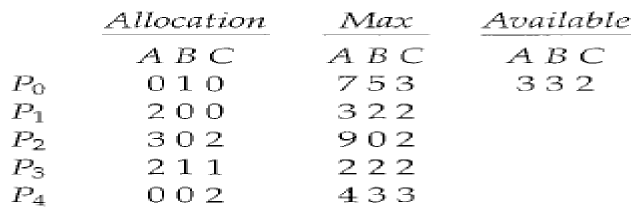
LAB-4(19-25 April 2021)

Deadlocks

1. Consider a system with: five processes, P0 → P4, three resource types, A, B, and C. Type A has 10 instances, B has 5 instances, C has 7 instances. At time T0 the following snapshot of the system is taken.



	Allocation	*Max*	*Available*
	A B C	*A B C*	*A B C*
P_0	0 1 0	7 5 3	3 3 2
P_1	2 0 0	3 2 2	
P_2	3 0 2	9 0 2	
P_3	2 1 1	2 2 2	
P_4	0 0 2	4 3 3	

   Write a Linux C program to check weather system is in safe state or not. Also demonstrate the unsafe sequence by modifying any resource allocation.

2. Write a Linux C program to demonstrate that deadlock and starvation are opposite problems. Solution to deadlock problem causes more starvation and solution to the starvation problem can cause deadlock.

File Management

1. Write a program to open 2 files (the names of the files should be taken as command line arguments) one in read only mode and the other in write only mode and then read all the characters from the first file and write them into the second file. Also check whether the opened file exists and if not then catch the exception by writing the code for the error also. What is it doing, which system utility does it resemble?

2. Write a program to open 2 files (the names of the files should be taken as command line arguments) one in read only mode and the other in write only mode and then read the characters from the first file and write them into the second file. However do not write all the characters in the second file; write the first one and then every *n*th character ('*n*' can be any positive number here) from the the first file. Also check whether the opened file exists and if not then catch the exception by writing the code for the error also.

3. Write a program to open 2 files (the names of the files should be taken as command line arguments) one in read only mode and the other in write only mode and then read the contents of the first file and write them into the second file, word by word. However the words should be written in the reverse order, the first word of the first file should be the last word of second file and the last word of the first file should be the first word of the second file.

4. (a)Write a program to open a file in RDWR mode and read the characters from the file and append them in the same file.
   (b) For the above program in 2(a), read a string from the user and append the string into the opened file.

5. (a)Write a program to create 2 files using open system call and then write the string “open system call” in one file and “write system call” in the second file.
   (b) open 2 files, take the input from the user for the data and then using lseek() write the user data at the specified location at the beginning , at the end or at the location specified by the offset.