Operating Systems

Lab – 02

Objectives:

1. Practice using **vim** editor
2. Process management

Resources:

1. Video Lecture 05: https://youtu.be/7tFniseSLzM?list=PL7B2bn3G_wfBuJ_WtHADcXC44piWLRzr8
2. Video Lecture 07: https://youtu.be/2bYGoOTXrUg?si=_SFSA_d9qh62bkCu

Note: Make a directory named Lab-02 on your Desktop and perform following tasks in it.

Task 01: What is a text editor? Mention names of at least **five** text editors that you can use on Linux. What are the three different modes of **vim**, describe them briefly?

Task 02: Copy the **/etc/passwd** to **practice.txt** in Lab-02 and open **practice.txt** in vim and also make a copy of this file as **practice2.txt**.

a. Practice cursor movement commands:
   - Move the cursor to the beginning and the end of the **line** with single character commands
   - Move the cursor to the beginning and the end of the **file** with single character commands
   - Move the cursor to **line 20** without counting the lines.

b. Practice editing commands:
   - Delete the first line of the file (**Hint:** there is a delete-line command)
   - Delete line 7 and add words "**AND DYNAMICS**" right after line 15, on the same line.

c. Practice search and replace commands:
   - Go to the beginning of the file and find the next occurrence of the word "**usr**". Replace it with "**USER**".
   - Now find every occurrence of "**usr**" and replace it with "**USER**".

d. Save the file and **quit** vim, which returns you to the shell prompt.

e. Give a command to open **practice.txt** in vim so that
   - It opens at last line
   - It opens at line number 21
   - It opens at line containing string "**USER**"

f. Open **practice.txt** and **practice2.txt** in vim at once. (**Hint:** –o and -O)
   - Move your cursor in between two files
   - Copy some text from one file and paste in the other
   - Finally, save and quit both files

Task 03: Run **vim** to create a new file named **hello.c** and type a **hello world** program in it. Enter in last line mode, and execute a shell command that will compile the program and create all the temporary files including the final executable with the name of **hello** instead of **a.out** (**Hint**: !cmd). Finally, in the last line mode give a command to create a new shell and execute the executable **hello**, see its output and come back to vim again (**Hint**: sh) **vim** without mentioning any file name.

Task 04:

Write a C program to print "Learning Linux is Fun with Arif Butt. " on stdout

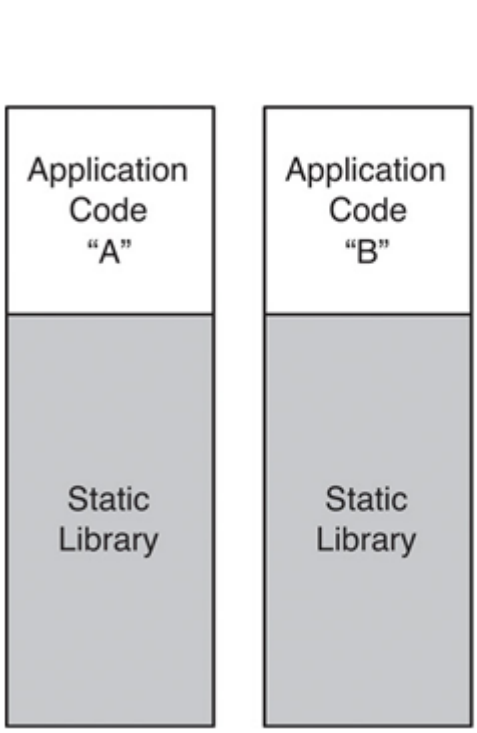
1. Create a **preprocessed** code file of your source C program.
2. Create an **object** file of your assembly code.
3. Create an **executable** file of your object file.

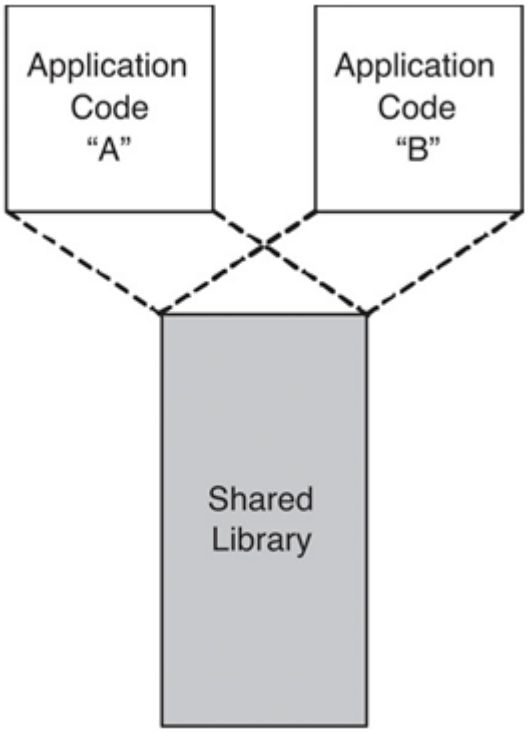
Note:

- *You must have a clear understanding of each file generated during the compilation process.*
- *Mention the file formats created during the task.*

Task 05:

Compile and link the given Program **Statically** and **Dynamically**. Give each binary a different name.






```c
#include<stdio.h>
#include<unistd.h>
#include<stdlib.h>
#include<fcntl.h>
int main(int argc, char* argv[]) {
      write(1, "Hi, I am your own cat ;)\n", 26);
       int fd = 0;
      if(argc > 1)
      {
              fd = open(argv[1], O_RDONLY);
      }
      int rv = 0; char buff[1024];
      while(rv = read(fd, buff, 1024))
      {
              write(1, buff, rv);
      }
      return 0;
}
```

Note:

You can link statically using --**static** flag with gcc.

The above program mimics the general behavior of "**cat**". Compare both files w.r.t sizes and "**strace**" both of the statically linked and dynamically linked "**cat**"**.** What difference do you see?

*Tried strace? Did you get a lot of stuff ⁉️. But don't worry, just look at **openat** syscall from sample output given below. Try to understand, why libc is being opened by this binary and not by static binary.*

```
→ tmp λ strace ./dynamic
execve("./dynamic", ["./dynamic"], 0x7ffd664f71c0 /* 55 vars */) = 0
brk(NULL)                               = 0x55dceddc3000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffef8d2b0e0) = -1 EINVAL (Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fc42c3c5000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", {st_mode=S_IFREG|0644, st_size=52967, ...}, AT_EMPTY_PATH) = 0
mmap(NULL, 52967, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fc42c3b8000
close(3)                                = 0
openat(AT_FDCWD, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\0\0\0\0"..., 832) = 832
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
pread64(3, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\0\0\0\0"..., 48, 848) = 48
pread64(3, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\f\221\2039x\324\224\323\236S"..., 68, 896) = 68
newfstatat(3, "", {st_mode=S_IFREG|0755, st_size=2220400, ...}, AT_EMPTY_PATH) = 0
pread64(3, "\6\0\0\0\4\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0@\0\0\0\0\0\0\0"..., 784, 64) = 784
mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3, 0) = 0x7fc42c000000
mprotect(0x7fc42c028000, 2023424, PROT_NONE) = 0
```

strace(ing) a dynamically linked binary

Task 06:

1. Display the "**disassembly**" of the executable in **intel** format.
2. Display "**Section Headers"** of the executable and note down the **count** of section headers.
3. Display the "**Program Headers"**
4. Display the "**ELF header"**.

Task 07: Practice running programs in the background, and switching programs from foreground to background and vice-versa. Practice using commands like **ps,top,vmstat, free,uptime,watch**