Department of Computer Science Forman Christian College (A Chartered University) Lahore

BY LOVE SERVE ONE ANOTHER

Estd. 1864

SYSTEM PROGRAMMING
COMP 440

LAB: 05

Name:

Roll Number:

Date:

My First System Program

Learning Objectives

In this lab, students will learn

- Reading and Writing Files in C
- Command line arguments
- Terminals in *NIX and Control Sequence Introducer
- Device Special Files in *NIX
- Determining actual size of the terminal window
- Performing reverse video and curser handling using CSI

Equipment Required:

In this lab we will use lab computers equipped with any flavor of UNIX/LINUX.

Tasks

1. To understand the functionality of more command.
2. To write a program using system/library calls that can simulate the more command.

Expected Deliverables

Students are required to show their work to the lab staff/resource person.

Lab Task 1

You are given the following program that simulates a bare bone version of more command. Type the program in gedit, execute it and try to understand the sequence of instructions given in the program.

```
//more1.c

#include <stdio.h>

#define SCREEN_ROWS 23
#define LINELEN 512
#define SPACEBAR 1
#define RETURN 2
#define QUIT 3
#define INVALID 4

void do_more_of(FILE * fp);
int get_user_input();

int main(int argc,char *argv[])
{
    FILE *fp;
    int i=0;
    if(argc==1)
     do_more_of(stdin);
    else
     while(++i<argc)
     {
          fp=fopen(argv[i],"r");
          if(NULL!=fp)
          {
               do_more_of(fp);
               fclose(fp);
          }
          else
               printf("Skipping %s \n",argv[i]);
     }
    return 0;
}

//=================================================================
==//

void do_more_of(FILE *fp)
{
    char line[LINELEN];
    int num_of_lines=SCREEN_ROWS;
    int getmore=1;
    int reply;
    while(getmore && fgets(line,LINELEN,fp))
    {
     if(num_of_lines==0)
     {
          reply=get_user_input();
          switch(reply)
          {
               case SPACEBAR:
                    num_of_lines=SCREEN_ROWS;
                    break;
               case RETURN:
                    num_of_lines++;
                    break;
               case QUIT:
                    getmore=0;
                    break;
               default:
                    break;
          }
     }
     if(fputs(line,stdout)==EOF)
          exit(1);
     num_of_lines--;
    }
}

//============================================================
=//

int get_user_input()
{
    int c;
    printf("\033[7m more? \033[m");
    while((c=getchar())!=EOF)
     switch(c)
     {
          case 'q':
               return QUIT;
          case ' ':
               return SPACEBAR;
          case '\n':
               return RETURN;
          default:
               return INVALID;
     }
}
```

Lab Task 2

Observe the working of above program and compare it with the actual version of more. Try using following commands:

```
$ ls /bin | ./more1
```

You can also create a file with numbers ranging from 1 to 30, one on each line. Use following assuming num.txt is the name of your file

```
$ cat num.txt | ./more1
```

Observe the behavior of your version of more and compare it with that of original more. Using /dev/tty concept discussed in class, modify the code of more1.c to rectify the observed problem.