

CS306 – Introduction to Perl  
Spring 2006  
Homework Assignment #5  
Due: Friday April 7, 2006 (12:20pm)

Please see the guidelines on homework #1 on how to submit your homework. Also, see the following additional guidelines:

- All programs must use strict and must contain an appropriate level of commenting throughout.
- All work must be original. This is not a new requirement, however, I have been noticing cut-and-paste from the slides, which is not acceptable.

### Homework #5

Question 1. What are the differences between symbolic and hard references? Illustrate each with a code example. What are the advantages of a hard reference over a symbolic reference?

Question 2. What is anonymous data? Give an example of the creation of anonymous data. What are the advantages of anonymous data?

Question 3. Write code that creates an arrayref that points to a two-dimensional anonymous array.

Question 4. John has four children, Nancy, Sally, Peter and John Jr. Write code to create a hashref to an anonymous hash that has a key 'children', which holds an anonymous array of John's children's names.

NOTE: Questions and Program below require the lecture to be delivered 3/24. You can look ahead in the slides if you want to get a head start.

Question 5. Given the following:

```
$steamref = ['blazers', 'crimson tide', 'tigers'];
```

What are the three different dereferencing syntaxes to change the value of 'crimson tide' to 'tide'?

Question 6. Explain autovivification and give a code example to illustrate the principle.

Program 1. Company Directory. You are to write a program that implements a company-wide phone book for a multi-national corporation. The data hierarchy is as follows:

AcmeCorp  
Country  
Branch  
Department  
Employee Entries

An example of a record might be:

```
$acme->{usa}->{boston}->{sales}->{lname => 'Smith', fname => 'John', phone => '713-555-1212', email => 'jsmith@usa.acme.com'}
```

This program should read in the data from a file containing lines formatted like:

```
lname=Smith|fname=John|email=jsmith@usa.acme.com|department=sales|branch=boston|country=usa  
phone=713-555-1212 (this would all be one line in the file itself)
```

and build the data structure in memory.

Then the program should implement the following operations:

1. Add an entry
2. Delete an entry
3. Modify an entry (only allow modification of phone, email, branch)
4. List branch – list all the employees of a particular branch
5. Last Name Search – perform a search for a particular last name and display that user's full record.  
Consider the case of two people with the same last name.

Note that your employee record data structure does not need to follow mine exactly! For example, you might want to consider that the email address is guaranteed to be unique for each entry. Items guaranteed to be unique might make good hash keys. You have total flexibility over the representation of your employee entries internally, as long as the functionality is correct. Having a unique key might help in the case where a search return multiple records.

Note that you -must- start with an \$acme reference at the top, and have tiers for country, branch, and department. However, the objects that the department tier contains (the employee records) can look however you want, as long as they contain the data and perform the functionality asked for here. An array of hashes, one per employee, might work. So might a single hash with a series of unique keys, each of which points to another hash to represent that employee. There are many options, think about your data design a bit first.