4039

M.Sc./II Sem. - 2015
INFORMATICS - Paper IT-23
Operating Systems

Time: 3 hours *Maximum Marks: 75*

(Write your Roll No. on the top immediately on receipt of this question paper)
(Answer Question Number 1 and any 4 from the rest)

Q1. a) Considering the following states of a process, draw a state transition diagram. Also clearly mention the *events* on which state transitions take place.

Process States – *new, ready, running, waiting, terminated.*

b) You need to copy one source file to a destination file. Write the steps needed to do this using only system calls (***open, read, write, close***).

c)Consider the following code fragment,

```
#include<stdio.h>
#include<unistd.h>
int main()
{
        fork();
        fork();
        fork();
        printf("Hello\n");
        return 0;
}
```

How many times the string ""Hello\n" will be printed and why?

d) Mention the steps to service a "page fault" in a demand paged Virtual Memory sytem with a suitable diagram.

e) Suppose there is an *external command* "mycommand" available in a standard "Linux" like operating system distribution. What do you mean by external command? What may be the other type of command? Support your answer with an example.

[3 x 5]

Q2. a) Consider the following code fragment and then answer the questions.

```
#include <stdlib.h>
#include <stdio.h>
#include <unistd.h>
char *c ;
int main()
{
        int ret = -1;
        ret = fork();
```

```
if( ret == 0)
{
    printf("Child : \n");
    c = malloc(sizeof(char));
    *c = 'A';
    printf("The value in variable c is %d\n", *c );
    exit(0);
}
else
{
    printf("Parent :\n");
    printf("The value in variable c is %c\n", *c + 1);
    free(c);
    exit(0);
}
}
```

i) What is the purpose of the "fork()" system call? What are the return values of this system call? Mention the return value if this call fails, for some reason. **[3]**

ii) What do you think will be the output of the program? Is there any chance of run-time memory fault, if this source code is compiled and executed? Explain briefly. **[4]**

iii) The library function "malloc" allocates memory dynamically. What will happen to the allocated memory when a process terminates? (Suppose library function "free" is not used explicitly to free memory.) **[2]**

b) Consider a set of processes {P1, P2, P3, P4}. The arrival time and the CPU time needed for the processes are given below.

Process	Arrival Time	CPU Time
P1	0	8
P2	1	4
P3	2	9
P4	3	5

Calculate the average waiting time for the following cases,

i) Using FCFS scheduling

ii) Using Non Preemptive SJF scheduling

iii) Using Preemptive SJF scheduling **[6]**

Q3. The file listing command "ls -li" is being executed in a "Linux" Shell and the output is shown below. Study the output and then answer the questions.

```
[localhost msc_iic]$ ls -li
total 4
```

656030 -rw-rw-r--. 2 user user 0 Apr 18 23:11 1.txt

656030 -rw-rw-r--. 2 user user 0 Apr 18 23:11 2.txt

655871 drwxrwxr-x. 2 user user 4096 Apr 18 23:10 dir

---

i) What is the purpose of the "-li" after "ls"? **[2]**

ii)What is the significance of the first column of the output? For the first two rows the value of the first column is the same. Explain this observation. **[2 + 3]**

iii)What is the significance of the 3rd column of the output? Briefly mention how this will be different when you create a "regular file" and when you create a "directory". **[2 + 3]**

b) Write a shell script, that will count the "***number of directories only***" in your current working directory. **[3]**

Q4. a) Briefly mention the purpose of a "make file". Write a makefile from the description given below.

"The name of your final executable should be ***myoutput***. It depends on four object files ***main.o***, ***f1.o***, ***f2.o***, ***f3.o***. These object files depend of source files named as ***main.c***, ***f1.c***, ***f2.c***, ***f3.c***. There is a single header file "***myheader.h***" which is referred by all these source files. The compiler that you need to use is "***gcc***". Also you need to specify a target which can remove all the object files and the final executable." **[5]**

b) Consider the following code fragment (ignore the absense of any header files), which tries to read data from a text file named "example.txt". Assume that the file is created using an editor, having five characters written "abcde". The code compiles successfully. Answer the questions that follow. **[10]**

```
#define MAXBUF 10
int main()
{
  int fd;
  int ret;
  char buffer[MAXBUF];
  fd = open("example.txt", O_RDONLY);
  printf("%d\n", fd);
  ret = read(fd, buffer, 5);
  printf("%d\n", ret);
  printf("%s", buffer);
  exit(0);
}
```

i) What do you mean by "file descriptor" ? How it different from a "file pointer" ?

ii) What will be value printed for "fd" and why?

iii) What will be the value printed for "ret" and why?

iv) It is seen that while printing the "buffer", the output is not proper. Why? How can

you correct this?

v) The file is not closed in the program. Will this result in any problem?

Q5. a)Consider the following code fragment (ignore the absense of any header files), which tries to invoke the standard word count program (wc) with the help of the "execlp" system call. Answer the question that follows.

```
int main()
{
    char *beforeexec = "Before Calling Exec";
    char *afterexec = "After Exec";
    write(1, beforeexec, strlen(beforeexec));
    execlp("wc","wc","call_wc.c",NULL);
    write(1, afterexec, strlen(afterexec));
    return 0;
}
```

i) Briefly state the working of the "execlp" system call. What is the meaning of "l" and "p" in "execlp"? Why is "wc" written twice in the arguments of execlp? What is the return value of "execlp" call? **[3 + 2 + 2 + 2]**

b) Consider a semaphore variable **S.** Define the operations **Wait(S)** and **Signal(S).**

Suppose two processess P1 and P2 are two be scheduled in such a way that statement S2 in Process P2 will be executed only after statement S1 of Process P1. Implement this requirement with the help of a semaphore. **[4 + 2]**

Q6. a) In the context of IPC (Inter Process Communication) objects, what is the difference between

"***Process Persistence***" and "***Kernel Persistence***"?

Out of the given IPC objects classify them as "Process Persistent" or "Kernel Persistent" as appropriate.

IPC Objects – *PIPE, FIFO, SHARED MEMORY, MESSAGE QUEUE, SEMAPHORE.* **[3 + 2]**

b) Consider two arbitrary commands, cmd1 and cmd2. You have entered ***cmd1|cmd2*** on the command prompt given by the shell. Based on this answer the following questions,

i) What is the significance of the "|" character to the shell? **[2]**

ii) Achieve the same effect with the help of redirection operators (< and >). **[2]**

iii) Show an example of this replacing cmd1, and cmd2 with some ***actual commands*** (Assume Linux Operating System) **[2]**

c) Consider the standard command "**wc**" to count the number of charaters, words and lines in a file.

Suppose in the commnad prompt you write "***wc myfile.txt***" and "***wc < myfile.txt***" where myfile.txt is an arbitrary text file. Mention briefly if there will be any difference in output between these two cases and why? **[2]**

d) Briefly mention the similarities and differences bwteen a "PIPE" and a "NAMED PIPE". **[2]**