CSE 3100 Systems Programming
Lab #6

Welcome to Lab 6! At the start of each lab, you will receive a document like this detailing the tasks you are to complete. Read it carefully. Try your best to finish labs during the lab session. However, labs are not due right after each lab period.

1 Reading and Writing

In this section we will create two general functions that can read and write strings of characters to a file descriptor. Since these strings can be of variable length, we will have to read or write one character at a time.

1.1 Writing

The `write_message` function will iterate through the string that is stored in `message` and write one character at a time to the file descriptor `fd` using the built in `write` function. We can easily iterate through the entire string without going beyond the end of the string by finding the length of the string using the `strlen` function. Do not write the null terminator. Here is the function signature for `write_message`:

```
void write_message(char * message, int fd);
```

1.2 Reading

The `read_message` function will loop, reading a character from the file descriptor `fd` into a buffer, until a new line character is reached. At this point, a new line character and a null character should be added to indicate the end of the string. Here is the function signature:

```
char* read_message(int fd);
```

You'll notice that we defined `BUFFER_SIZE` at the top of the file. The buffer should initially be of size `BUFFER_SIZE`, but should expand if needed!

2 Pipes

Main is currently broken and you will have to use pipes to fix it! Sounds like you're going to be a plumber today! The `pipe` function can be used to create a one directional pipe. After creating these pipes, we can fork to create sub-processes. The goal of the main function is to create 2 children where child 1 will encrypt such that child 2 can not read a message. The parent function will print a message and send it to child 1 through a pipe. Child 1 will take this message, encrypt it using the `encrypt` function, and send it to child 2 through a pipe. Child 2 will take this encrypted message, print it, and send it back to the parent. At this point the parent will take the encrypted message, decrypt it using the `decrypt` function and print the original message. Don't forget to close the file descriptors!