

Andre J Plath

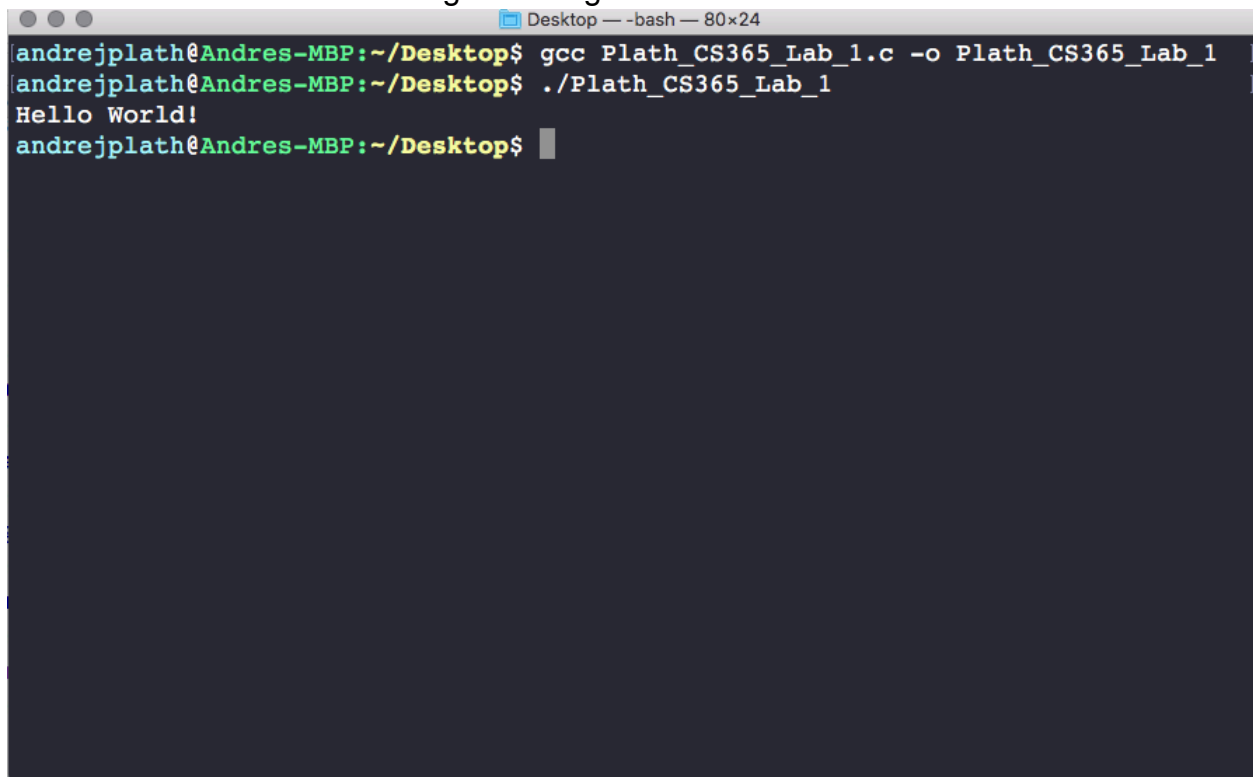
Professor Yanwei Wu

CS365 Operating Systems and Networking

January 18, 2018

CS365 Lab 1

1. Part IV: Hello World Program using c.

A terminal window titled "Desktop — -bash — 80x24" with a dark background. The prompt is "andrejplath@Andres-MBP:~/Desktop\$". The first command is "gcc Plath_CS365_Lab_1.c -o Plath_CS365_Lab_1", which is followed by a new prompt. The second command is "./Plath_CS365_Lab_1", which is followed by the output "Hello World!" and a new prompt. The third prompt is "andrejplath@Andres-MBP:~/Desktop\$" followed by a cursor.

```
andrejplath@Andres-MBP:~/Desktop$ gcc Plath_CS365_Lab_1.c -o Plath_CS365_Lab_1
andrejplath@Andres-MBP:~/Desktop$ ./Plath_CS365_Lab_1
Hello World!
andrejplath@Andres-MBP:~/Desktop$
```

2. Part V: Created a small simple fork() program. I using Virtual box so my terminal output looks strange.

The image shows two windows from a virtual machine. The top window is a terminal titled 'CS365_Ubuntu [Running]'. It shows the execution of a C program. The user runs 'gcc Plath_CS365_Lab_1_Part_2.c' and then './a.out'. The output shows a sequence of 'Hello from Parent Process!' and 'Hello from Child Process!' messages with their respective process IDs (27688, 27691, 27690, 27689, 27694, 27693, 27692, 27695). The bottom window is a code editor titled 'Plath_CS365_Lab_1_Part_2.c'. It contains the source code for the program, which uses 'fork()' to create child processes and 'getpid()' to print the process ID.

```
andre@andre-VirtualBox:~/Downloads$ gcc Plath_CS365_Lab_1_Part_2.c
andre@andre-VirtualBox:~/Downloads$ ./a.out
Hello from Parent Process! Process id is 27688
andre@andre-VirtualBox:~/Downloads$ Hello from Child Process! Process id is 27691
Hello from Parent Process! Process id is 27690
Hello from Parent Process! Process id is 27689
Hello from Child Process! Process id is 27694
Hello from Parent Process! Process id is 27693
Hello from Child Process! Process id is 27692
Hello from Child Process! Process id is 27695
```

```
1  #include<stdio.h>
2  #include<sys/types.h>
3  #include<unistd.h>
4
5  void forkprogram()
6  {
7
8      // child process because return value zero
9      if (fork() == 0)
10         printf("Hello from Child Process! Process id is %d\n", getpid());
11
12     // parent process because value return is non-zero
13     else
14         printf("Hello from Parent Process! Process id is %d\n", getpid());
15 }
16
17
18 int main()
19 {
20     fork();
21     fork();
22     forkprogram();
23     return 0;
24 }
25
```