CSC209H Worksheet: Function Pointers and System Call Error Checking

1. Remember that we can use the name of a function as the pointer to the function. This allows us to create variables that are pointers to functions. The syntax can be a little confusing.

   For the statements below, identify whether the statement is A) a function signature, B) declaration of a function pointer variable, C) assigning the return value of a function to a variable, or D) assigning a pointer to a function to a variable.

   Then label the relevant parts of the statement: variable name, return value, argument(s). Explain to your neighbour what each line means.

   (a) `int simple(char *str, int length);`

   A

   (b) `int (*x)(char *s, int l);`

   B

   (c) `int z;`
   `z = simple("abc", 30)`

   C

   (d) `x = simple;`

   D

   (e) `int (*complex(int index))(char *s, int l);`

   A

   The function complex() has an int parameter and returns a function pointer to a function that has parameter (char *, int) and returns an int.

   (f) `int (*y)(char *s, int z) = complex(2);`

   B

   C / D

2. Add the error checking for the following calls. Discuss with your neighbour what the possible errors from the following system calls. (Feel free to cheat by reading the man page.) How important is it to check for errors? Should the program exit immediately?

```
FILE *fp;

fp = fopen(argv[1], "r");
if (fp == NULL) {
    perror("fopen");
    exit(1);
}
int num;

if (fread(&num, sizeof(int), 1, fp) == 0) {
    perror("fread");
    exit(1);
}

char *str;

str = malloc(sizeof(char) * 1024);
if (str == NULL) {
    perror("malloc");
    exit(1);
}
```