POSIX I/O

The fun stuff!
POSIX

Posix is a family of standards specified by the IEEE. These standards maintains compatibility across variants of Unix-like operating systems by defining APIs and standards for basic I/O (file, terminal, and network) and for threading.

1) What does POSIX stand for?

   Portable Operating System Interface

1) Why might a POSIX standard be beneficial? From an application perspective? Versus using the C stdio library?

   - More explicit control since read and write functions are system calls and you can directly access system resources.
   - POSIX calls are unbuffered so you can implement your own buffer strategy on top of read()/write().
   - There is no standard higher level API for network and other I/O devices
Review from Lecture

```c
ssize_t read(int fd, void *buf, size_t count)
```

<table>
<thead>
<tr>
<th>Condition</th>
<th>Result Conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>An error occurred</td>
<td>result = -1, errno = error</td>
</tr>
<tr>
<td>Already at EOF</td>
<td>result = 0</td>
</tr>
<tr>
<td>Partial Read</td>
<td>result &lt; count</td>
</tr>
<tr>
<td>Success!</td>
<td>result == count</td>
</tr>
</tbody>
</table>
New Scenario - Messy Roommate

- The Linux kernel is now your roommate
- There are N pieces of trash in the room
- There is a single trash can, `char bin[N]`
  - (For some reason, the trash goes in a particular order)
- You can tell your roommate to pick it up, but he/she is unreliable
New Scenario - Messy Roommate

NumTrash pickup(roomNum, trashCan, Amount)

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Condition</th>
</tr>
</thead>
</table>
| “I tried to start cleaning, but something came up” (got hungry, had a midterm, room was locked, etc.) | NumTrash == -1  
errno == excuse |
| “You told me to pick up trash, but the room was already clean”           | NumTrash == 0                      |
| “I picked up some of it, but then I got distracted by my favorite show on Netflix” | NumTrash < Amount                 |
| “I did it! I picked up all the trash!”                                   | NumTrash == Amount                 |
How do we get the room clean?

- Use a loop. What’s the (high level) goal?
  - Pick up all N pieces of trash
- What if the roommate returns -1 with an excuse?
  - If it’s a valid excuse, stop telling them to pick up trash
  - If it’s not, start over at the top of the loop
- What if the room is already clean?
  - Stop telling the roommate to pick up trash
- What if the roommate only picked up some of it?
  - Record how much they picked up, and tell them to pick up the rest
- What if the roommate picked up everything you asked?
  - Our goal has been reached!

<table>
<thead>
<tr>
<th>NumTrash pickup(roomNum, trashCan, Amount)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumTrash == -1, errno == excuse</td>
</tr>
<tr>
<td>NumTrash == 0</td>
</tr>
<tr>
<td>NumTrash &lt; Amount</td>
</tr>
<tr>
<td>NumTrash == Amount</td>
</tr>
</tbody>
</table>

That’s it!
How do we get the room clean?

```
NumTrash pickup(roomNum, trashCan, Amount)

- NumTrash == -1, errno == excuse
- NumTrash == 0
- NumTrash < Amount
- NumTrash == Amount
```

What do we do in the following scenarios?
How do we get the room clean?

NumTrash pickup(roomNum, trashCan, Amount)

<table>
<thead>
<tr>
<th>Condition</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NumTrash == -1, errno == excuse</td>
<td>Decide if the excuse is</td>
</tr>
<tr>
<td>NumTrash == 0</td>
<td>reasonable, and either</td>
</tr>
<tr>
<td>NumTrash &lt; Amount</td>
<td>let it be or</td>
</tr>
<tr>
<td>NumTrash == Amount</td>
<td>ask again.</td>
</tr>
</tbody>
</table>

I have to study for cse333! I'll do it later.

Decide if the excuse is reasonable, and either let it be or ask again.
How do we get the room clean?

The room is already clean, dawg!

Stop asking them to clean the room! There’s nothing to do.
How do we get the room clean?

NumTrash pickup(roomNum, trashCan, Amount)

- NumTrash == -1, errno == excuse
- NumTrash == 0
- NumTrash < Amount
- NumTrash == Amount

I picked up 3 whole pieces of trash! What more do you want from me?

Ask them again to pick up the rest of it.
How do we get the room clean?

They did what you asked, so stop asking them to pick up trash.

I did it! The whole room is finally clean.
Worksheet Exercise 3

- Write the string `buf` to the file `333.txt`.
- Do not use the `bytes_left` method from lecture.
Worksheet Exercise 7

- Write a C program that is analogous to `ls`.