CS 350 Operating Systems, Fall 2018
Homework Assignment 1

Out: 9/15/2018 Sat.
Due: 9/22/2018 Sat. 23:59:59

Kevin Andrade

Full points: 100

1. (15 points) Let's assume we have a system with a single CPU. There are three processes (P1, P2, and P3) running the following logic. Process state can be one of the three states: READY, RUNNING, and WAITING, which behave in the way as discussed in class. Fill in the blanks with the states of each process below. If the process does not exist yet, put a "X" in the blank.

	Process state		
	`P1`	`P2`	`P3`
① P1 is loaded into memory and starts executing in `main()`.	running	x	x
② P1 calls `fork()` and creates P2, but P1, the parent, keeps running.	running	Ready	x
③ P1 issues an I/O request; P2 starts executing at the return from `fork()`.	waiting	running	x
④ P2 calls `fork()` to create P3; P2 keeps running.	waiting	running	Ready
⑤ P2's time slice expires; P3 starts running.	waiting	Ready	running
⑥ P1's I/O completes (but there is no other changes)	Ready	Ready	running
⑦ P3 waits for user input. P1 runs.	running	ready	waiting

2. (10 points) Besides the three common process states, i.e., READY, RUNNING, and WAITING, there is a separate TERMINATED state for exited/terminated processes. Can we remove processes from the system when they exit or are terminated, thereby eliminating the need of having the TERMINATED state? Why?

3. (30 points) Answer the questions related to the code below. Note: there is no buffering for the `printf`s.

```
int main(void) {
    int i = 0;
```

```
    printf("A");
    i++;
    fork();
    printf("B");
    i++;
    fork();
    printf("C");
    i++;
    fork();
    printf("D");
    i++;
    return 0;
}
```

(1) (5 points) Is "ABCBCDCDCDDDDDD" a feasible output? If no, why?

(2) (5 points) Is "ABBCDCCDDCDDDDD" a feasible output? If no, why?

(3) (5 points) Is "ABCDDCDDBCDDDCD" a feasible output? If no, why?

(4) (5 points) Is "ABCCDDCBCDDDDDD" a feasible output? If no, why?

(5) (5 points) Is "ABCBDCCDCDDDDDD" a feasible output? If no, why?

(6) (5 points) How many processes are created (including the top level process) with the above code? What is the largest value of `i` among all the processes before they exit?

4. (15 points) Answer the questions related to the code below. Note: there is no buffering for the `printf`s.

```
int main(void) {
    int i = 0;
    printf("A");
    i++;
    if (fork() > 0) wait(NULL);
    printf("B");
    i++;
    if (fork() > 0) wait(NULL);
    printf("C");
    i++;
    if (fork() > 0) wait(NULL);
    printf("D");
    i++;
    return 0;
}
```

(1) (10 points) List all the possible outputs of the above code. If there are more than three, just list three.

(2) (5 points) How many processes are created (including the top level process) with the above code? What is the largest value of `i` among all the processes before they exit?

5. (30 points) Try the homework questions 4, 5, 6, 7, and 8 in OSTEP-4 (`https://bit.ly/2xdSx7R`). The needed "`process-run.py`" simulator can be downloaded at: `https://bit.ly/2xmzHL5`.

(Note: you may need to read the README file of the simulator and play with it, for example, try the questions 1-3, to understand how it works.)

(1) (15 points) When a process P issues an I/O, the OS has two options:

- one is switching to another process while P is doing I/O (i.e., the "-S SWITCH_ON_IO" option in the simulator); and
- the other is not switching but waiting until P is completely finished (i.e., the "-S SWITCH_ON_END" option in the simulator).

Which option is better? Why?

(2) (15 points) When an I/O of process P completes, the OS also has two options:

- one is immediately scheduling P to run on the CPU (i.e., the -I IO_RUN_IMMEDIATE option in the simulator); and
- the other is continue running the process that is on the CPU at the time when P's I/O completes (i.e., the -I IO_RUN_LATER option in the simulator).

Is the IO_RUN_IMMEDIATE option always better than the IO_RUN_LATER option? If yes, why? If no, in what circumstances IO_RUN_IMMEDIATE is better than IO_RUN_LATER?

Submission instructions

1. Type your answers using whatever text editor you like, remember to include the index number of each question.
2. Export the file to PDF format.
3. Name the PDF file based on your BU email ID. For example, if your BU email is "abc@binghamton.edu", then the PDF file should be named as "hw1_abc.pdf".
4. Not following the above instructions will lead to 5 points off.
5. Submit the PDF file to myCourses website before the deadline.