CS4411 Operating Systems HW1 (72 pts)

Problem 1: When a user process is interrupted or causes a processor exception, the x86 hardware switches the stack pointer to a kernel stack, before saving the current process state. Explain why.(2 pts)

Problem 2: Most hardware architectures provide an instruction to return from an interrupt, such as iret. This instruction switches the mode of operation from kernel-mode to user-mode.

1) Explain where in the operating system this instruction would be used. (2 pts)
2) Explain what happens if an application program executes this instruction. (2 pts)

Problem 3: What happens if we run the following program on UNIX? (3 pts)

```
main() {
   while (fork() >= 0)
      ;
}
```

Problem 4: How many processes are created if the following program is run? (3 pts)

```
void main(int argc, char ** argv) {
    forkthem(5)
}
void forkthem(int n) {
   if (n > 0) {
      fork();
      forkthem(n-1);
   }
}
```

Problem 5: “What is the output of the following programs? (Please try to solve the problem without compiling and running the programs.) (5 pts)

```
// Program 1
main() {
   int val = 5;
   int pid;”

  if (pid = fork())
      wait(pid);
```

```
    val++;
    printf("%d\n", val);
    return val;
}

// Program 2:
main() {
    int val = 5;
    int pid;
    if (pid = fork())
        wait(pid);
    else
        exit(val);
    val++;
    printf("%d\n", val);
    return val;
}
```

Problem 6: Given the following mix of tasks, task lengths, and arrival times, compute the completion and response time for each task, along with the average response time for the FIFO, RR, and SJF algorithms. Assume a time slice of 10 milliseconds. (15 pts)

Task	Length	Arrival Time
0	85	0
1	30	10
2	35	15
3	20	80
4	50	85

Problem 7: Three tasks, A, B, and C are run concurrently on a computer system. (15 pts)

- Task A arrives first at time 0, and uses the CPU for 100 ms before finishing.
- Task B arrives shortly after A, still at time 0. Task B loops ten times; for each iteration of the loop, B uses the CPU for 2 ms and then it does I/O for 8 ms.
- Task C is identical to B, but arrives shortly after B, still at time 0.

Assuming there is no overhead to doing a context switch, identify when A, B and C will finish for each of the following CPU scheduling algorithms:

1) FIFO
2) Round robin with a 1 ms time slice
3) Round robin with a 100 ms time slice

4) Multilevel feedback with four levels, and a time slice for the highest priority level is 1 ms.
5) Shortest job first.

Problem 8: Round-robin schedulers normally maintain a list of all runnable processes, with each process occurring exactly once in the list. What would happen if a process occurred twice in the list? Can you think of any reason for allowing this? (5 pts)

Problem 9: Measurements of a certain system have shown that the average process runs for a time T before blocking on I/O. A process switch requires a time S, which is effectively wasted (overhead). For round-robin scheduling with quantum Q, give a formula for the CPU efficiency for each of the following: (20 pts)

(a) $Q = \infty$
(b) $Q > T$
(c) $S < Q < T$ (d) $Q = S$
(e) Q nearly 0