Running Unix commands

```
gert@gert-Latitude-5290:~/Documents/Spring-2024/Systems-Programming/CS43203-Systems-Programming/StallsmithGarrett-CS43203-smsh/Code$ gcc smsh.c -o smsh
gert@gert-Latitude-5290:~/Documents/Spring-2024/Systems-Programming/CS43203-Systems-Programming/StallsmithGarrett-CS43203-smsh/Code$ ./smsh
smsh$ ls
smsh  smsh.c
smsh$ pwd
/home/gert/Documents/Spring-2024/Systems-Programming/CS43203-Systems-Programming/StallsmithGarrett-CS43203-smsh/Code
smsh$ cd
smsh$ pwd
/home/gert
smsh$ cd /home/gert/Documents/Spring-2024/Systems-Programming/CS43203-Systems-Programming/StallsmithGarrett-CS43203-smsh/Code
smsh$ cat smsh.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/wait.h>

#define MAX_LINE_SIZE 1024
#define MAX_AGUMENTS 1024

// Read commands found in Shell
void read_command(char *command) {
    // Pass command as pointer so we can edit memory address with no need to return a value

    // Get command from standard input
    // As long as command is smaller than MAX_LINE_SIZE
    fgets(command, MAX_LINE_SIZE, stdin);

    // Searches for first instance of newline character ("\n")
    // Knowing this, we will know where the command ends
    // We now know the entire command
    // Remove the newline for execution by replacing it with a null terminator
    command[strcspn(command, "\n")] = '\0';
}

// Parse all arguments found in command
void parse_command(char *command, char *args[]) {
    char *token;    // Define character for tokenization
```

Running arbitrary programs

```
smsh$ python3
Python 3.11.6 (main, Oct  8 2023, 05:06:43) [GCC 13.2.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> i = 0
>>> for i in range(i, 10):
...     print(i)
...
0
1
2
3
4
5
6
7
8
9
>>>
```

Running script file

```
gert@gert-Latitude-5290:~/Documents/Spring-2024/Systems-Programming/CS43203-Systems-Programming/StallsmithGarrett-CS43203-smsh/Code$ ./smsh ../script.txt
RUNNING: cat smsh.c

#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <sys/wait.h>

#define MAX_LINE_SIZE 1024
#define MAX_AGUMENTS 1024

// Read commands found in Shell
void read_command(char *command) {
    // Pass command as pointer so we can edit memory address with no need to return a value

    // Get command from standard input
    // As long as command is smaller than MAX_LINE_SIZE
    fgets(command, MAX_LINE_SIZE, stdin);

    // Searches for first instance of newline character ("\n")
    // Knowing this, we will know where the command ends
    // We now know the entire command
    // Remove the newline for execution by replacing it with a null terminator
    command[strcspn(command, "\n")] = '\0';
}
```