P-Question 2.2: A Simple Program Starter

Use the files in folder **p2** of the assignment template. You may only modify and upload the file `run_program.c`.
An important feature of every shell is to start external programs.

a. Write a function with the following features: **4 P-pt**

- Starts a program that may be specified by its full path and name (i.e., /usr/bin/who) or only by its name if it is located in one of the directories contained in the `PATH` environment variable. Do not use `system()`.
- Passes the supplied arguments on to the new process.
  Note 1: `args` contains a list of pointers to arguments. The end of the list is indicated with a NULL pointer.
  Note 2: `args[0]` is the first real argument (this is different to `argv` vector in a Linux system call, in which `argv[0]` has a special meaning and `argv[1]` is the first real argument).
- Waits for the newly created process to exit.
- Returns the special error value 255 to report an error condition and 0 to indicate success.

```
int run_program(char *file_path, char *args[]);
```

- If `file_path` is `NULL`, you should return the special error value. If `args` is `NULL`, you should run the program without arguments. (Your program starter should not be terminated by an exception in either of the two cases.)

b. Modify the starter to return the exit status of the previously started/exited process. Keep a return value of 255 to indicate error conditions in your own program. **2 P-pt**

Note: As explained on Piazza, additional test cases are available on skel. You can run these tests with the command
`/home/sty24/bin/runStudTests.sh A2/p2` *`folderwithyourp2solution`*
These tests cover more cases than the main.c program in the template. They are similar to the tests we will use for evaluating your solution (but there is no guarantee that passing all tests will give you full score for the assignment – we may add additional tests and/or manually inspect your code if it indeed correctly solves the problem).

Total:
10 T-pt
10 P-pt