

## Homework Assignment #3

Due: Sunday, September 22, 2021

1. Using the program shown in below, explain what the output will be at LINE A

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
int value = 5;
int main()
{
    pid_t pid;
    pid = fork();
    if (pid == 0) { /* child process */
        value += 15;
        return 0;
    }
    else if (pid > 0) { /* parent process */
        wait(NULL);
        printf("PARENT: value = %d",value); /* LINE A */
        return 0;
    }
}
```

1. When a process creates a new process using the fork() operation, which of the following states is shared between the parent process and the child process?
  - a. Stack
  - b. Heap
  - c. Shared memory segments

2. Describe the actions taken by a kernel to context-switch between processes.

3. What are the zombie process and orphan process respectively?

4. How many times does this code print "hello"? and why?

```
#include <stdio.h>
#include <unistd.h>
void main(int argc, char**argv) {

    int i;
    for (i = 0; i < 3; i++){
        fork();
        printf("hello\n");

    }
}
```

5. Using the program shown in below, explain what the output will be at lines X and Y.

```
#include <sys/types.h>
#include <stdio.h>
#include <unistd.h>
#define SIZE 5
int nums[SIZE] = {0,1,2,3,4};
int main()
{
    int i;
    pid_t pid;
    pid = fork();
    if (pid == 0) {
        for (i = 0; i < SIZE; i++) {
            nums[i] *= -i;
            printf("CHILD: %d ",nums[i]); /* LINE X */
        }
    }
    else if (pid > 0) {
        wait(NULL);
        for (i = 0; i < SIZE; i++)
            printf("PARENT: %d ",nums[i]); /* LINE Y */
    }
    return 0;
}
```

6. The Collatz conjecture concerns what happens when we take any positive integer  $n$  and apply the following algorithm:

$$n = \begin{cases} n/2, & \text{if } n \text{ is even} \\ 3 \times n + 1, & \text{if } n \text{ is odd} \end{cases}$$

The conjecture states that when this algorithm is continually applied, all positive integers will eventually reach 1. For example, if  $n = 35$ , the sequence is

35, 106, 53, 160, 80, 40, 20, 10, 5, 16, 8, 4, 2, 1

Write a C program using the `fork()` system call that generates this sequence in the child process. The starting number will be provided from the command line. For example, if 8 is passed as a parameter on the command line, the child process will output 8, 4, 2, 1. Because the parent and child processes have their own copies of the data, it will be necessary for the child to output the sequence. Have the parent invoke the `wait()` call to wait for the child process to complete before exiting the program. Perform necessary error checking to ensure that a positive integer is passed on the command line.