National University of Computer and Emerging Sciences, Lahore Campus

Course:	Operating System	Course Code:	CS-2006
Quiz:	Quiz:1 sample 1	Marks	10
Name:	MS -	Section Roll No.	H

Q:1- Consider the following events that happen during a context switch from (user mode of) process P to (user mode of) process Q, triggered by a timer interrupt that occurred when P was executing, in a Unix-like operating system design studied in class. Arrange the events in chronological order, starting from the earliest to the latest. **[2]**

(A)The CPU program counter moves from the kernel address space of P to the kernel address-space of Q.
(B) The CPU executing process P moves from user mode to kernel mode.
(C) The CPU stack pointer moves from the kernel stack of P to the kernel stack of Q.
(D) The CPU program counter moves from the kernel address space of Q to the user address space of Q.
(E) The OS scheduler code is invoked.

$B \rightarrow E \rightarrow C \rightarrow A \rightarrow D$

Q:2- Consider the following program which uses the fork()and wait() system calls. What is the result written by the program to the standard output stream? Lets suppose parent process have process id= 123 and child process have process id= 345. **[3]**

```
int main()
 {
        int i;
        pid_t pid;
        pid = fork();
        if(pid > 0)
        {
        pid_t w1=wait(NULL);
        printf ("value is %d \n" , w1);
        printf ("Parent starts\n ");
        for (i=2; i<=10; i+=2)
                printf ("%d\n",i);
        printf ("\nParent ends\n");
        }

        else if (pid == 0)
        {
        printf ("Child starts\n ");
        for (i=1; i<10; i+=2)
                printf ("%d \n",i);
        printf ("\nChild ends\n");
        }
}
```

Output:

```
child starts
1
3
5
7
9
child ends
Value is 345
Parent starts
~~[illegible]~~
2
4
6
8
Parent ends
```

Q3:- Consider the following program which uses the fork() system call. What is the result written by the program to the standard output stream? What is the resulting process tree? [2+3]

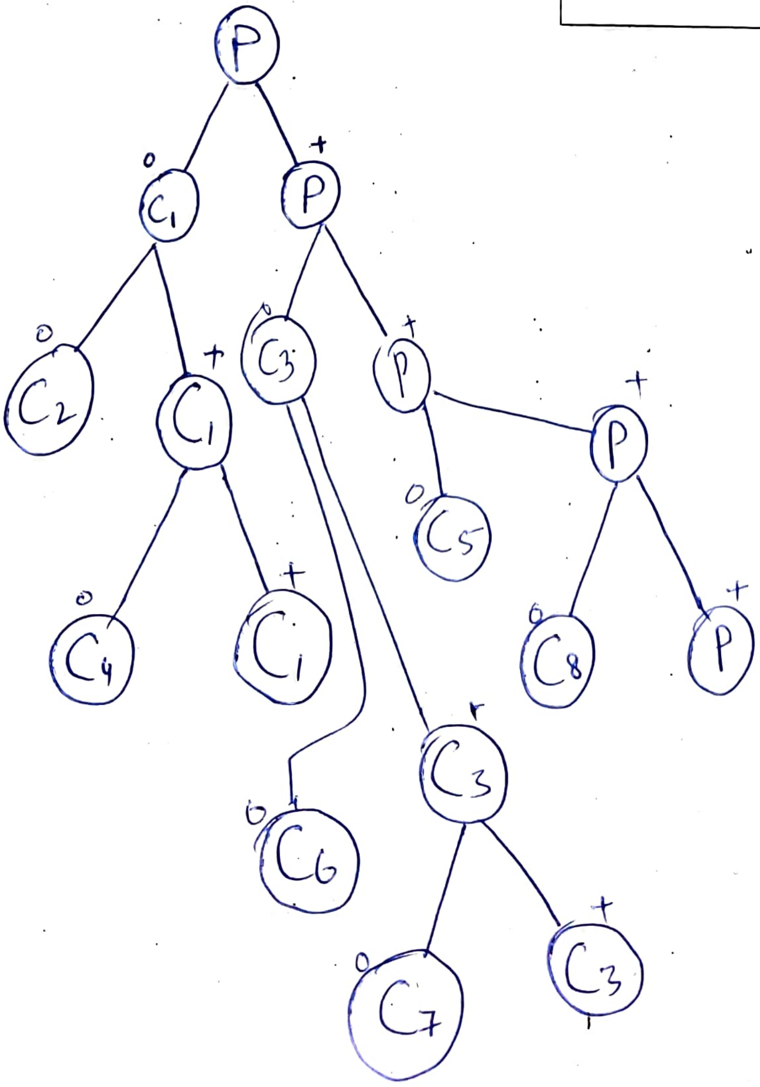
```
#include <stdio.h>

int i:

int main ()
{
for (i=0 ; i < 2 ; i++)
     {
     printf("i: %d \n", i);
     if (fork()) /* call #1 */
          fork(); /* call #2 */
     }
}
```

Output:
Process tree:

i : 0
i : 1
i : 1
i : 1



P
0
+
C1
P
0
C2
+
C1
0
C3
+
P
+
P
0
C5
0
C4
+
C1
0
C8
+
P
r
C3
0
C6
0
C7
+
C3