

# CS 345 - Programming Languages Spring 2008

## Homework #1

Due: 2pm CST (in class), January 29, 2008

**YOUR NAME:** \_\_\_\_\_

### Collaboration policy

**No collaboration** is permitted on this assignment. Any cheating (*e.g.*, submitting another person's work as your own, or permitting your work to be copied) will automatically result in a failing grade. The Computer Sciences department code of conduct can be found at <http://www.cs.utexas.edu/users/ear/CodeOfConduct.html>

### Late submission policy

This homework is due at the **beginning of class** on **January 29**. All late submissions will be subject to the following policy.

You start the semester with a credit of 3 late days. For the purpose of counting late days, a “day” is 24 hours starting at 2pm on the assignment's due date. Partial days are rounded up to the next full day. You are free to divide your late days among the take-home assignments any way you want: submit four assignments 1 day late, submit one assignment 3 days late, *etc.* After your 3 days are used up, no late submissions will be accepted and you will automatically receive 0 points for each late assignment.

You may submit late assignments to Vitaly Shmatikov (TAY 4.115C—slide under the door if the office is locked). **If you are submitting late, please indicate how many late days you are using.**

**Write the number of late days you are using:** \_\_\_\_\_

## Homework #1 (26 points)

### Problem 1 (5 points)

Describe some design trade-offs between efficiency and safety or reliability in any programming language you know.

### Problem 2 (5 points)

The first meeting of the committee that designed ALGOL took place in 1958. It set three goals for the new language:

1. The syntax of the language should be as close as possible to standard mathematical notation, and programs written in it should be readable with little further explanation.
2. It should be possible to use the language for the description of algorithms in publications.
3. Programs in the new language must be mechanically translatable into machine language.

Which goal, in your opinion, was most novel for computing at that time? Which goal was most difficult to achieve?

### Problem 3

Here is the pseudocode for QUICKSORT.

```
quicksort(A)
{
  if (length(A) < 2) return A
  else {
    pick some x in A
    A1 = { y in A : y < x }
    A2 = { y in A : y > x }
    A3 = { y in A : y = x }
    quicksort(A1)
    quicksort(A2)
    return concatenation of A1, A3, and A2
  }
}
```

A direct implementation of the above algorithm would not be very efficient. When evaluating `quicksort(A)`, a recursive call must be made to `quicksort(A2)`. When this call is made, the context of `quicksort(A)` must be kept on the stack in order to remember the value returned by `quicksort(A1)`, with which the result of `quicksort(A2)` will be concatenated after the call to `quicksort(A2)` returns.

#### Problem 3a (8 points)

Implement QUICKSORT in C or C++ so that it is **tail-recursive**, *i.e.*, there is no need to remember the calling context. Instead, the array is completely sorted at the end of the chain of recursive calls, without returning to the calling function.

#### Problem 3b (8 points)

Implement QUICKSORT so that it does not use recursion at all.

### Submission instructions for Problem 3

**IMPORTANT**: Your code must **compile** using `gcc` or `g++`. Any submission that does not compile or, when executed, does not print the correctly sorted list of numbers from `array.txt` will automatically receive 0 points.

1. Submit a paper printout of your source code, stapled to this homework. The printout **must** be processed using the following command:

```
enscript -C -2Gr -Ec <yourfile> -o <outputfile.ps>
```

2. Submit your source code electronically using the following command:

```
turnin --submit austin hw1 <filename1> <filename2> ...
```

Your code should take as its input a text file called `array.txt`, which contains the unsorted array in the form of 20 or so space-separated integers, *e.g.*, `5 981 14 3 ....`. It should print the sorted list on the screen as its output.