ASP activity 1

Groupmembers

1. Emily Namm
2. Emmett Kogan

*Not including code to check for errors

Case 1 - prog args > file

```
pid = fork();                           // fork shell process
if (!pid) {                                     // if child process
        fd = open();                    // open file
        dup2(fd, 1);                    // set file to replace stdout
        execl(prog, prog, args..., NULL);       // exec prog w/ args
}

// Wait for all children to die before continueing
while(wait(NULL) > 0);
```

Case 2 - prog args < file

```
pid = fork();                           // fork shell process
if(!pid) {                                      // if child process
        fd = open();                    // open file
        dup2(fd, 0);                    // set file to replace stdin
        execl(prog, prog, args..., NULL);       // exec prog w/ args
}

// Wait for all children to die before continueing
while(wait(NULL) > 0);
```

Case 3 - prog1 args1 | prog2 args2

```
int pfd[2];
pipe(pfd);

int pid = fork();
if (!pid) {
        // Child 1
        close(pfd[0]);
        dup2(pfd[1], 1);
        execl(prog1, prog1, args1..., NULL);

        close(pfd[1]);
        exit(0);
}

pid = fork();
if (!pid) {
        // Child 2
        close(pfd[1]);
        dup2(pfd[0], 0);
        execl(prog2, prog2, args2..., NULL);

        close(pfd[0]);
        exit(0);
}

close(pfd[0]);
close(pfd[1]);

// Wait for all children to die before continueing
while(wait(NULL) > 0);
```

Case 4 - prog1 args1; prog2 args2

```
if(!fork()) {
        // Child 1
        if(!fork()) {
                // Child 2
                execl(prog1, prog1, args1..., NULL);
                exit(0);
        }

        while(wait(NULL) > 0);
        execl(prog2, prog2, args2..., NULL);
        exit(0);
}

// Wait for children to terminate
while(wait(NULL) > 0);
```

Case 5 - prog1 args1 && prog2 args2

```
if(!fork()) {
        // Child 1

        int wstatus;
        int pid = fork();

        if(pid == 0) {
                // Child 2
                execl(prog1, prog1, args1..., NULL);
                exit(0);
        }

        waitpid(pid, &wstatus, 0);
        if (WIFEXITED(wstatus))
                execl(prog2, prog2, args2..., NULL);

        exit(0);
}

// Wait for children to terminate
while(wait(NULL) > 0);
```

Case 6 - prog1 args1 || prog2 args2

```
if(!fork()) {
        // Child 1
        int wstatus;
        int pid = fork();

        if(pid == 0) {
                // Child 2
                execl(prog1, prog1, args1..., NULL);
                exit(0);
        }

        waitpid(pid, &wstatus, 0);
        if (!WIFEXITED(wstatus))
                execl(prog2, prog2, args2..., NULL);

        exit(0);
}

// Wait for children to terminate
while(wait(NULL) > 0);
```