ASSIGNMENT NO. 5

AIM :- Write X86 ALP to find, a) Number of Blank spaces b) Number of lines c) Occurrence of a particular character. Accept the data from the text file. The text file has to be accessed during Program_1 execution and write FAR PROCEDURES in Program_2 for the rest of the processing. Use of PUBLIC and EXTERN directives is mandatory.

APPARATUS :

- Core 2 duo/i3/i5/i7 - 64bit processor
- OS – ubuntu 32bit/64bit OS
- Assembler used –nasm (the netwide assembler)
- Editor Used – gedit

THEORY :

NEAR & FAR Procedures

The 80x86 supports near and far subroutines. Near calls and returns transfer control between procedures in the same code segment. Far calls and returns pass control between different segments. The two calling and return mechanisms push and pop different return addresses.

Call instruction uses the following syntax:

```
call    ProcName
```

and the ret instruction

```
ret
```

Algorithm for program_1

1. Start
2. Initialize all the sections needed in programming
3. Display “Enter file name” message using Print macro expansion
4. Accept file name using Accept macro and store in filename buffer
5. Display “Enter character to search” message with the expansion of Print macro

6. Read character using Accept macro expansion
7. Open file using fopen macro
8. Compare RAX with -1H if equal then display error message “Error in Opening File” with Print macro expansion else go to step ix
9. Read content of opened file in buffer
10. Store file length in abuf_len
11. Call far_procedure
12. Stop

Macros:

Macro 1

1. Name : Print

2. Purpose: to display the messages by replacing the whole code by simple macro declaration

3. I/P: sys_write call Number i.e rax=1, File descriptor (for Standard output rdi=1), Buffer Address in rsi, and length of Buffer in rdx. Then Call syscall.

Macro 2

1. Name : Accept

2. Purpose: to accept input from the user by replacing the whole code by simple macro declaration

4. I/P: sys_read call Number i.e rax=0, File descriptor (for Standard input rdi=0), Buffer Address in rsi, and length of Buffer in rdx. Then Call syscall.

Macro 3

1. Name : fopen

2. Purpose: to open a file in given mode

3. I/P: sys_write call Number i.e rax=2, File name in rdi, Mode of file in rsi (R=0,W=1,RW=2), and file permission in rdx. Then Call syscall.

Macro 4

1. Name : fread

2. Purpose: to read the content of file

3. I/P: sys_read call Number i.e rax=0, File descriptor in rdi , Buffer Address in rsi, and length of Buffer in rdx. Then Call syscall.

Macro 5

1. Name : fclose
2. Purpose: to close opened file
3. I/P: sys_read call Number i.e rax=3, File handler in rdi. Then Call syscall.

Algorithm for Far Procedure

1. Name : far_procedure
2. Purpose: to count 1. Number of Blank spaces 2. Number of lines 3. Occurrence of a particular character.
3. I/P : Content stored in buffer
4. Algorithm for Procedures
    i. Start
    ii. Load effective address of buffer in RSI
    iii. Load content of abuf_len in RCX
    iv. Load content of char in BL
    v. Move value of RSI in AL
    vi. Compare AL with 20H (ASCII value of space) if not equal then go to step vii else increment content of scount
    vii. Compare AL with 10H (ASCII value of line) if not equal then go to step viii else increment content of ncount
    viii. Compare AL with BL if not equal then go to step ix else increment content of ccount
    ix. Increment RSI
    x. Repeat from step vi if RCX is not equal to zero
    xi. Display “Number of space” message with the expansion of Print macro.
    xii. Move content of scount in RBX
    xiii. Call display8num procedure
    xiv. Display “Number of lines” message with the expansion of Print macro.

xv. Move content of ncount in RBX

xvi. Call display8num procedure

xvii. Display “Number of Occurrence of Character” message with the expansion of Print macro.

xviii. Move content of ccount in RBX

xix. Call display8num procedure

xx. Ret

xxi. Stop

CONCLUSION:

_____

_____

_____

_____