

# CSCI 132 Practical Unix Programming

## Homework 3

### Solution

Saad Mneimneh  
Computer Science  
Hunter College of CUNY

#### **PART 1**

Read chapter 3 in the UNIX book and do exercises 1, 2, 4, and 5 (of course “your own Unix machine” refers to a machine in the lab).

#### **ANSWER:**

ex1: maybe at work, hierarchy in a company or institute. Yes, a public library could be organized in a hierarchical manner.

ex2: the files .test, .cshrc, ../, .dot., and .HiMom are hidden files, the others are not. A hidden file begins with a . (dot).

ex4: The following are absolute file names:

```
/Personnel/Taylor,D.  
/home/taylor/business/California
```

These are not:

```
../..  
Recipe:Gazpacho
```

#### **PART 2**

In class, we wrote a Perl program to find the best overlap between two strings. Here’s the program that we came up with:

```
#convert strings to arrays a1 and a2 (assume a2 is smaller)
my $l=0;
for (my $i=0; $i<=#a2; $i=$i+1) {
    my $good=1;
    for (my $j=0; $j<=$i; $j=$j+1) {
        if ($a1[$j] ne $a2[ $#a2-$i+$j]) {
            $good=0;
            last;
        }
    }
}
if ($good) {
```

```

        $l=$i+1;
    }
}
# l is now the best overlap to the left

```

The above program identifies the best overlap to the left. To find the best overlap to the right, we can simply reverse both strings and repeat the same thing. However, we did not account for overlaps in the middle; for instance:

```

ababaca
  |||
  baba

```

Write a program to find the best overlap of two strings by considering overlaps to the left, right, and middle. Make your program read the strings from the input.

**ANSWER:** Here's a program to test for an overlap in the middle. Basically, array a2 (the smallest) must be completely contained in a1.

```

my $contained;
for (my $i=0; $i<=#a1-#a2; $i=$i+1) { #possible positions in a1
    $contained=1; #assumed contained until proven otherwise
    for (my $j=0; $j<=#a2; $j=$j+1) { #go through a2 and compare to a1
        if ($a2[$j] ne $a1[$i+$j]) {
            $contained=0;
            last; //exit inner loop if there is a mismatch
        }
    }
}
my $l_middle=0;
if ($contained==1) {
    $l_middle=#a2+1;
}

```

For completeness, here's a code to reverse an array.

```

my @b1=@a1;
my @a1_reverse=();
while ($#b1>=0) {
    $e=pop(@b1);
    push(@a1_reverse,$e);
}
#now @a1_reverse is a1 reversed

```

**PART 3:**

Write a program to read a string from the input and print all its non-empty substrings. For instance, if the string is “saad”, then the substrings are: “s”, “a”, “a”, “d”, “sa”, “aa”, “ad”, “saa”, “aad”, and “saad”.

**ANSWER:**

```
#!/usr/bin/perl -w
use strict;

my $s=<STDIN>;
chomp($s);
my @a=split(//,$s);
for (my $i=0; $i<=$#a; $i=$i+1) {
    for (my $j=$i; $j<=$#a; $j=$j+1) {
        for (my $k=$i; $k<=$j; $k=$k+1) {
            print $a[$k];
        }
        print '\n';
    }
}
```