ICS EXE 3

March 24, 2020

1 Process

Assume we have the following code:

```c
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int x = 10;
int main() {
  int y = 1, i;
  for (i = 0; i < 2; ++i) {
    pid_t pid = fork();
    if (!pid) {
      x += y;
      printf("x in child: %d\n", x);
      exit(0);
    }
    y++;
    x <<= 1;
    printf("x in parent: %d\n", x);
  }
  return 0;
}
```

1.1

Give a possible output of the program

1.2

Based on Q1.1, how to use waitpid to make sure the "x in child" and the "x in parent" are interleaved? Will this influence the value of x in output?

1.3

Based on Q1.1, please list all possible output order. (or show the constraints on the output order).

1.4

Based on Q1.1, if we accidentally delete the line 14, please give a possible output.

2 System Call

Consider the scenario where you are coding the lab. You have thought for a while without operating on your editor. After a key is pressed on keyboard, it displays in your screen. You can view this procedure as the combination of executing 'getchar' and 'printf'. What are the system calls the two functions finally issue? Can you describe how control flow switches between user mode and kernel mode in this procedure?

3 Waitpid and Zombie

Assume we have the following code

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <unistd.h>
4  #include <sys/types.h>
5  #include <sys/wait.h>
6
7  int main(void) {
8    if (fork() == 0) {
9      if (fork() == 0) { // Proc 1
10       exit(0);
11     } else { // Proc 2
12       waitpid(-1, NULL, 0);
13       exit(0);
14     }
15   } else { // Proc 3
16     while (waitpid(-1, NULL,
17                    WNOHANG) > 0) ;
18     exit(0);
19   }
20   return 0;
21 }
```

3.1

How many zombie processes could the program leave for init to reap? Please show all possible number and show one execution order for each. (Assume that shell will not reap the main process)

3.2

What if we remove the line 12?