**Assignment No.8**

|  |
| --- |
| **Title of Assignment:**  Write X 86 menus driven Assembly Language Program (ALP) to implement OS (DOS) commands TYPE, COPY and DELETE using file operations. User is supposed to provide command line arguments in all cases. |
| **Relevant Theory:**  **Command Line Arguments**  It is possible to pass arguments to programs when they are executed.  There are 2 parameters:  1. *argc* refers to the number of arguments passed.  2. *argv[]* is a pointer array which points to each argument which is passed to program.  *\*argv[0]* is the name of the program invoked, which means that *\*argv[1]* is a pointer to the first  argument supplied, and *\*argv[n]* is the last argument. If no arguments are supplied, *argc* will be  one. Thus for n arguments, *argc* will be equal to n + 1. The program is called by the command  line,  ./prog\_name.exe para1 para2 ….. …… paran  **System Call for Unlinking a file:**  **Call with :**  Rax : 87 function number for unlinking a file  Rdi : file name  **Design Analysis/ Implementation Logic:**  **Algorithm**   1. Pop the stack & store the value in ECX register. 2. Move this value in ECX to argc variable.. 3. Compare argc with 3 if not equal print error and goto step 4. Pop the stack to ignore file name. 5. Pop the stack again to get base address of first command line parameter i.e. file name 1. 6. Read it from the base address character by character till null character in varaiable fname1 7. Append fname1 with 0(Null) 8. Pop the stack again to get base address of second command line parameter i.e. file name 2. 9. Read it from the base address character by character till null character in varaiable fname2 10. Append fname2 with 0(Null) 11. Print Menu 12. Get choice from user 13. If choice is 31H goto step 18 14. If choice is 32H goto step 26 15. If choice id 33H goto step 35 16. If choice is 34H goto step 36 17. Else goto step 11 18. Open file1 19. Compare rax with 0 if less then print error and goto step 36 20. Copy rax in fd\_s (file descriptor for file1) 21. Read 80 character from file in buffer 22. Compare rax with 0 if 0(EOF) then goto step 25 23. Write the buffer on console 24. Goto step 21 25. Goto step 11 26. Open file1 in read mode (flag=0) 27. Compare rax with 0 if less then print error and goto step 28. Open file2 in write mode with O\_crete flag (101) 29. Compare rax with 0 if less then print error and goto step 30. Read 80 character from file1 in buffer 31. Compare rax with 0 if 0(EOF) then goto step 33 32. Write the buffer in file2 33. Goto step 29 34. Goto step 11 35. Unlink the file goto step 11 36. End |
| **Testing:**  **Test Conditions:**  **Input:**  Command line Input : two file names  **Output:**  If choice =1 ,file displayed on console  If choice =2,file2 as a copy of file 1  If choice =3,file 1 deleted. |
| **FAQs:**   1. Explain System call for file opening, file closing, reading from file and writing to file, unlinking a file. 2. Explain file opening modes. 3. Explain command line parameters and how to retrieve them. |
| **Conclusion:**  Successfully implemented the menu driven ALP for type, copy and delete command using command line parameters. |