Homework 1 for CS153 (Fall 2017)

Due: Tuesday 10/17

Instructions:

* Be brief. You will be graded for correctness, not on the length of your answers.

* Make sure to write legibly. Incomprehensible writing will be assumed to be incorrect.

I. Consider a single CPU system with an active process A. Explain what happens in the following circumstances including any interrupts, system calls, etc., and how they are handled until a process is back to running again (4.5 points)

a) Process A forks a new process B
b) Process A is running, and needs to read from a file
c) A timer interrupt occurs while A is running.

II. Which of the following requires a system call? Explain. (4.5 points)

a) A process needs to call a function in a library that computes a random number
b) A process needs to send a network packet
c) A process needs to send a signal (software interrupt) to its child

III. Consider the following program:

```c
int count = 0; //shared variable since its global

void twiddledee() {
    int i=0;
    for (i=0; i<2; i++) {
        count = count * count; //assume count read from memory once here
    }
}

void twiddledum() {
    int i=0;
    for(i=0; i<2; i++) { count = count - 2; }
}

void main() {
    thread_fork(twiddledee); thread_fork(twiddledum);
    //wait for both threads to finish
    print count;
}
```
1. What are the min. and max. values that could be printed in main? (4 points)
2. Repeat part 1 considering that i is also a shared variable (Bonus 3 points – tricky!)
3. Describe a potential schedule of execution that will result in the value printed out being equal to 0. Assuming there is only one CPU core, clearly specify when the transitions between the Ready and Running states occurs for each thread in this execution. (5 points)

IV. Consider the following program:

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

int main () {
    pid_t pid1=0, pid2=0;
    int i=3;
    pid1=fork();
    if( pid1 != 0 ) {
        printf("%d\n", ++i);
        pid2 = fork();
        if(pid2 != 0) {
            waitpid(pid1, NULL, 0);
            printf("%d\n", ++i);
            waitpid(pid2,NULL, 0);
            printf("%d\n", ++i);
        } else {
            fork();
            printf("%d\n", ++i);
        }
        exit(0);
    } else {
        printf("%d\n", ++i);
    }
    printf("%d\n", ++i);
}
```

a. How many processes are created during the execution of this program? Explain. (2 points)

b. List all the possible outputs of the program (4 points)