CSCI-UA.0202-001 Operating Systems

Homework 2

Due: Friday, October 7, 2022

These problems should be done on your own. We're not going to be grading them strictly (we'll mainly look at whether you attempted them). But they will be reinforcing knowledge and skills, so you should totally work through them carefully.

Please save your answers as a single PDF file and upload it to Brightspace.

Question 1

Alice has the impression that "*arrays are the same as pointers*" in C, so she writes the following program to test if that's really the case.

```
#include <stdio.h>

size_t f(const char *c) {
  return sizeof(c);
}

int main() {
  const char a[] = "Hello, CS202!";
  const char *b = "Hello, CS202!";

  printf("size of a is %lu\n", sizeof(a));
  printf("size of b is %lu\n", sizeof(b));
  printf("f(a) returns %lu\n", f(a));
  printf("f(b) returns %lu\n", f(b));
}
```

Alice expects to see all four lines print 14. However, when she runs this program, she sees different results.

(1) Which line(s) prints 14? Why?

(2) What do other line(s) print? Why?

(3) In fact, `sizeof()` shouldn't be used to get the length of a string at all. Why?

Question 2

Bob just learned the `exec` family of system calls in his CS202 class, and he wants to try it out himself. So he writes the following program.

```
#include <stdio.h>
#include <unistd.h>

int main() {
  printf("one");
  execl("/bin/echo", "/bin/echo", "two", NULL);
  printf("three");
}
```

Bob expects to see "one" and "two" printed but not "three". However, when he runs this program on the CIMS compute servers, only "two" is printed. He examines his screen carefully with his magnifying glass but still cannot find any trace of "one". Bob gets so confused, so he turns to you for help.

(1) Can you explain to Bob what's happening here? Can you propose a way to make both "one" and "two" printed?

(2) Can you propose a way to make all "one", "two", and "three" printed?

Question 3

For each of the following system calls, give a condition that causes it to fail: `fork`, `exec`, and `wait`.

Question 4

To a programmer, a system call looks like any other call to a library function. Is it important that a programmer know which library functions result in system calls? Under what circumstances and why?