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 This is a declaration or signature of the function `simple` which takes a string and an int and returns an int. It does not execute or define the function.

   (b) `int (*x)(char *s, int l);`
   B This is a declaration of a variable named `x` whose type is pointer to function. The functions to which `x` can point must take a string and an int and return an int.

   (c) `int z;`
   `z = simple("abc", 30)`
   C This calls the function `simple` and assigns its return value to `z`.

   (d) `x = simple;`
   D This assigns the function `simple` to variable `x`. It does not execute the function.

   (e) `int (*complex(int index))(char *s, int l);`
   A This is a declaration or signature of the function `complex` which takes an int and returns a function pointer. The function pointer it returns can point to functions that take a string and an int and return an int.

   (f) `int (*y)(char *s, int z) = complex(2);`
   B The left-hand side of this statement is a declaration of a variable named `y` whose type is pointer to function. The functions to which `y` can point must take a string and an int and return an int.
   C The right-hand side of this statement calls the function `complex` with the argument 2 and then the '=' assigns the return value from `complex` to `y`.
   D Since `complex` returns a function pointer, D is also a correct answer here.

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");




int num;

fread(&num, sizeof(int), 1, fp);




char *str;

str = malloc(sizeof(char) * 1024);
```

   SOLUTIONS: See the file `check_errors.c`