

CS 345 - Programming Languages Spring 2008

Homework #6

Due: 2pm CDT (in class), April 10, 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 **April 10**. 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 #6 (34 points)

Problem 1

Consider the following Java implementation of a simple publish-subscribe system. When the publisher changes state, all subscribers should be notified and updated automatically.

```
public class Publisher {
    private List subscribers = new LinkedList();
    private String data;

    public interface Subscriber {
        public void updateData(String newData);
    }
    public void addSubscriber(Subscriber subscriber) {
        subscribers.add(subscriber);
    }
    public void publishData(String newData) {
        data = newData;
        Iterator i = subscribers.iterator();
        while(i.hasNext()) {
            ((Subscriber)i.next()).updateData(newData);
        }
    }
}
```

Problem 1a (2 points)

Is the above implementation thread-safe? Explain.

Problem 1b (4 points)

Suppose we make `publishData` and `addSubscriber` methods synchronized. Does this solve the problem? If not, give a specific example and explain what goes wrong.

Problem 1c (4 points)

Suppose the `addSubscriber` method is synchronized, and the `publishData` method is as follows:

```
public void publishData(String newData) {
    List copyOfSubscribers;
    synchronized(this) {
        data = newData;
        copyOfSubscribers = new LinkedList(subscribers);
    }
    Iterator i = copyOfSubscribers.iterator();
    while(i.hasNext()) {
        ((Subscriber)i.next()).updateData(newData);
    }
}
```

Is the new implementation thread-safe? If not, give a specific example and explain what goes wrong.

Problem 2

Consider the following imperative program:

```
function f(x) { return x+1; }  
function g(y) { return 1-y; }  
f(g(0));
```

Problem 2a (3 points)

Write the above program fragment as a λ -expression.

Problem 2b (2 points)

Evaluate your expression by choosing, at each step, the reduction that eliminates the *leftmost* λ that can be reduced.

Problem 2c (2 points)

Evaluate your expression by choosing, at each step, the reduction that eliminates the *rightmost* λ that can be reduced.

Problem 3

Problem 3a (5 points)

Translate the following ML program into λ -calculus:

```
(let fun subtract(x) =  
      fn(y) => let fun unaryminus(z)=~z in  
                x+unaryminus(y)  
              end  
in  
  subtract 10  
end) 20
```

Problem 3b (2 points)

Reduce the resulting λ -expression.

Problem 4 (10 points)

Read section 14.2.10 in Tucker & Noonan. It describes a Scheme program which uses backtracking to solve the EIGHT QUEENS problem.

Write a Scheme program using backtracking to solve the KNIGHT'S TOUR problem. A chess knight, starting from the (1,1) square (bottom left square) of a chess board, has to visit every square on the board. The knight is only allowed to move according to chess rules. Given the size of the board (N), your program will compute a sequence of $N^2 - 1$ chess-knight moves that, starting from (1,1), will visit every square (i, j) , where $i \leq N, j \leq N$ **exactly once**.

Tips and hints

We have created an account for each student enrolled in the course at this website: <http://z.cs.utexas.edu/users/arvindn/wordpress> (or follow the link from the course website) You can obtain useful tips for the current take-home assignment by logging into your account. Your username is your UT EID. You must **update your password** for each assignment.

We may use some of the passwords in our research on password security. **IMPORTANT:** Do **not** use your UT Direct, UT CS or any other existing password for the above account!

Submission instructions for Problem 4

1. Submit a paper printout of your Scheme code, stapled to the first page of this homework (the one showing your name and the number of late days you are using, if any). The printout **must** be processed using the following command:

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

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

```
turnin --submit austin hw6 <filename1> <filename2> ...
```