Labs – I

1.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>

#include <sys/types.h>
#include <fcntl.h>
#include <stdlib.h>

int main()
{
        int fd, PAGE_SIZE, ret;
        char  *buf;
        struct stat  *sbuf

        PAGE_SIZE = getpagesize();
        buf = (char *) malloc(PAGE_SIZE + 10);
        if(buf == NULL) {
                printf(“No memory \n”);
                exit(1);
        }
        fd = open(“./test.txt”, O_RDONLY);
        if( fd < 0) {
                perror(“open:”);
                exit(1);
        }
        /********************************
         Enabling page read
         *******************************/
        ret = read(fd, buf, PAGE_SIZE);
        if( ret == -1) {
                perror(“read:);
                exit(1);
        }
        fprintf(stderr,”%s\n”, buf);
        fflush(stderr);
        /*******************************************
         Repositioning the file descriptor to the start of the file
         ******************************************/
        ret = lseek(fd, 0, SEEK_SET);
        if(ret == -1) {
                perror(“lseek:”);
                exit(1);
        }
        /*******************************************
         Read till end of the of file
         ******************************************/
```

```
     ret = fstat(fd, sbuf);
    if( ret == -1) {
        perror(“fstat : “);
        exit(0);
    }

    ret = read(fd, buf, sbuf->st_size);
    if(ret == -1) {
        perror(“read:”);
        exit(1);
    }
    close(fd);
    return 0;
}
```

2.

```
#include <stdio.h>
#include <sys/stat.h>
#include <fcntl.h>

int main()
{
    int fd;

    fd = open(“./demo.txt”, O_CREAT | O_NONBLOCK | O_RDWR);
    if( fd < 0) {
        perror(“open:”);
        exit(1);
    }
    close(fd);
    return 0;
}
```

3.

```
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
#include <fcntl.h>
#include <sys/sendfile.h>
#include <stdlib.h>

int main()
{
    int fdin, fdout, ret;
    struct stat *buf;
```

```
            /****************************************
             Open one existint file in read mode
             **************************************/
            fdin = open(“./src.txt”, O_RDONLY);
            if( fdin < 0) {
                        perror(“open fdin:”);
                        exit(1);
            }
            /*******************************************
             Create or open the destination file
             ******************************************/
            fdout = open(“./dst.txt”, O_WRONLY | O_CREAT);
            if( fdout < 0) {
                        perror(“open fdout:”);
                        exit(1);
            ret = fstat(fd,buf);
            if(ret == -1) {
                        perror(“fstat:”);
                        exit(1);
            }
            ret = sendfile(fdout, fdin, 0, buf->st_size);
            if(ret  == -1) {
                        perror(“sendfile :”);
                        exit(0);
            }
            close(fd);
            return 0;
            }
```

4.

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/stat.h>
#include <fcntl.h>

int main()
{
            int fd_in, fd_out, ret;

            /************************************************************
             Open the existing named pipe. Before this write one application
             that creates named pipe using mkfifo and keep on write data on
             to it.
             ************************************************************/
            fd_in = open(“./src.tx”, O_RDONLY);
            if(fd_in == -1) {
                    perror(“open fd_in:”);
                    exit(1);
            }

            fd_out = open(“./dst.txt”, O_WRONLY | O_CREAT);
            if(fd_out == -1) {
                    perror(“open fd_out:”);
                    exit(1);
            }
            ret = splice(fd_in, 0, fd_out, 0, 10);
            if( ret == -1) {
                    perror(“splice:”);
                    exit(1);
            }
            close(fd);
            return 0;
}
```

5.

```
#include <stdio.h>

#include <sys/types.h>

#include <dirent.h>

#include <stdlib.h>



int main()

{

    DIR *dir;

   struct dirent *list;



   dir = opendir("../rough");

   if(dir == NULL) {

        perror("opendir:");

        exit(1);
    }

   while(1) {

        list = readdir(dir);

        if(list == NULL)

        break;

        fprintf(stderr,"%s\t", list->d_name);

        fflush(stderr);

   }

   printf("\n");

   return 0;

}
```

8.

```
#include <stdio.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <sys/fcntl.h>
#include <stdlib.h>

int main()
{
      int fd, ret;
     char buf[30];

       fd = open("./pr.txt", O_RDWR|O_CREAT);
       if(fd == -1) {
           perror("open:");
            exit(1);
     }
     ret = pwrite(fd,"hello world", 12,0);
```

```
            if(ret == -1) {
                  perror("pwrite:");
                   exit(1);
            }
            ret = pread(fd, buf, 12, 0);
            if( ret == -1) {
                 perror("pread:");
                 exit(1);
            }
         fprintf(stderr,"%s\n", buf);
          fflush(stderr);
        return 0;
}
```