CSE 305 Programming Languages Spring, 2010
Homework 5
Maximum Points: 24
Due 10:30 AM, Friday, February 26, 2010

Professor Shapiro

February 12, 2010

Write the answers in a file named `hw5.txt`. Put your name and user name at the top of the file, and submit the file, using the UNIX command, `submit_cse305 hw5.txt`. You are also to submit five additional files as instructed below.

1. (4) You may wonder if there is any advantage to using a symbol rather than a string; for example whether there's any difference in Ruby between using the string `"jpbona"` vs. the symbol `:jpbona`. Both, after all, are objects. One question is, are multiple occurrences of the "same" string automatically stored in the same memory location; what about multiple occurrences of the "same" symbol. If "two" strings are stored in the same memory location, they may be tested for pointer (reference) equality, but if they're in different locations, the equality tester would have to compare them character by character. Use `irb`, the interactive Ruby shell to investigate this issue. You start `irb` by entering `irb` as a Unix command, and leave it by entering `quit` to the `irb` prompt. The Ruby pointer equality method is `equal?`. For example,

   ```
   <timberlake:Test:1:176> irb
   irb(main):001:0> 4.equal?(3+1)
   => true
   irb(main):002:0> quit
   <timberlake:Test:1:177>
   ```

   (a) (3) Put in your submission file a transcript of a run of `irb` in which you investigate whether two occurrences of the "same" string are stored in the same memory location, and whether two occurrences of the "same" symbol are.

   Also put your input lines in a file named `hw5.rb`, and submit that file.

   (b) (1) What do you conclude about the question from this test?

Continued on next page.

2. (7) Can you use strings as the cases in a C switch statement? Is the following C program legal? What does it do?

```
#include <stdio.h>

char* daysChild(char* d)
{
  // Rhyme from Lisa Yannucci, Mama Lisa’s World,
  //    http://www.mamalisa.com/?t=hes&p=1420&l=S, 2009.
  switch(d)
    {
    case "Sunday": return "’Sundays child is full of grace.";
      break;
    case "Monday": return "Monday’s child is full in the face.";
      break;
    case "Tuesday": return "Tuesday’s child is solemn and sad.";
      break;
    case "Wednesday": return "Wednesday’s child is merry and glad.";
      break;
    case "Thursday": return "Thursday’s child is inclined to thieving.";
      break;
    case "Friday": return "Friday’s child is free in giving.";
      break;
    case "Saturday": return "Saturday’s child works hard for a living.";
      break;
    default: return "Unknown day’s child is a mystery.";
    }
}

int main()
{
  char* d = "Wednesday";
  printf("%s\n", daysChild(d));
  return 0;
}
```

   (a) (2) Try to compile this program (using `gcc -Wall ...`), and, if it compiles, run it. Insert a transcript of the compile and run (if it runs) here in your submission file.

   (b) (3) Change the program to use an enumeration type instead of strings for the days. Put that changed program in a file named `daysChild.c` and submit it. Also insert your changed program here in the submission file.

   (c) (2) Compile this program (using `gcc -Wall ...`) and run it. Insert a transcript of the successful compile and run here in your submission file.

Continued on next page.

3. (5) One use of jagged arrays is to store triangular arrays without wasting space on the empty cells. Just as maps (associative arrays) are a generalization of arrays, so are jagged maps, which are maps of maps, a generalization of jagged arrays, and we could use jagged maps to store triangular maps. A common use of triangular maps is for mileage charts. For example, the following is a small mileage chart[1] showing distances between a few US cities:

	Buffalo	Chicago	Los Angeles	New York
Atlanta	859	674	2182	841
Buffalo		522	2554	372
Chicago			2054	802
Los Angeles				2786

To use this chart to find the distance between two cities, take the alphabetically smaller city, and read across its row until you find the column of the other city. The number you find there is the distance in miles between the two cities.

Write a Python program to read the names of two cities and print the distance between them. You should represent the mileage chart as a jagged dictionary (Python's name for an associative array), using strings of city names as the keys. Here's an example run of such a program, showing what your program should be able to do:

```
<timberlake:Test:1:206> python mileage.py
Starting city (bye to quit):  Buffalo
Destination city:  New York
It is 372 miles from Buffalo to New York.

Starting city (bye to quit):  Los Angeles
Destination city:  Chicago
It is 2054 miles from Los Angeles to Chicago.

Starting city (bye to quit):  Atlanta
Destination city:  Baltimore
I don't know how far it is between Atlanta and Baltimore.

Starting city (bye to quit):  Albany
Destination city:  Buffalo
I don't know anything about Albany.

Starting city (bye to quit):  bye
Good bye.
```

Note that the basic structure of this program is the same as that of `echo.py`.

(a) (3) Put your program in a file named `mileage.py` and submit it. Also put it here in your submission file.

(b) (2) Put a transcript of your run of your program here in your submission file.

Continued on next page.

[1] Data taken from *Rand McNally Road Atlas & Major City Guide*, 1994

4. (5) Fortran 77 and earlier versions of Fortran do not have pointers. However, pointers may be simulated by parallel arrays. Edit the following program to replace the question marks (“?”) with appropriate integers so that when you compile and run the program, it prints the 12 months in the correct order starting with `"January"`. Copy the edited program into a file named `months.f`. Compile the program with the Unix command

```
f77 -o months months.f
```

and then run it with the Unix command

```
./months
```

Here’s the program:

```
      Program months

      Character*9 monName(12)
      Integer i, firstMonth, nextMonth(12)

      Data monName/"November", "April", "September", "May",
     C             "December", "June", "February", "October",
     C             "January", "July", "March", "August" /

      Data firstMonth/?/
      Data nextMonth/?,?,?,?,?,?,?,?,?,?,?,?/

      i = firstMonth
 10   If (i .NE. 0) Then
        Print *, monName(i)
        i = nextMonth(i)
        Goto 10
      EndIf
      End
```

(a) (3) Submit your `months.f` file, and put it here in your submission file.

(b) (2) Put a transcript of your run of your program here in your submission file.

Continued on next page.

5. (3) The Lisp function `mapcar` takes a first argument of type function, and n additional additional arguments, which must be lists, and where n is the number of argments that the function takes. `mapcar` returns a list of the value of the function applied to the first element of each list, the second element of each list, *etc.* Here is an example of an interactive run of Common Lisp in which `mapcar` is used to compute a list of sums:

   ```
   <timberlake:Test:1:208> composer
   International Allegro CL Enterprise Edition
   8.1 [Linux (x86)] (Aug 21, 2009 10:13)
   Copyright (C) 1985-2007, Franz Inc., Oakland, CA, USA.  All Rights Reserved.

   This development copy of Allegro CL is licensed to:
      [4549] University at Buffalo

   ; Autoloading for package "asdf":
   ;   Fast loading /util/acl81/code/asdf.fasl
   ;; Optimization settings: safety 1, space 1, speed 1, debug 2.
   ;; For a complete description of all compiler switches given the
   ;; current optimization settings evaluate (explain-compiler-settings).
   ;;---
   ;; Current reader case mode: :case-sensitive-lower
   cl-user(1): (mapcar #’+ ’(1 2 3 4) ’(5 6 7 8))
   (6 8 10 12)
   cl-user(2): :ex
   ; Exiting
   <timberlake:Test:1:209>
   ```

   Put a transcript of an interactive Common Lisp run in which you use `mapcar` to compute a list of averages here in your submission file. (Hint: the form you have Lisp evaluate will look like

   ```
       (mapcar ____________ ’(1 2 3 4) ’(5 6 7 8))
   ```

   but with the blank filled in with a lambda expression.) Also put that input in a file called `hw5.cl`, and submit it.