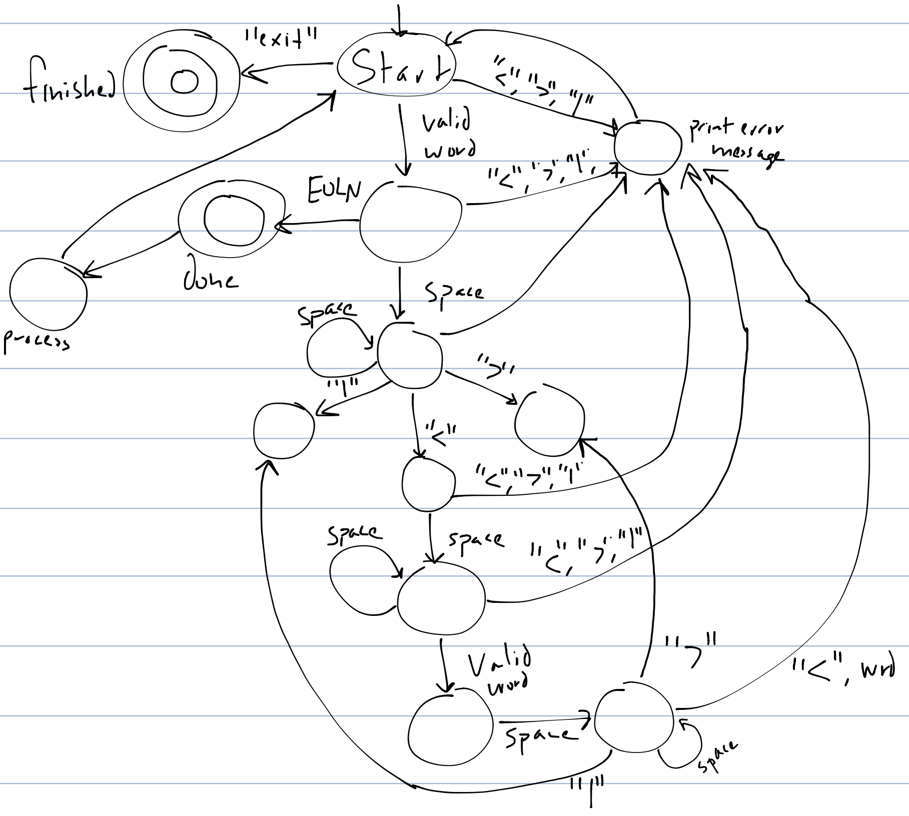
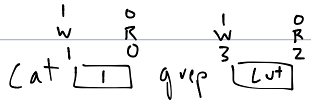


```
cat scores | grep uva | cut c4
```

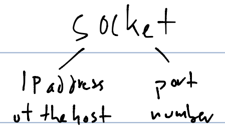
```
char* cat_args[] = {"cat", "scores", NULL};
char* grep_args[] = {"grep", "uva", NULL};
char* cut_args[] = {"cut ", "c4", NULL};
  int pipes[4];
 pipe(pipes);
 pipe[pipes+2];

        1st child
 if (fork() == 0) { must close pipes so parent can access default i/o
     dup2(pipe[1], 1);
```

```
        close(pipes[0]);
        close(pipes[1]);
        close(pipes[2]);
        close(pipes[3]);
        execvp(*cat_args, cat_args);
    }
    else {     2nd child
        if ( fork==0){
            dup2(pipes[0], 0); read 1
            dup2(pipes[3], 1); write 2
            close(pipes[0]);
            close(pipes[1]);
            close(pipes[2]);
            close(pipes[3]);
            execvp(*grep_args, grep_args);
            }
            else {
                if(fork()==0){
                    dup2(pipes[2], 0);
                    close(pipes[0]);
                    close(pipes[1]);
                    close(pipes[2]);
                    close(pipes[3]);
                    execvp(*cut_args, cut_args);
                    }
```



```
        }
    }
     for(i=0;i<3;i++){
        wait(&status);
     }
```



install a phone jack = create a socket

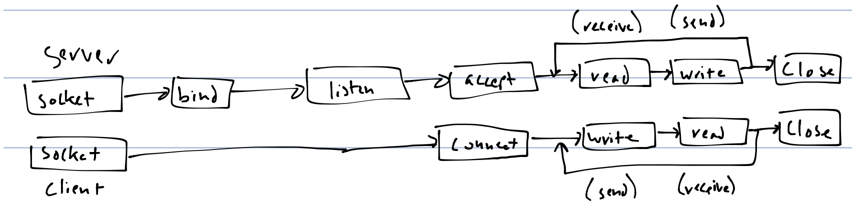
get a phone number = bind an address to the socket

listen to the phone see if it rings = listen to the address for an incoming request (LISTEN)

pick up the phone = accept the incoming request (ACCEPT)

talk on the phone = communicate with the requester (SEND/RECEIVE)

hang up the phone = disconnect the communication (CLOSE)


server
socket
bind
listen
accept
(receive)
(send)
read
write
close
socket
client
connect
write
read
close
(send)
(receive)