CISC 220 Data Structures - introductory items

Professor: David Saunders TA: Hogyeong Jeong

A course on the Design, Analysis, Implementation (in C++), and Testing (and uses) of data structures.

We have a Sakai site. It will contain code and reading material. It's home page index will be updated frequently.

We have a syllabus, read it. We'll review it next time.

We have a Piazza site. A place for class discussion, to ask and answer questions.

wordcount.cpp, a simple yet powerful word counting C++ program

```
/* word occurrence count program    -bds 2004Feb, 2013Aug
 *
 * This program takes a text and prints a table of the
 * words, each with its number of occurrences in the text.
 *
 * The text is from the standard input.
 */

#include <map>
#include <iostream>
using namespace std;
```

wordcount.cpp continued

```
int main() {
  // build the table
  map<string,int> WC; // WC is the word counts table.
  string word;
  while ( cin >> word )
    WC[word]++;

  // output the results
  map<string,int>::iterator p; // pointer into the map
  for (p = WC.begin(); p != WC.end(); ++p)
    cout << p->second << " " << p->first << endl;
}
```

Notes on C++ and the power of good data structures

1. C++ code file suffix is .cpp (or .C or .cc).
2. Use "`#include <iostream>`" to get the type `string` and the objects `cin, cout, >>, <<, endl`.
3. For type T (int, double, a struct, etc) and a type C on which the "<" comparison operator is defined (think number types or strings). A `map<T,C>` acts much like an array of T, but the indices are of type C instead of being ints.
4. A C++ map is a sophisticated and powerful data structure – think about it.
5. This is a course in how to make high performing data structures such as `map`.

Clicker questions

1. Which is a valid C++ file suffix?
   - A .C
   - B .cpp
   - C .cc
   - D All of the above

Clicker questions

1. Which is a valid C++ file suffix?
   - A .C
   - B .cpp
   - C .cc
   - D All of the above

2. Suppose you have a C++ (or Java) system including I/O support, but no library containing an associative structure such as the STL map. In other words, you have to write your word occurrence counting program "from scratch". How long would it take?
   - A short: an hour or two.
   - B long: A day, maybe a week.
   - C short: half a day, but the resulting program would be sloooow on large texts.
   - D Who cares, I'd never be foolish enough to work without good tools.

Problem analysis illustrated on some simple examples

Given an array of numbers, what is a good way to solve each of these problems?
The solution function is allowed to move the numbers around in the array.
The goal is to use a minimum number of comparisons.

max **Find the largest entry in the array.**

max-min Find the largest entry and the smallest entry in the array.

max-2 Find the largest entry and the second largest entry in the array.

Solution to the max problem

```
double arrayMax( double A[], int n ) {
// A is of size n.  Return largest entry in A.
  double max = A[0];
  for (int i = 1; i < n; ++i)
    if (max < A[i]) max = A[i];
  return max;
}
```

How many comparisons of array entries?
Is this the best one can do?

arrayMax is optimal

Proposition

`arrayMax` *uses* $n-1$ *comparisons of array entries.*
Any *solution to the max problem uses at least* $n-1$ *comparisons.*

Proof.

The first statement is evident: The only comparison is in the if condition. It is done for $i = 1, 2, \ldots, n-1$.
For the second statement, consider any version of arrayMax that someone may come up with. Think of the data as being manipulated by an "adversary" out to make the program do the wrong thing. Observe that every entry (except the reported largest one, x) must "lose" a comparison. Otherwise, the adversary can doctor the data so that the program is wrong: If some entry, y, (other than the reported largest, x) never loses a comparison, then the adversary could make y = x+1, without causing any comparison result in the algorithm to change. Thus that implementation cannot possibly be correct. QED.

arrayMax and priority queues

An important data structure is the priority queue: A priority queue has two operations: (1) inserting an item in the queue and (2) extracting the highest priority item.

Operating systems use priority queues to manage timesharing of running jobs. Web services use priority queues to manage service delivery timeliness. Algorithms use priority queues to manage searches through data (shortest path problems), etc., etc..

arrayMax could be thought of as a step toward a priority queue design: Insertion would be to add an element to the array, extraction would be arrayMax (modified to remove the item). Turns out: *we can do much better than this*.