Welcome!

Program Design and Analysis II

Professor: Jeffrey Forbes

M, W 4:25-5:40
BI01 LSRC
CPS 100
What is Computer Science?

What is it that distinguishes it from the separate subjects with which it is related? What is the linking thread which gathers these disparate branches into a single discipline? My answer to these questions is: it is the art of designing efficient and elegant methods of getting a computer to solve problems, theoretical or practical, small or large, simple or complex. It is the art of programming a computer. What is the linking thread which it is related to these disparate subjects with which it is related? What is that distinguishes it from the art of programming a simple computer? — C.A.R. (Tony) Hoare
What is the nature of intelligence? How can one predict the performance of a complex system? What is the nature of human cognition? Does the natural world 'compute'?

It is the interplay between such fundamental challenges and artificial intelligence, state-of-the-art distributed computing systems, algorithms, mathematics. Your Google search for a friend depends on computer security innovations that make computer science so interesting. The results from even the most esoteric computer science research programs often have widespread practical impact. Computer science depends upon the innovations in mathematics. Your Google search for a friend depends on computer security innovations that make computer science so interesting. The results from even the most esoteric computer science research programs often have widespread practical impact. Computer science depends upon the innovations in mathematics.

What is the nature of intelligence? How can one predict the performance of a complex system? What is the nature of human cognition? Does the natural world 'compute'? Programming = Computer Science
Efficient design, programs, code

Object-oriented design

Standard and Duke CPS

Greedy methods, ... 

Mathematics, design

Efficient methods, ... 

Programming?

Engineering, scientist: what

transaction language, but ... 

idioms, its idioms, its

Data structures, libraries, design principles

and algorithms, Trees.

Using the language: Java

Engineer, scientist, what toolkits do you bring to

programming?

Know data structures

...
Course Overview

Lectures, Recitations, Quizzes, Programs

Recitation based on questions given out in previous week
• Discuss answers, answer new questions, small quiz
• More opportunities for questions to be answered

Lectures based on readings, questions, programs
• Online quizzes used to motivate/ensure reading
• In-class questions used to ensure understanding
• More opportunities for questions to be answered

Programs
• Theory and practice of data structures and OO Programming
• Fun, practical, tiring, …

Exams/Tests
• Weekly programs and longer programs
• Theory and practice of data structures and OO Programming
• Final: open book
• Semester: closed book

CPS 100
Frequently Asked Questions

What is the prerequisite? (choose one)
- CPS 6
- 4 or 5 on AP Computer Science AB exam
- 4 or 5 on AP Computer Science AB exam
- 4 or 5 on AP Computer Science AB exam
- CPS 6

What about CompSci 100E?
You should take 100E if you took Engr 53 and do not have Java experience

How does this course fit into the curriculum?
You should take 100E if you took Engr 53 and do not have Java experience

What is recitation? Is it required? When does it start?
Recitation is a more hands-on section where you will do problems and discuss solutions. Your work there will be graded.
Recitation begins tomorrow

What is recitation? It's the foundation of your whole computer science education.
Solid grounding in programming, data structures, and algorithms
Required for majors & minors

How do I keep up to date?
- Read web page regularly
  http://www.cs.duke.edu/courses/spring05/cps100
- Read discussion forum regularly
- Read your email

CPS 100

Did you ask any good questions today?

Ipsa bene volutave eos accusam ad similique reque.

Make study groups with your classmates
Go to office hours
Participate in class

Fear of public displays of ignorance prevents learning
Children are born curious
Questions which illuminate help nourish ideas

Arno Penzias

Ipsa bene volutave eos accusam ad similique reque.

Did you ask any good questions today?

On the subject of questions…
In class, we will use the Personal Response System (PRS)

Buy receiver at bookstore, use throughout Duke career

ConceptTest: multiple choice questions that highlight an important concept gleaned from the lecture and/or reading

It's OK not to know!

Important concepts gleaned from the lecture and/or reading

Discuss your assignment responses in class

Keep in folder

In-class work

Why respond?

Work in groups

With your neighbor and revote

Peer instruction: after seeing the results, you will have counter

Effort, participation, and altruism

Counts toward in-class work score

Just in Time Teaching

Makes class better!

In-class work

In-class work

Why respond?

In class, we will use the Personal Response System (PRS)
Practice Questions

What year are you?
1. Frosh
2. Soph
3. Junior
4. Senior
5. Grad
6. Grad Plus

What kind of computer do you own/use most often?
1. Windows PC
2. Mac
3. Linux Box
4. Other
5. Other

How many web pages does Google index?
1. 1 million
2. 10 million
3. 100 million
4. 1 billion
5. 10 billion
6. 1 trillion
7. A googolplex
8. A googolplex

What's Google?
What do we decide what simplicity, elegance, ... \( \rightarrow \) time, \( CPU \) time, \( \text{CPU time} \), \( \text{Runtimes, space, your run-time, at-all programs, \( run \) anywhere} \) \( \rightarrow \) fast programs, small programs, small programs, small programs...

Generality, simplicity, tension between tradeoffs are important.

How do we decide what simplicity, elegance, Einstein...\( \rightarrow \) what does XP say about goals in programming? Efficient: what are our simple, elegant, quick, structural, algorithmic, data..., programming, design, programming, design, programming, design.

Tradeoffs
Count number of different words in an array, how can we accommodate more than one approach?

Three (or more) approaches:

- Sort the array of strings, then iterate through and compare $\text{list}[I]$ to $\text{list}[I+1]$. If the same, do nothing, otherwise add one to a count.
- Create a map of words to $X$, add all, and then iterate through and compare, otherwise add one.
- Create a map of words to $X$, add all, and then count the number of keys.

Do we design in code/wordcount?
This is valid and correct Java code, questions?

```java
import java.util.*;

public class SetUniqueCounter implements UniqueCounter {
    public int uniqueCount(String[] list) {
        TreeSet set = new TreeSet();
        set.addAll(Arrays.asList(list));
        return set.size();
    }
}
```
Some Java Vocabulary and Concepts

Java has a huge standard library

Java methods have different kinds of access/inter/intra class

API browseable online, but Eclipse IDE helps a lot

Primitive types (int, char, double, boolean) are not objects but

Protected and package methods ...

Private methods ...

Public methods ...

Everything else is literally an instance of class Object
Arrays are typed and fixed in size when created. Unlike vector/tvector in C++/Tapestry, arrays don't have to be filled, but cannot be expanded. Can store int, double, String, Foo, ... Array/List objects grow themselves intelligently. Accessing elements can require a downcast. Stores objects, not primitives. 

ArrayList (and related class Vector and interface List) grows. Can store int, double, String, Foo, ... Don't have to fill the array, but cannot expand it. Not like vector/tvector in C++/Tapestry.

Arrays are typed and fixed in size when created. Basic data structures and algorithms.
We want to know how many times 'the' occurs. Do search engines do this? Does the number of occurrences of "basketball" on a page raise the priority of a webpage in some search engines?

**Constraints on solving this problem**

- We must read every word in the file (or web page)
- Search for the word? Avoid counting twice? Store?
- Are there fundamental limits on any of these operations?

**Downside of this approach for search engines**

- A webpage in some search engines
- Occurrences of "basketball" on a page raise the priority of
- Do search engines do this? Does the number of
- We want to know how many times "the" occurs

**Improvements**

Where should we look for data structure and algorithmic
public class SlowUniqueCounter implements UniqueCounter{
    public int uniqueCount(String[] list) {
        int count = 0;
        int diffSize = list.length;
        for(int k=0; k < diffSize; k++) {
            String word = list[k];
            count++;
            for(int j=k+1; j < diffSize; j++) {
                if (list[j].equals(word)) {
                    list[j] = list[diffSize-1];
                    diffSize--;
                }
            }
        }
        return count;
    }
}

What does try to do? Why is it wrong?
How fast is fast enough?

/** pre: a contains a.size() entries
 * post: return true if and only if key found in a
 */
boolean search(ArrayList a, String key)
{
    for(int k=0; k < a.size(); k++)
    if (a[k].equals(key)) return true;
    return false;
}

Java details: parameters? Return values?ArrayLists?

How do we measure performance of code? Of algorithm?

Does processor make a difference? G5? Itanium? 64-bit?
Tradeoffs in processing and counting

What is best case, worst case, average case?
Are there kinds of data that make one approach preferable?

How do we look up word, how do we add word
If we look up words one-at-a-time and bump counter if we haven't seen a word, is this slower than previous idea?

Frog, frog, frog, rat, tiger, tiger, tiger, tiger
Read words, then sort, determine # unique words?
Inheritance and Late/Dynamic Binding

Why is this an advantage?

Why can we pass different kinds of objects to test?

```java
public static void test(UniqueCounter uc,
String[] list)
{
    double start = System.currentTimeMillis();
    int count = uc.uniqueCount(list);
    double end = System.currentTimeMillis();
    System.out.println(count + " unique words" +
    (end - start) / 1000 + " seconds" +
    System.out.println(uc.getCount("
    int count = uc.getCount("test");
    double start = System.currentTimeMillis();
    System.out.println("test") +
    public static void test(UniqueCounter uc,
        Consider new algorithm for determining unique word count

Benefits of inheritance, interfaces
Why inheritance?

Add new shapes easily without changing much code.

```java
Shape s1 = new Circle();
Shape s2 = new Square();
```

Interface/abstract base class:

```
interface or abstraction
```

Function called at runtime:

```
Concrete subclass
```

```
Inheritance called at runtime
```

```
Interface or abstraction
```

```
: void shape ()
```

```
: void circle ()
```

```
All abstract functions implemented
```

Later we'll override.

```
"is-a" view of inheritance
```

```
"is-a" view of inheritance
```

Substitutable for, usable.

Later we'll override.

```
"is-a" view of inheritance
```

```
User's eye view: think and program with abstractions, realize different, but conforming implementations, don't commit to something concrete until as late as possible.
```

```
Concrete subclass
```

```
Inheritance called at runtime
```

```
Interface or abstraction
```

scoreEntry

Full House, Large Straight

Shape

Mammal
What is behavior of a shape?

```java
void doShape(Shape s) {
    System.out.println(s.area());
    System.out.println(s.perimeter());
    s.expand(2.0);
    System.out.println(s.area());
    System.out.println(s.perimeter());
}

Shape s1 = new Circle(2);
Shape s2 = new Square(4);
Shape s3 = new Rectangle(2, 5);
doShape(s1); doShape(s2); doShape(s3);```

Example of inheritance
Inheritance (language independent)

First view: exploit common interfaces in programming
- Iterators in Java or C++/Tapestry

Second view: share code, factor code into parent class
- Code in parent class shared by subclasses
- Subclasses can override inherited method
- Subclasses can override and call
- Polymorphism/Late (run-time) binding (compare: static)
- Function actually called determined when program runs, not when program is compiled

Inheritance (language independent)
Review/Preview: Anagrams/Jumbles

Brute-force approach to finding anagrams/solving Jumbles

Brute-force often thought of as "lack of thought"

What if the better way requires too much thought?

What if there’s nothing better?

--

Is there a better method?

What's the problem here?

What if there's nothing better?

Brute-force often thought of as "lack of thought"
Who is Alan Perlis?

- It is easier to write an incorrect program than to understand a correct one.
- Simplicity does not precede complexity, but follows it.
- If you have a procedure with ten parameters you probably missed some.
- If you're explaining your program, wake him up when you're explaining your program.
- If a listener nods his head, he probably missed some of your explanation.
-Won first Turing award.

http://www.cs.yale.edu/homes/perlisa/quotes.html

Who is Alan Perlis?