CS 451 Exam

My Name Is:

My Email Is:

1. Show The PCBs

1. Process ID 2 executes the following code. Assuming there are no errors, show the contents of the initial and any new PCBs after the code runs. The initial PCB for PID 2 is given below. Assign new processes a PID using any consistent means you like.

```
int pid;
pid = fork();
if (pid == 0) {
   /* child */
   pid = fork();
   sleep_forever();
} else if (pid > 0) {
   /* parent */
   chdir("/home");
   pid = fork();
   sleep_forever();
}
```

CODE

PID:	2
Parent PID:	1
User ID:	47
Current Working Dir:	"/"

INITIAL PCB

PID:	2
Parent PID:	1
User ID:	47
Current Working Dir:	"/home"

PID:	3
Parent PID:	2
User ID:	47
Current Working Dir:	"/"

PID:	5
Parent PID:	2
User ID:	47
Current Working Dir:	"/home"

PID:	4
Parent PID:	3
User ID:	47
Current Working Dir:	"/"

2. Show The Output Of This Program

```
int main(int argc, char** argv) {
  printf("main ");
  foo();
  return 0;
}

sthread_ctx_t *g1 = NULL;
sthread_ctx_t *g2;

void foo() {
  int cnt = 0;

  cnt++;
  printf("foo%d ", cnt);

  if (g1 == NULL) {
    g1 = (sthread_ctx_t*)malloc(sizeof(sthread_ctx_t));
    g2 = sthread_new_ctx(foo);
    sthread_switch(g1, g2);
  } else {
    sthread_switch(g2, g1);
  }

  printf("!foo%d ", cnt);
}
```

main foo1 foo1 !foo1

3

3. Scheduling

Suppose that the following processes arrive for execution at the times indicated. Each process will run the listed amount of time. Assume non-preemptive scheduling.

Process	Arrival Time	Burst Time
P1	0.0	8
P2	0.4	4
P3	1.0	1

1. What is the average turnaround time assuming FCFS?
    1. T1 = 8; T2 = 11.6; T3 = 12. (T1+T2+T3)/3 = 10.53
2. What is the average turnaround time assuming SJF?
    1. T1=8; T2=12.6; T3=8. (T1+T2+T3)/3 = 9.5
3. What is the average turnaround assuming SJF if all jobs arrive at time 0.0.
    1. T1=1; T2=5; T3=13. (T1+T2+T3)/3 = 6.33

4

4. Scheduling (II)

Assume a preemptive system running only CPU intensive jobs for which a preemptive context switch occurs at interval Q and takes time L. Define "efficiency" to be the fraction of time that the system is actually doing useful computation (as opposed to scheduling or switching).

1. Symbolically, show the formula for the system's efficiency.

Q/(Q+L)

5

5. Readers/Writers

Describe the intended scheduling policy of this variant to the readers/writers code discussed in class.

A: Gives priority to writers.

```
INTEGER AW=0, WW=0, AR=0, WR=0;
SEMAPHORE OkToWrite, OkToRead, Lock;
```

```
Acquire Read Lock
        P(lock)
        IF ((AW+WW) == 0) {
                V(OkToRead);
                AR++;
        } else WR++;
        P(OkToRead);

        /* READ DATA */

Release Read Lock
        P(lock)
        AR--;
        if (AR == 0 && WW > 0)  {
                V(OkToWrite);
                AW++;WW--;
        }
        V(lock)
```

```
AcquireWriteLock
        P(lock);
        if (AW + AR + WW == 0)  {
                V(OkToWrite);
                AW++;
        } else WW++;
        V(lock);
        P(OkToWrite);

        /* WRITE DATA */

ReleaseWriteLock
        P(lock);
        AW--;
        if (WW > 0 )  {
                V(OkToWrite);
                AW++; WW--;
        } else while (WR > 0)  {
                V(OkToRead);
                AR++; WR--;
        }
        V(lock);
```

Hint: [A/W][R/W] = [Active/Waiting][Reader/Writer]

6

6. Short Answer

1. Give an example from Operating Systems that demonstrates you understand the difference between policy and mechanism.
    1. Mechanism: Test-and-Set, Policy: Mutex
2. Who were John Atanasoff and Clifford Berry?
    1. Physicist and grad student who built ABC computer at Iowa State.
3. What was technologically novel about the Univac?
    1. Tape
4. Why was Multiprogramming a good idea?
    1. Increase utilization by overlapping IO and computation
5. Why are user programs not allowed to disable interrupts?
    1. Could deny global preemption and defeat multiprogramming.

7