



CS230: Data Structures

Spring 2007

Monday, January 29, 2007
Handout #2

Reading:
Problem Set:

Downey: Chapters 1, 2, 3, Sections 4.1-4.7, 4.13, 4.15-4.17, Chapters 5 and 6
Problem Set #1 due Tuesday, February 13

1 - 1



Objectives of CS230

- Teach main ideas of programming
 - Data abstraction
 - Modularity
 - Performance analysis
 - Standard abstract data structures (ADTs)
- Make you a more competent programmer
 - Designer, tester, analyzer, debugger
- Help you develop a project worth showing off
- Have fun in the process

1 - 2



Why ADTs?

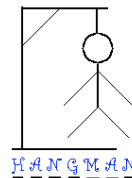
- Allow you to write complex programs easier
 - To keep mental track of complex data interaction
 - To reuse code (yes!)
 - To improve code performance
- Allows modularity of large projects
 - Easier to understand large chunks of code
 - Easier to collaborate with large teams
- To combat Sergeant Spaghetti Code

1-3



For example...

- Create a hangman program
- Here is the GUI



- Need a way of managing words
 - Create a new set of words
 - Add a new word
 - Delete a used word
 - Check if the set is empty
 - Grab a random word from the bag

1-4



WordSet.java

```
/** methods to create and manipulate a set of words */
public class WordSet {

    /** constructs an empty WordSet that can store any number of words */
    public WordSet ();

    /** returns true if the WordSet contains no words */
    public boolean isEmpty();

    /** adds a word to the WordSet, if it is not already there */
    public void addWord (String word);

    /** returns a random word from the WordSet */
    public String selectWord ();

    /** removes a word from the WordSet */
    public void removeWord (String word);

}
```

1-5



Hangman.java

```
public static void main (String args[]) {

    // application of the WordSet data type to the hangman game

    WordSet words = new WordSet();           // create an empty WordSet
    words.addWord("existentialism");         // add some words to list
    words.addWord("neuropharmacology");
    words.addWord("zymurgy");
    words.addWord("glockenspiel");

    while (!words.isEmpty()) {               // game loop
        String targetWord = words.selectWord(); // select a random word
        // ... play Hangman using the current targetWord ...
        list.removeWord(words);             // remove word from list
    }

}
```

1-6



Efficiency Matters in this Course!

E.g.: I'm thinking of an integer between 1 and 1000.
I will tell you if any guess is low, high, or correct.
What's the minimum number of guesses you need?

This is the key idea behind *binary search*.

Can use binary search to quickly find items in a sorted array:

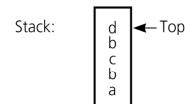
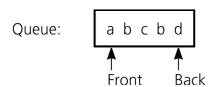
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	6	7	9	12	17	24	33	37	38	42	57	77	84	85	98	173

1 - 7



Some Fundamental Data Structures

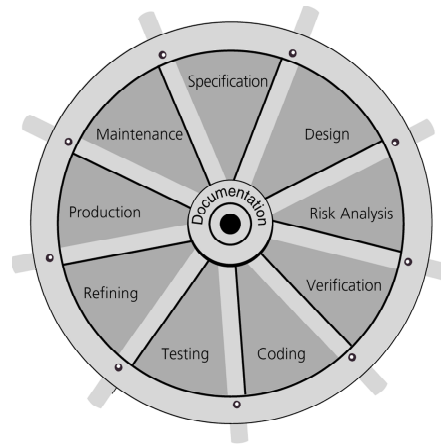
- Array
- Vector (growable array)
- Linked list
- Stack
- Queue
- Priority Queue
- Set
- Bag
- Table
- Tree
- Graph



1 - 8



The Life Cycle of Software



The life cycle of software as a water wheel that can rotate from one phase to any other phase

1 - 9



Writing, Compiling, and Running Java

```
public class Welcome {  
    public static void main (String args[]) {  
        System.out.println("Welcome to CS230!");  
    }  
}
```

- Linux command to compile a Java program
- javac Welcome.java
- Linux command to run a Java program
- java Welcome

1 - 10



Java Review: Example

```
public class Counter {  
  
    private int count;  
    private int id;  
    private static int numCounters = 0;  
  
    public Counter () {  
        numCounters++;  
        id = numCounters;  
        count = 0;  
        System.out.println(this);  
    }  
  
    public void inc () {  
        count++;  
        System.out.println(this);  
    }  
}
```

```
public String toString () {  
    return "[" + id + ":" + count + "];"  
}  
  
public static void main (String [] args) {  
    Counter a = new Counter();  
    Counter b = new Counter();  
    a.inc(); a.inc(); b.inc(); a.inc();  
}  
}
```

1 - 11



Administrivia

- Labs (please sign up for one on sheet)
 - Lab 01: Tue 1:30 -- 3:20pm
 - Lab 02: Tue 6:00 -- 7:50pm
- Accounts
- Textbooks (or lack thereof)
- Problem Sets
 - Collaboration Policy
 - Lateness Policy/Lateness Coupons
 - Drop-in Tutors/One-on-one tutors
- Exams
 - Exam 1: Take-home (Tue. Mar. 6 - Tue Mar. 13)
 - Exam 2: In Lab (Tue. May 1)
 - No Final Exam!
- Final Project

1 - 12