```
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab04$ ./lab4b
Wait: 198
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab04$
```

```c
/*****************************************
 *
 *               lab2a
 *
 *  This program illustrates the use of the
 *  execve system call.
 *
 *****************************************/

#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
#include <string.h>
#include <errno.h>
#include <sys/wait.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>


extern char **environ;

int main(int argc, char **argv) {
    int pid;
    int ret;
    int status;

    if((pid = fork())) {
        if(pid < 0) {
            printf("Fork error: %s\n",strerror(errno));
            exit(1);
        }
        printf("Wait: %d\n", wait(&status));
    } else {
        int fin = open(argv[1], O_RDONLY);
```

```
        int fout = open(argv[2], O_WRONLY | O_CREAT | O_TRUNC, 0644);
        dup2(fin, 0);
        dup2(fout, 1);
        ret = execve("lab4a", argv, environ);
        if(ret < 0) {
            printf("Execve failed: %s\n", strerror(errno));
            exit(1);
        }
    }

    exit(0);
}

/*****************************************
 *
 *              lab1.c
 *
 *  A simple copy program that demostrates
 *  basis system calls.
 *
 *  Usage: lab1 infile outfile
 *
 *****************************************/

#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <errno.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char **argv) {
    int fin;
    int fout;
    int n;
    char buffer[512];
    int ret;
```

```
    if(argc != 3) {
        printf("Usage: lab1 infile outfile\n");
        exit(1);
    }

    fin = 0;
    if(fin < 0) {
        printf("Can't open input file: %s\n",strerror(errno));
        exit(1);
    }

    fout = 1;
    if(fout < 0) {
        printf("Can't open output file: %s\n",strerror(errno));
        exit(1);
    }

    //My Code
    // char buf1[100] = "/Lab01/out.txt";
    // struct stat buf;
    // fstat(fin, &buf);
    // int statchmod = buf.st_mode & (S_IRWXU | S_IRWXG | S_IRWXO);
    // printf("Input File Permissions: %o\n", statchmod);
    // chmod(buf1, statchmod);

    // struct stat buf2;
    // fstat(fout, &buf2);
    // int statchmod1 = buf2.st_mode & (S_IRWXU | S_IRWXG | S_IRWXO);
    // printf("Output File Permissions: %o\n", statchmod1);

    n=1;  // Get the process started
    while(n > 0) {
        n = read(fin, buffer, 512);
        if(n < 0) {
            printf("Error on read: %s\n",strerror(errno));
            exit(1);
        }
        ret = write(fout, buffer, n);
        if(ret < 0) {
            printf("Erroc on write: %s\n",strerror(errno));
```

```
            exit(1);
        }
    }

    close(fin);
    close(fout);
    exit(0);
}
```