
SCHOOL OF COMPUTER ENGINEERING

KIIT DEEMED TO BE UNIVERSITY

CS-29002

Operating Systems Laboratory

Lecture:	0	Internal Assessment Marks:	60
Tutorial:	0	End Term Marks:	40
Practical:	2 Hrs/Week	Credits:	1
Faculty: Dr.NamitaPanda Section: CSE#5			

Software Requirements

- GCC compiler
- Unix platform (Currently used Ubuntu12.04 LTS)

Hardware Requirements

- Processor: 386 and above
- RAM: 32 MB and above

Course objectives

The objective of this lab is to give an idea about different components of operating system and their interactions happened in an operating system. Specifically, in this course, we mostly focus on the UNIX operating system, where student can get the practical understanding about how to manage the processes with respect to process creation, process communication process synchronization, threading, manage file, etc.

List of Sample experiments

Lab 1:

- UNIX commands
- How to write and execute a shell program
- Expression evaluation and command line arguments
- Assignments
 1. Verify different UNIX commands (syntax to use with options)
Some list (but not limited): basename, mkdir, ls, echo, read, expr, kill, who, which, date, users, vim, touch, bash, cat, hostname, rm, mv, chmod, top and so on.

2. Write a shell program to add two numbers using command line argument.



3. Write a shell program to convert distance into meter, cm and km.

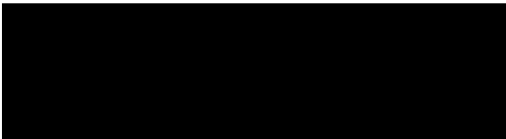


4. Write a shell script to extract the data from the date command, and display the result in different format.(if the date command gives “Thu Jan 2 14:21:54 IST 2014” then display the result in “2/Jan/2014/14.21”)

Vary the date command with different options.

Lab 2:

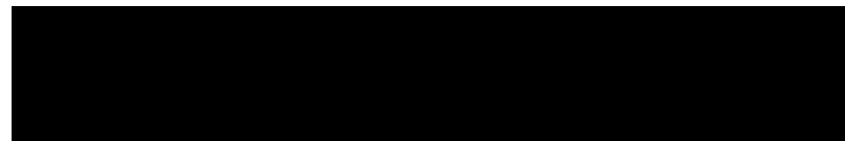
- logical operator, relational operator used in bash shell
- How to write down the conditional statements in bash shell using logical and relational operator?
- Different control statements (if, elif, case, while, for)
- Assignments
 - Write a shell program to find whether a given year is a leap year or not.



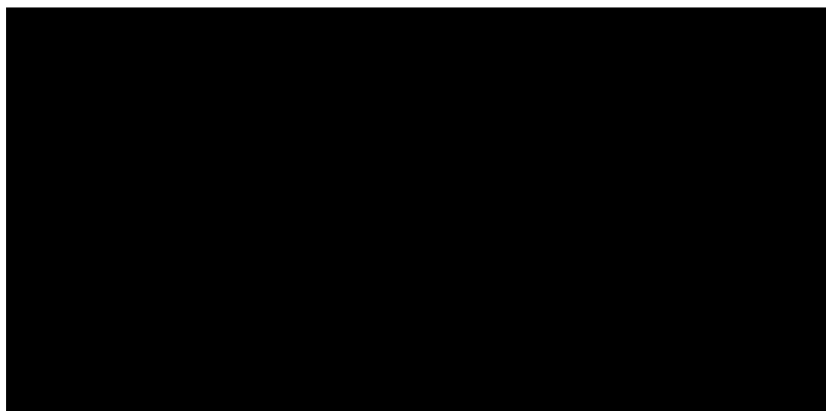
- Write a shell program to find greatest among three numbers.



- Write a shell program to display the prime number between 1 and hundred.



- Execute the “ps -ef” and “ps -ux” command. Write a shell program that takes the output of these two commands and display the detail about a process whose parent id is 2. The detail about a process will be shown as it is shown during the execution of “ps -ux” command.



- Write a shell script to extract the data from the date command, and display the result in numerical form with 12 hour format. (if the date command gives “Thu Jan 2 14:21:54 IST 2014” then display the result in “02/01/2014/2:21:54 PM”)

```
kiiit@KIIT-6185:~$ ./t.sh
Sat Jan  6 10:50:38 IST 2024
06/01/2024/10:50:38 AM
```

- Let a directory present in the home directory called “XYZ” that consists of few files and directories. Write a shell script to move all the files present in the “XYZ” directory to a subdirectory called “MyFile” and all the subdirectories present in the “XYZ” directory to a subdirectory called “MyDir”.

Lab 3:

- Different Loop statements (while, for, until, case)
- Assignments
 - Write a shell script to reverse a given integer.

```
#!/bin/bash
# Script to reverse a given integer
# Example: Input: 12345, Output: 54321

# Read an integer from the user
read num

# Reverse the integer
rev=""
while [ $num -gt 0 ]
do
    digit=$((num % 10))
    rev=$((rev * 10 + digit))
    num=$((num / 10))
done

# Print the reversed integer
echo $rev
```

- Write a shell script to verify whether the given string is a palindrome or not.

```
#!/bin/bash
# Script to verify whether the given string is a palindrome or not

# Read a string from the user
read str

# Reverse the string
rev=""
for (( i=0; i<${#str}; i++ ))
do
    rev="${str:$i:1}$rev"
done

# Compare the original string with the reversed string
if [ $str == $rev ]
then
    echo "The string is a palindrome."
else
    echo "The string is not a palindrome."
fi
```

- Write a shell script to find the following

1. Home directory
2. Bash version
3. Host name
4. current directory
5. exit

```
#!/bin/bash
# Script to find the following information

# 1. Home directory
echo "Home directory: $HOME"

# 2. Bash version
echo "Bash version: $BASH_VERSION"

# 3. Host name
echo "Host name: $(hostname)"

# 4. current directory
echo "current directory: $(pwd)"

# 5. exit
exit 0
```

- Write a shell program which takes maximum 8 integer type arguments through command line and do the following operation:
 - If the first argument/last result (a) is divisible by second argument (b) then new result=a/b
 - Else If (a%b != 0) and b is divisible by 5 then new result=a*b

- Else if ($a > b$) then new result= $a-b$
- Else new result= $a+b$

Lab 4:

- Implementation of array in shell script.
- Assignments
 - Write a shell script that will take 3 file names as command line argument and concatenate first two files line by line and store the result in third file.

o Write a shell script that will store all the files in different directory named as

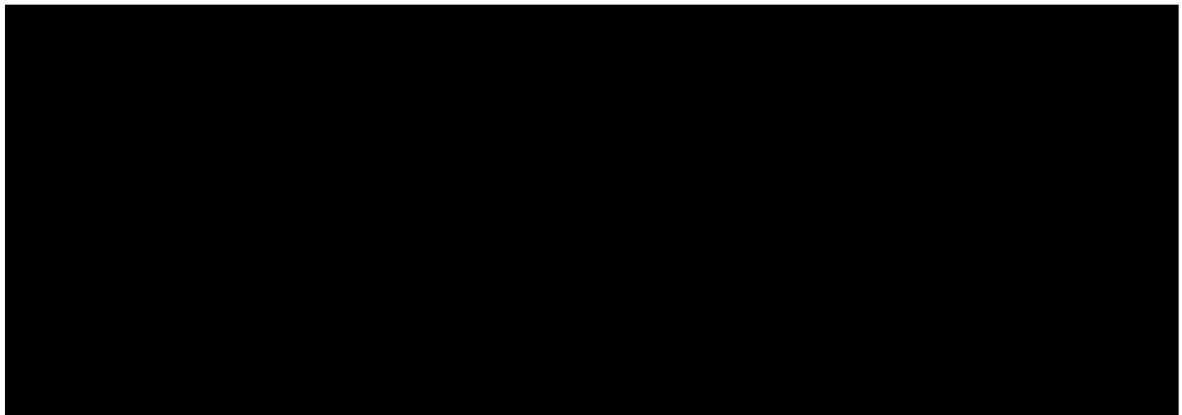
```

kilit@kilit-VirtualBox:~/Desktop$ ./concat_files.sh file1.txt file2.txt result.txt
Content of file1.txt
OS lab3
Content of file2.txt
OS lab4
Content of result.txt
Concatenated file1.txt and file2.txt line by line and stored in result.txt.
Now content of result.txt is
OS lab3
OS lab4
kilit@kilit-VirtualBox:~/Desktop$
  
```

x_1, x_2, \dots where maximum size of each directory is X.

Lab 5:

- Discuss a generalized way to design the CPU scheduling algorithm
- Assignments
 - FCFS algorithm implementation in C using structure / linked list



- **Home assignments**
 - SJF(non-preemptive) algorithm implementation in C

Lab 6:

- Discuss a generalized way to implement preemptive CPU scheduling algorithms
- Assignments
 - Preemptive SJF algorithm implementation in C using structure / linked list

```

Enter the number of processes: 4
Enter arrival time and burst time for process 1: 1 4
Enter arrival time and burst time for process 2: 2 7
Enter arrival time and burst time for process 3: 5 10
Enter arrival time and burst time for process 4: 6 12

Process Arrival Time    Burst Time    Waiting Time    Turnaround Time
1          1              4              28              32
2          2              7              24              31
3          5             10              18              28
4          6             12              15              27
Average Waiting Time: 21.25
Average Turnaround Time: 29.50

```

- **Home assignments**

- Round robin algorithm implementation in C

Lab 7:

- Discuss basics of round robin algorithm implementation
- How to create a processes using fork system call
- Use of wait function
- Assignments
 - Create different process tree using fork system call

```

ayan@ayan-vm:~/Desktop/exam$ ./exp.out
Parent of all: 7585
Child with id: 7586 and its Parent id: 7585
Child with id: 7587 and its Parent id: 7585
Child with id: 7588 and its Parent id: 7586
Child with id: 7590 and its Parent id: 7586
Child with id: 7589 and its Parent id: 7587
Child with id: 7591 and its Parent id: 7586
Child with id: 7592 and its Parent id: 7590
Child with id: 7593 and its Parent id: 7592

```

- Write a program where a process waits for all its child processes

Note: “child has terminated”--- this sentence does not print before “1.Hello from child”because of wait.

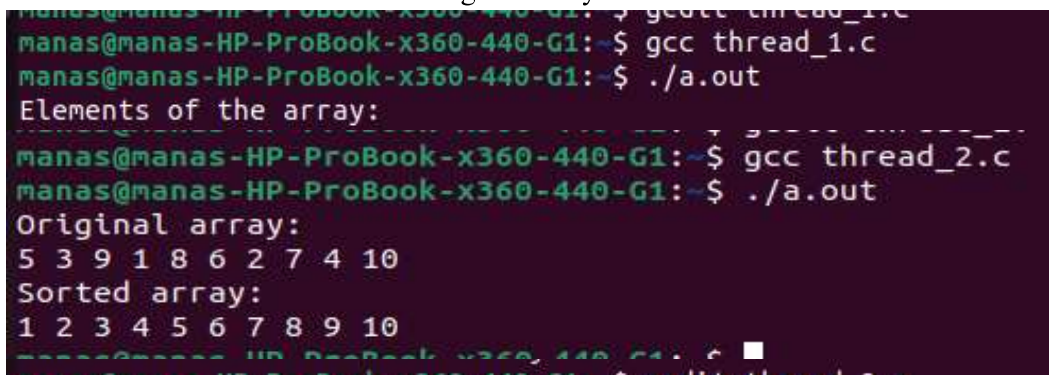
Lab 8:

- Discuss about process communication using pipe
 - how to create pipe between two processes
 - What is the use dup function? How it can be utilized in pipe line commands

- Assignments
 - Write a program that creates n level process structure where each level consists of exactly one process. Here every parent process makes some modification to the original message and sends it to its child and this process repeats till it reached at the last child in the process tree.
- **Home assignment**
 - Write a program to create a ring topology among n number of process and message has to be passed using the pipe in clock wise or anti clock wise direction.

Lab 9:

- Discuss the thread creation using pthread library
- Assignments
 - Write a program to create two threads where one thread adds half of the element from the beginning of the array and another thread add half of the element from the end of the array. And at the end main process shows the sum of all the elements in the given array.



```

manas@manas-HP-ProBook-x360-440-G1:~$ gcc thread_1.c
manas@manas-HP-ProBook-x360-440-G1:~$ ./a.out
Elements of the array:
5 3 9 1 8 6 2 7 4 10
manas@manas-HP-ProBook-x360-440-G1:~$ gcc thread_2.c
manas@manas-HP-ProBook-x360-440-G1:~$ ./a.out
Original array:
5 3 9 1 8 6 2 7 4 10
Sorted array:
1 2 3 4 5 6 7 8 9 10

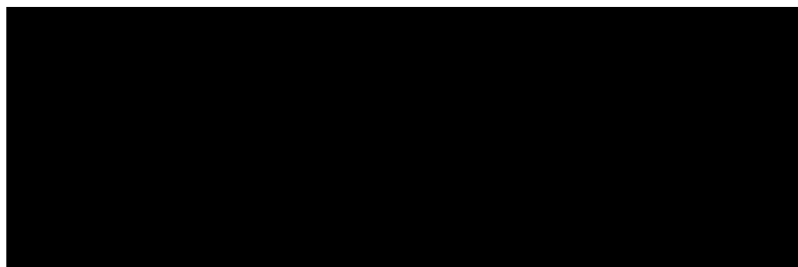
```

Write a program to implement the merge sort using multiple threads

◦

Lab 10:

- Discuss different functions related to Mutex lock using POSIX library
- Discuss different functions related to conditional variable using POSIX library
- Discuss different functions related to semaphore using POSIX library
- Assignment
 - Write a program for reader-writer problem using lock.



- Write a program to implement critical section using semaphore

Home Assignments:

- Producer consumer problem using semaphore
- Producer consumer problem using mutex lock
- Dining philosophers problem using semaphore
- Dining philosophers problem using mutex lock

Guidelines

1. *Students should be regular and come prepared for each laboratory class.*
2. *Students should bring their lab record and practice note books to every class.*
3. *The prescribed text and reference books and class notes can be kept ready for reference if required.*
4. *Students have to complete their lab experiments in the lab and be capable to explain and show the modifications, output results as and when required by the Faculty/Lab Programmers/Teaching Assistants responsible for that lab. All these are to ensure that the students' capability is built up to understand, debug and modify codes as per the requirements.*
5. *In case a student misses a class, it is his/her responsibility to complete the missed experiment(s).*
6. *The code written by the student should meet the following:*
 - i) *Program should have proper input prompt messages(**Menu Driven**) and descriptive output.*
 - ii) *Input validation should be done (data type, range error etc.) and give appropriate error messages and suggest corrective actions.*
 - iii) *Comment lines should be used to give Students Name, Roll No, Branch, Section, Subject, Problem Statement, describe functions, Expected Input & Output and all the key Logic.*
 - iv) *Program should be indented properly. Variables and functions should be meaningfully named. [USE CODING STANDARDS].*
 - v) *Try to optimize the code wherever possible.*

7. *All programs should be verified by different values and lengthy inputs. [with minimum VALIDATIONS]*
8. *Once the experiment(s) get executed, they should show the program and results to the Faculty/Lab Programmers/Teaching Assistants responsible for that lab. and copy the same in their observation book and get signed which ensures him/her with day to day performance marks*
9. *Student should submit his/her record by/in the next lab session to ensure the lab record marks gets evaluated. Failing to do so reduces the lab record marks.*
10. *Lab examination questions need not necessarily be limited to the questions in the laboratory manual, rather it could include some variations and/or combinations of questions.*
11. *Students are strictly advised to take care of their personal belonging all the time. The University is not responsible and liable to any lost of personal belonging in the computer labs.*

Scheme of Evaluation

Maximum marks for Database Lab is **100**, which is divided into Continuous Internal Assessment and **End-Sem** Final Evaluations.

Internal Assessment (Continuous evaluation over the semester):	60Marks
End-Tem Evaluation (At the End of the Semester):	40 Marks
Minimum Marks for passing in the Lab:	50 Marks