CS 2413 Final

1. [15] Suppose a perl program, **grab**, is invoked with the command:

```
grab -a file1 file2 file3
```

and suppose file1 contains the lines "dir1" and "dir2" while file 2 contains "dir3" and "dir4" and file3 contains "dir5", "dir6" and "dir7".

At the start of the program:

(a) What is the list value of **@ARGV**?

(b) What is the value of **$#ARGV**?

(c) What value is used in the assignment for **@ARGV**:

```
    $val = @ARGV;
```

(d) If the following statements are executed first.

```
-a  $a = shift;
fil3 $b = pop;
    $d = <>;
```

i. What is the value of **$a**?

ii. What is the value of **$b**?

iii. What is the value of **$ARGV**?

(e) On the other hand if the following statement were to be executed first.

```
    $a = shift;
    $b = shift;
    @d = <>;
```

i. What is the value of **$b**?

ii. What is the value of **@d**?

2. [20] Write a Perl script, called `replines`, which will replace a block of lines in the second file with the corresponding lines from the first file. Thus to replace lines 15 through 25 in `outfile` with the corresponding lines from `infile` the command would be:
`replines 15 25 infile outfile`
Notice that all the other lines of `outfile` are still in the file. You may assume that `outfile` has the same number of lines as `infile`.

3. [20] Write a Perl script, **pgrep**, which will search all **TEXT** files which are under any of the directories whose names are given on the command line for the regular expression which is given as the first argument on the command line. A sample invocation:
`pgrep 'cat.*dog' file1 file2 file3`

[20] An ISP (Internet Service Provider) is concerned about a user named **maynard** spamming email so they want a tool to look in the file `/var/log/syslog` and pull out information about maynard's email usage on March 7. Write a perl script which will determine the total size of the outgoing email written by maynard as well as the total number of email messages received by maynard on March 7. The syslog file will include several months worth of data for many users. The lines (simplified from the real syslog data) look like:

```
Mar  5 21:58:18 medusa sendmail[6432]: to=<maynard@medusa>, stat=Sent
Mar  7 00:06:37 medusa sendmail[7248]: from=<maynard@medusa>, size=681
```

The output should look like:

```
maynard
      1573889 bytes mailed out
      798 emails received
```

5. [25] Write a C **function** with prototype: int ftruncate(int fd , offset length)
`int delrec(unsigned int num, char *file, unsigned int rsz);` A file can be considered to be a sequence of record of a fixed size, say **rsz** where the first **rsz** bytes are to be considered record 0, the next **rsz** bytes are record 1, ... The function **delrec** will delete the rsz bytes in record **num** in the file, **file**, moving every record from that point on forward one record. The function must both open and close the file. If successful, **delrec** will return a 0 else **delrec** will return -1. The function should use low-level I/O.

6. [25] Implement a low-level popen, called **ppopen**. The prototype is:

```
int ppopen(const char *command, const char *type);
```
pg 437

The function ppopen will fork and exec **command** after having created one pipe. If type == "r" then the pipe will be set for **command** to write back to the parent process via stdout while if type == "w" then the pipe will be set for the parent process to write over the pipe to **command**'s stdin. The function ppopen will return the appropriate file descriptor so that the parent can communicate, in the appropriate direction, with **command**.

7. [25] You have a program called **snort** which sniffs the network and writes to a file, `/var/log/alert`, all of the suspicious packets. Unfortunately there are many of these and you just want to watch some particular types of packets as they arrive. Write a C **program**, called **gsnort**, that will look at all new entries in the file `/var/log/alert` and will print those which match your regular expression **as they arrive** to stdout. This could be done on the command line using a pipe of the form

```
tail -f /var/log/alert | egrep regexp
```

where **regexp** is the regular expression of interest. A **sample** invocation would be

```
gsnort 206.145.48.32
```

8. [25] Write a C program which will create a **total** of 37 processes all of which will have their own separate pipe, i.e. all processes will be able to write to all other processes but each process will only read from its own pipe. In addition each process will have an assigned index, 0...36.

   Each process will check it's pid and if its pid is even, it will write its pid to each of the processes with even indices while if its pid is odd it will write its pid to each of the processes with odd indices (but no process writes to itself). Then each process will read all the messages written to its pipe and print on stdout, for each message, a line similar to:
   Process 35894 heard from process 35992
   Be careful that your code doesn't hang!

9. [25] In the system that you are writing, many programs (called clients) will be executing and will need other programs, called servers, to process some data. In this version of the system each program will write to a well known fifo (named pipe) with the name "`/tmp/dispatch`". You should assume that this fifo already exists. The dispatcher program reads a message from the fifo, execs the appropriate server and then writes the data to the server. The server is expecting its data via stdin and will write its data to stdout. The dispatcher must be certain that the server's stdout is redirected to the fifo, created by the client, named in the message. The format of the message is:

Data	**Field (first byte - last byte)**
program name	0-255
return fifo name	256-511
data	512-1023

   Write the dispatcher program for this system assuming that there may be several dispatcher programs running simultaneously. In addition the program should be written so that zombies are not created. Note that the dispatcher does not terminate but constantly reads messages from the dispatch pipe.