Comp 120: Programming Abstractions and Methods

Practice for midterm exam

To-do list

- Respondus Practice – we are using this browser for Midterm exam
- Revel Programming: chapter 15 project 5
  - Due today
- PSA3
  - Due Thursday, October 1st
- PSA4:
  - Due on Tuesday, October 8
  - Posted already so that you can start early.
  - The topics of the programs for the PSA are GUI programming, and recursion, so working on them before the exam would be helpful (although not required).
- Midterm Exam:
  - In class on Thursday, October 1st for the entire class period.
  - Closed book, closed note, closed electronics.
  - Multiple choice, multiple answer and short answer questions.
  - You will take the test using the Responsus Lookdown browser on Blackboard.
  - You should have take the practice quiz using this browser to make sure you are all set up

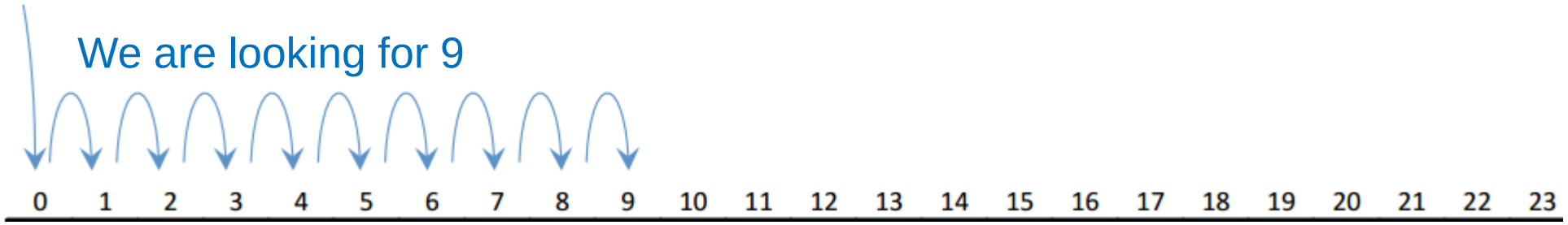
Practices for the Midterm

Comp 120 – fall 2020

Directory Search

- Write a function called dir_search to return a list of paths to a specified filename within a specified directory.
- For example, if a particular directory called test has three copies of a file named f1.txt in it, one directly in test, one in a subdirectory called d2, and one in a subdirectory of a subdirectory, d1/d4, then the function call dir_search(“test”, “f1.txt”) should return [“test/f1.txt”, “test/d2/f1.txt”, “test/d1/d4/f1.txt”]
- The ”os” module has a function called listdir that returns a list of files and subdirectories in a directory.
- The “os.path” module has a function called isfile that return True if the parameter is a file (and not a directory).

Search algorithm: Linear Search



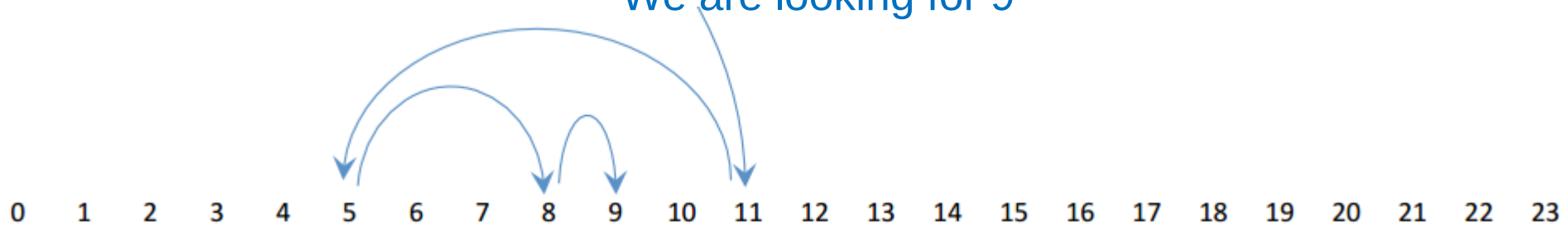
- Algorithm: we search linearly from top to bottom and check each value.

```
j = 0
While I haven't found the value
{
    Select next value from the search area  j = j + 1
}
```

- If we have 100 values in the search area, how many tries will this algorithm need?
  - Best case: *1* try
  - Average case: ½ n tries.
  - Worst case: *n* tries

Search algorithm: Binary search

We are looking for 9



- Algorithm:

```
SearchArea = The entire value pool (1-100)
     While SearchArea is not empty and I haven't found target
     {
          Pick the middle value v in the SearchArea
          If v == target, it is done!
          If v < target,
               SearchArea = first ½ of the current SearchArea
          If v > target,
               SearchArea = second ½ of the current SearchArea
     }
```

- If we have 100 values in the search area, how many comparisons will this algorithm need?
  - Best case: 1 comparison
  - Average case: $(\log_2 n)/2$
  - Worst case: $\log_2 n$

Your computer will guess it right with at most 7 tries because $\log_2 100$=6.643856

$\log_2$ (300000000) = 28.16038726

Values in the searching pool must be sorted.

Summary on algorithms

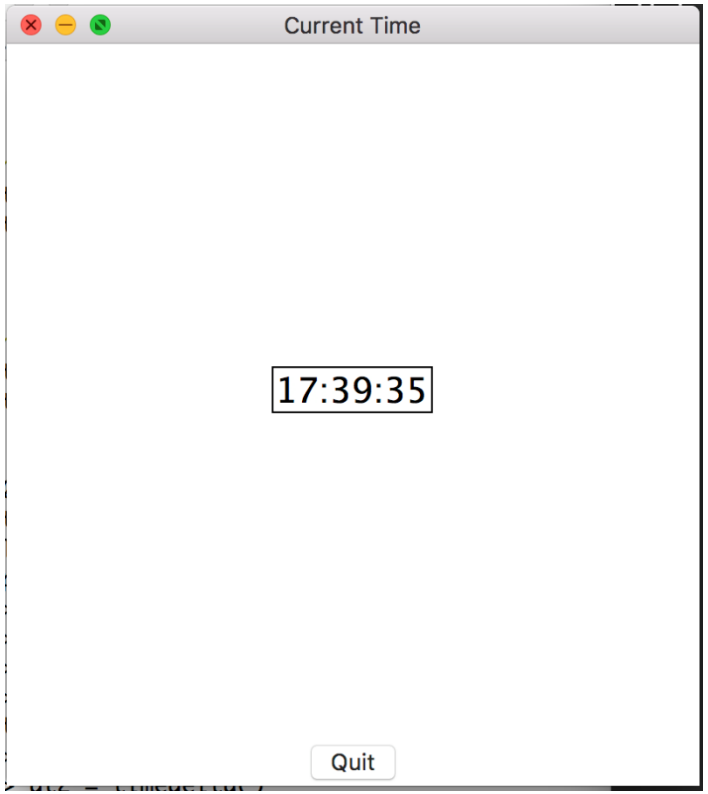
- Algorithm:
  1. An algorithm is a set of steps to solve a problem.
  2. An algorithm works on input data.
  3. An algorithm works out a result.
  4. The same problem can be solved by different algorithms.
  5. The quality of an algorithm can be judged by its accuracy and efficiency .

Exception handling: open a file

- Write a function that reads a file and returns a list containing the averages of the numbers on each line.
  - Numbers on a line are separated by spaces.
  - Lines that are empty or have invalid entries on them are ignored. The parameter to the function should be the name of the file.
  - If the file cannot be opened, return None.

GUI example

- Here is a screenshot of a program that simply displays the current time, with a quit button.



GUI example, continued

- We will review the code for this example.
- Now, modify it so that the window continuously displays the current time, once the start button is pressed, and stops updating the time when the stop button is pressed.
- Also displays hundredths of a second