holding office hours all day to assist with any last-minute issues with the Scratch assignments. If you are coming for help with Scratch, you must bring your files with you. I don't have access to your UB filespace from my office.
- 

PROGRAMMING WITH SCRATCH

- <http://scratch.mit.edu>

 - You can find resources for Scratch on this site.

 - You can download Scratch from this site.
 - It is free
 - Available for Windows and Mac

 - If you have your own computer, I would recommend downloading so you can work on Scratch in preparation for Exam 3.
- 

PROCESS

- Phase 1: Understand the problem
- Phase 2: Get an idea of how an algorithmic procedure might solve the problem.
- Phase 3: Formulate the algorithm and represent it as a program.
- Phase 4: Evaluate the program for accuracy and for its potential as a tool for solving other problems.



PHASE 1

- In a class/lecture:

● **READ** the problem/assignment

- Ask questions...



PHASE 1

- In the real world:
 - Talking to the customer
 - Asking questions
 - Observing Behaviors

- This can take weeks or months



PHASE 2 – “BRAINSTORMING”

- Formal term: Design of software/software systems



PHASE 3

○ Algorithm

- An ordered set of unambiguous, executable steps that defines a terminating process.




PHASE 3 - ALGORITHM

- An ordered set ... of steps
 - Steps are performed in a particular order.
 - There is a step 1, step 2, etc.




PHASE 3 - ALGORITHM

- Unambiguous steps
 - At any point, it must be clear exactly what we currently have and how we move to the next step.
 - No creativity
 - No conjecture
 - No guessing
- 

PHASE 3 - ALGORITHM

- Executable steps
 - “Doable”

 - Make a list of all the positive integers
- 

PHASE 3 - ALGORITHM

- Terminating process
 - It must end




PHASE 3

- “Represent it as a program”
 - Translate the steps into a language the computer will understand



PHASE 3

○ Böhm-Jacopini Theorem


- Three elements are needed to combine any basic set of instructions into more complex ones
 - Sequencing – there is an order in which the elements of a program are executed
 - Selection - choice
 - Repetition – doing something over and over
- 

PHASE 3

○ Variables

- Used to store information

○ Procedures

- Not in Scratch
 - Way to organize and name certain bits of code
- 

PHASE 4

- Does it work?
- How do we know it works?

- Formal: Program Verification (Mathematical)

- Typical: Testing



PROGRAMMING LANGUAGES

- Syntax
 - Grammar

- The syntax of a program is checked before it is run.



SYNTAX CHECKING

Compilers

- Take the program code, check for grammar, translate into machine language.
- The program is run at another time.

Interpreters

- Take the program code, checks for grammar, and then runs the program immediately and sometimes interactively.




SEMANTICS


- Meaning



PROGRAMMING LANGUAGES

- First generation
 - Machine languages
 - Second generation
 - Assembly languages (hardware level)
 - Mnemonics to stand in for op-codes
 - Third generation
 - Closer to natural language
- 

THIRD GENERATION PARADIGMS


- Imperative
 - Programs are developed as a sequence of instructions.
 - Declarative
 - Programmer describes the problem rather than the solution.
 - The system derives the solution.
- 

THIRD GENERATION PARADIGMS


○ Functional

- Views programs as a set of functions (mathematical sense).

○ Object-Oriented

- Programs made up of a group of units (objects) that interact to solve a problem
- 


BACK TO SOME ALGORITHMS...

- Suppose you have a group of files on your computer and you want to know if a file named “StudyGuide.docx” is there. How would the computer/you do this?
- 


SEARCHING

- Assumption: The files are actually a list.

Dog.png
winword.exe
StudyGuide.docx
Chat20101110.log
Nov8-12withBlanks.pdf
Exam2.docx
project2.sb
helpfiles.txt

- Is the file we are looking for in the list? How do we know?
- 

LINEAR SEARCH ALGORITHM


- If <you are not at the end of the list>
 - Look at the first element in the list
 - If <it is the element you seek>
 - declare success
 - Else
 - repeat using the next element in the list as the starting point
 - Else
 - declare failure.
- 

WHAT ABOUT THIS LIST?

Chat20101110.log
Dog.png
Exam2.docx
helpfiles.txt
Nov8-12withBlanks.pdf
project2.sb
StudyGuide.docx
winword.exe

- Does our algorithm change?
- 

BINARY SEARCH ALGORITHM

- If <List is not empty>
 - Select “middle” entry of the list
 - If <middle is what we are looking for>
 - Declare success
 - If <what we are looking for comes before middle>
 - Do this again with the first half of the list
 - If <what we are looking for comes after middle>
 - Do this again with the second half of the list
 - Else
 - Declare failure
- 

DIFFERENCES?

- Linear search can be performed on any list
- Binary search must be performed on a sorted list
- Binary search is faster than linear search on a sorted list in most cases.



HOW DO WE SORT?

- Insertion Sort Algorithm
- Suppose this list

Dog.png
winword.exe
StudyGuide.docx
Chat20101110.log
Nov8-12withBlanks.pdf



INSERTION SORT

- We will sort lists of increasing size
- First, sort a list of size 1.

Dog.png
winword.exe
StudyGuide.docx
Chat20101110.log
Nov8-12withBlanks.pdf

list of size 1



INSERTION SORT

- Now a list of size 2

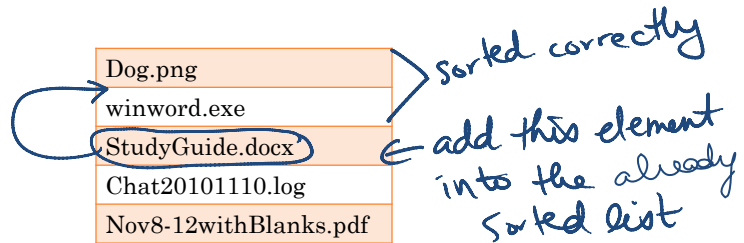
Dog.png
winword.exe
StudyGuide.docx
Chat20101110.log
Nov8-12withBlanks.pdf

sorted
add this element to
the already sorted
list



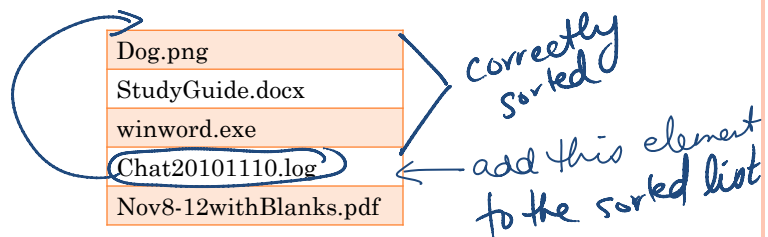
INSERTION SORT

- Now a list of size 3



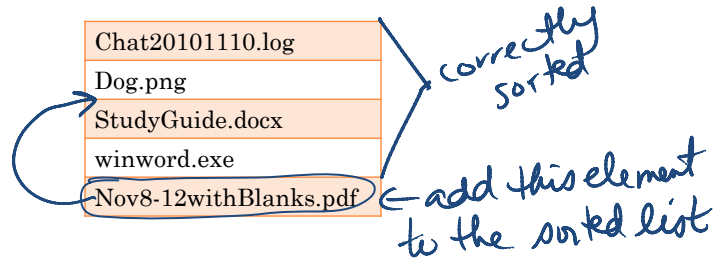
INSERTION SORT

- Now a list of size 4



INSERTION SORT

- Now a list of size 5



INSERTION SORT

- Answer

