George Suarez

CSE 460

Homework 2

1. Write a simple shell that is similar to what we have discussed in class but contains enough code that it actually works so you can test it.
   For simplicity, you may assume that all commands are in the directory */bin*.

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

void read_command(char cmd[], char *par[])
{
    char line[1024];
    int count = 0, i = 0, j = 0;
    char *array[100], *pch;

        // Read one line
    for (;;)
    {
        int c = fgetc(stdin);
        line[count++] = (char)c;
        if (c == '\n')
            break;
    }
    if (count == 1)
        return;
    pch = strtok(line, " \n");

    // parse the line into words
    while (pch != NULL)
    {
        array[i++] = strdup(pch);
        pch = strtok(NULL, " \n");
```

```
    }
    // first word is the command
    strcpy(cmd, array[0]);

    // others are parameters
    for (int j = 0; j < i; j++)
        par[j] = array[j];
    par[i] = NULL; // NULL-terminate the parameter list
}

void type_prompt()
{
    static int first_time = 1;
    if (first_time)
    { //clear screen for the first time
        const char *CLEAR_SCREE_ANSI = " \e[1;1H\e[2J";
        write(STDOUT_FILENO, CLEAR_SCREE_ANSI, 12);
        first_time = 0;
    }

    printf("> "); // display prompt
}

int main()
{
    char cmd[100], command[100], *parameters[20];
    // environment variable
    char *envp[] = {(char *)"PATH=/bin", 0};
    while (1)
    {                                   //repeat forever
        type_prompt();                  //display prompt on screen
        read_command(command, parameters); // read input from terminal
        if (fork() != 0)                // parent
            wait(NULL);                 //wait for child
        else
        {
            strcpy(cmd, "/bin/");
```

```
            strcat(cmd, command);
            execve(cmd, parameters, envp); // execute command
        }
    }
}
```

Output:

```
> ls -l
total 80
-rwxr-xr-x  1 georgesuarez  staff  16592 Apr 29 18:58 fcfs
-rw-r--r--  1 georgesuarez  staff   2655 Apr 29 18:58 fcfs.cpp
-rwxr-xr-x  1 georgesuarez  staff   9192 Apr 29 17:17 simple_shell
-rw-r--r--  1 georgesuarez  staff   1720 Apr 29 18:38 simple_shell.cpp
> ps -l
  UID   PID  PPID       F CPU PRI NI       SZ    RSS WCHAN     S          ADDR TTY          TIME CMD
  501 46935 46934    4006   0  31  0  4296240   1704 -       S              0 ttys000    0:00.04 -bash
  501 47024 46935    4006   0  31  0  4269776    836 -       S+             0 ttys000    0:00.00 ./simple_shell
> ls
fcfs                    fcfs.cpp                simple_shell            simple_shell.cpp
```

2. Suppose that the following processes arrive for execution at the times indicated. Each process will run the listed amount of time. In answering the questions, use non-preemptive scheduling and base all decisions on the information that you have at the time the decision must be made.

Process	Arrival Time	Burst Time
P1	0.0	6
P2	0.4	4
P3	1.0	2

a. What is the average **waiting** time for these processes with the FCFS scheduling algorithm?

$$Avg.Waiting\ Time = \frac{(9 + 5.6 + 0)}{3} = 4.86$$

b. What is the average **waiting** time for these processes with the SJF scheduling algorithm?

$$Avg.\ Waiting\ Time = \frac{(3.6 + 9 + 0)}{3} = 4.20$$

c. The SJF algorithm is supposed to improve performance but notice that we chose to run process P1 at time 0 because we did not know that two shorter processes would arrive soon. Compute what the average **waiting** time will be if the CPU is left idle for the first 1 unit, and then SJF scheduling is used. Remember that processes P1 and P2 are waiting during this idle time, so their waiting time may increase. This algorithm could be known as future-knowledge scheduling.

$$Avg. Waiting\ Time = \frac{3.6 + 9}{2} = 6.30$$

3. Consider the following set of processes, with the length of the CPU-burst time given in milliseconds.

Process	Burst Time	Priority
P1	8	3
P2	1	1
P3	2	3
P4	1	4
P5	4	2

The processes are assumed to have arrived in the order P1, P2, P3, P4, P5, all at time 0.

a. Draw four Gantt charts that illustrate the execution of these processes using FCFS, SJF, a non-preemptive priority (a smaller number implies higher priority), and RR (quantum = 1) scheduling.

<u>FCFS</u>

	P1	P2	P3	P4	P5
0	8	9	11	12	16

SJF

	P2	P4	P3	P5	P1
0	1	2	4	8	16

Non-Preemptive

	P4	P1	P3	P5	P2
0	1	9	11	15	16

RR

	P1	P2	P3	P4	P5	P1	P3	P5	P1	P5	P1	P5	P1	P1	P1	P1
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

b. Calculate the turnaround time of each process for each of the scheduling algorithms in part a).

Turn Around Time = Burst Time – Arrival time

FCFS

Process	Burst Time	Arrival Time	Turn Around Time
P1	8	0	8
P2	1	0	9
P3	2	0	11
P4	1	0	12
P5	4	0	16

SJF

Process	Burst Time	Arrival Time	Turn Around Time
P1	8	0	16
P2	1	0	1
P3	2	0	4
P4	1	0	2
P5	4	0	8

Non-Preemptive

Process	Burst Time	Arrival Time	Turn Around Time
P1	8	0	9
P2	1	0	16
P3	2	0	11
P4	1	0	1
P5	4	0	15

RR

Process	Burst Time	Arrival Time	Turn Around Time
P1	8	0	16
P2	1	0	2
P3	2	0	7
P4	1	0	4
P5	4	0	12

c. Calculate the waiting time of each process for each of the scheduling algorithms in part a).

FCFS

Process	Burst Time	Arrival Time	Turn Around Time	Waiting Time
P1	8	0	8	0
P2	1	0	9	8
P3	2	0	11	9
P4	1	0	12	11
P5	4	0	16	12

SJF

Process	Burst Time	Arrival Time	Turn Around Time	Waiting Time
P1	8	0	16	8
P2	1	0	1	0
P3	2	0	4	2
P4	1	0	2	1
P5	4	0	8	4

Non-Preemptive

Process	Burst Time	Arrival Time	Turn Around Time	Waiting Time
P1	8	0	9	1
P2	1	0	16	15
P3	2	0	11	9
P4	1	0	1	0
P5	4	0	15	11

RR

Process	Burst Time	Arrival Time	Turn Around Time	Waiting Time
P1	8	0	16	8

P2	1	0	2	1
P3	2	0	7	5
P4	1	0	4	3
P5	4	0	12	8

d. Which of the schedules in part a) results in the minimal average waiting time (over all processes)?

FCFS

$$Avg.Waiting\ Time = \frac{(0 + 8 + 9 + 11 + 12)}{5} = \frac{40}{5} = 8$$

SJF

$$Avg.Waiting\ Time = \frac{(8 + 0 + 2 + 1 + 4)}{5} = \frac{15}{5} = 3$$

Non-preemptive

$$Avg.Waiting\ Time = \frac{(1 + 15 + 9 + 0 + 11)}{5} = \frac{36}{5} = 7.2$$

RR

$$Avg.Waiting\ Time = \frac{(8 + 1 + 5 + 3 + 8)}{5} = \frac{25}{5} = 5$$