National University of Computer and Emerging Sciences, Lahore Campus

	Course:	Operating System	Course Code:	CS-220
	Program:	BS(Computer Science)	Semester:	Fall 2020
	Duration:	1.5 hour	Total Marks:	50
	Paper Date:	17th October, 2020	Weight:	15%
	Section:	ALL	Page(s):	6
	Exam:	Mid-1		

Instructions/Notes: Answer questions on the question paper. Attempt all questions. Programmable calculators are not allowed. Write answers clearly and precisely, if the answers are not easily readable then it will result in deduction of marks. Use extra sheet for rough work, **cutting and blotting on this sheet will result in deduction of marks**.

Name:_____**Roll No.:**_____**Section:**_____

Question 1 (4 points): Memory (RAM) is a resource that an OS needs to manage. Allocating certain areas of RAM to a process is one part of managing memory. List two other OS functions for managing RAM.

(1)

(2)

(1) Taking memory back when a process ends.

(2) Protecting one processs RAM from another process.

Question 2 (6 points): How does a process know where the next instruction of the program is? Where is that stored?

Program counter points to the next instruction. Stored in a instruction register.

Question 3 (6 points): Which of the following resources would or would not be shared by multiple threads running in the same process? Write "yes" or "no" at the front, in the empty space. "Yes" means that the resource will be shared.

- Program counter
- Stack memory
- Global variables
- Static data
- Open files

- _____
- _____
- _____
- _____
- _____

- Program counter
- Stack memory
- Global variables
- Static data
- Open files

- No
- No
- Yes
- Yes
- Yes

Question 4 (12 points): Suppose that you have a program that copies the contents of one source file to a new destination file. The table below shows the steps needed to do this. You have to fill the column which tells what system call will be used for each operation. The first line is already filled for you. Name of the system is not necessary, you can just name the right category. If no system call is used then write NA.

No	Call Sequence	System call interface
1	Acquire input file name (prompt to screen)	I/O operations - `read()`
1	Accept input	
2	Acquire output file name (prompt to screen)	
3	Accept input	
4	Open the input file	
5	If file does not exist, abort	
6	Create output file	
7	If file exists, abort	
8	Loop: Read from input file	
9	Loop: Write to output file	
10	Until read fails	
11	Close output file	
12	Write completion message to screen	
13	Terminate normally	

No	Call Sequence	System call interface
1	Acquire input file name (prompt to screen)	I/O operations `write()`
1	Accept input	I/O operations `read()`
2	Acquire output file name (prompt to screen)	I/O operations `write()`
3	Accept input	I/O operations `read()`
4	Open the input file	File System
5	If file does not exist, abort	Error Detection
6	Create output file	File system
7	If file exists, abort	Error Detection
8	Loop: Read from input file	File system
9	Loop: Write to output file	File system
10	Until read fails	NA
11	Close output file	File system
12	Write completion message to screen	I/O operations `write()`
13	Terminate normally	Program execution

Question 5 (10 points): Based on the table given below use the non-preemptive priority scheduling algorithm and draw a Gantt chart, also calculate the Average Waiting Time.

Process	Burst Time	Priority
P_1	8	5
P_2	6	1
P_3	1	2
P_4	9	3
P_5	3	4

P_2	P_3	P_4	P_5	P_1
0	6	7	16	19

Average Waiting Time = 0+6+7+16+19 = 48/5 = 9.6

Question 6 (12 points): You need to write a program to search an array using concurrent processes. The program searches the array for a given number, and prints the index of the number in the array. If the number is not found then the program shall output -1. You are required to use two child processes to speed up the search. The first child shall search the first half of the array, while the second shall examine the other half. You need not to use any fancy algorithm for the search; rather you can use the simple linear search. Assume no duplicates are there in the array.
Following is the code skeleton:

```
int main() {
        const int N = ...;
        int a[N] = ...;
        int key, i;
        cout << "Enter a number: ";
        cin >> key;

        // Your code to search the array comes here. Write the complete code of the main
            function on the last page.

        cout << "The number found at index: " << i;
        return 0;
}
```

```
int main()
{
        const int N = 10;
        int a[N] = {1,2,3,4,5,6,7,8,9,10};
        int key, i;

        cout << "Enter a number: ";
        cin >> key;

        int pd[2];
        //pipe(pd);
        pid_t pid;
        pid = fork();

                 if(pid < 0)
                 {
                          cout  <<  Error \n  ;
                          return 0
                  }

        if(pid == 0)
        {
                //child 1
                for(int l = 0 ; l < N/2 ; l++)
                {
                        if(key == a[l])
                        {
                                return l;
                        }
                }
                return -1;
        }

                 //parent

        pid1 = fork();
        if(pid1 < 0)
                  {
                            cout <<  Error \n  ;
                            return 0;
                    }
                    if( pid1 == 0)
        {
                //child 2
                for(int l = N/2 ; l < N ; l++)
                {
                        if(key == a[l])
                        {
                                return l;

                        }
                }
                return -1;

        }

    //parent
        int x , y;
    wait(&x)

        if(x == -1)
        {
              wait(&y);
          i = y;
        }
        else
        {
                i  = x
        }

        cout << "The number found at index: " << i << endl;
        return 0;
}
```