Title : Creating a Text Editor in DOS mode

Course Title: CSE103 Lab [Structured Programming]

Section: 08 [R 10:10AM-1:10PM]

Instructor: DHMAI

hasan.mahmood@ewubd.edu

Submitted By -

Abrar Khatib Lajim	- [2022-3-60-043]
Sumaiya Binte Naser	- [2022-2-60-155]
Md. Asif Hossain	- [2022-3-60-007]
Fariha Akter	- [2022-2-60-082]

Submitted to –

Dr. Hasan Mahmood Aminul Islam

Assistant Professor

Department of Computer Science & Engineering

Source Code:

https://drive.google.com/file/d/1wTxiBXPD6PS0CS2_6vamJJ05s4guAvy3/view?usp=sharing

Output:

https://drive.google.com/file/d/1glRz35aL-qBPhHuzVuFKga_wAQYTx1FT/view?usp=sharing

‘ Video ‘

```c
#include <stdio.h>       //include standard input/output library
#include <stdlib.h>     //include standard library functions
#include <conio.h>     //include console input/output library
#include <ctype.h>    //include library for character classification functions
#include <stdint.h>  //include library for integer data types

#define MAX_TEXT_LENGTH 1000  //define a macro constant with value 1000
char file_name[100];         //declare a character array to store file name
int x,y;
char ch;                         //declare a character variable
uint8_t choice, a, position;//declare unsigned integer variables
char text[MAX_TEXT_LENGTH];/* declare a character array of size 1000 */

int main()        //main function
{
    FILE *fp;   //declare file pointer

    printf("=> File Name To Create: ");/* display message to enter file name*/
gets(file_name);                  //get file name input from user
fp = fopen(file_name, "w");     //open the file with write mode
if (fp == NULL)   //check if file is successfully created
    {
     printf("- Error creating file.\n");//display error message
     exit(1);   //exit program
    }
printf("File Created Successfully!\n\n");  /* display message that file is created successfully */
```

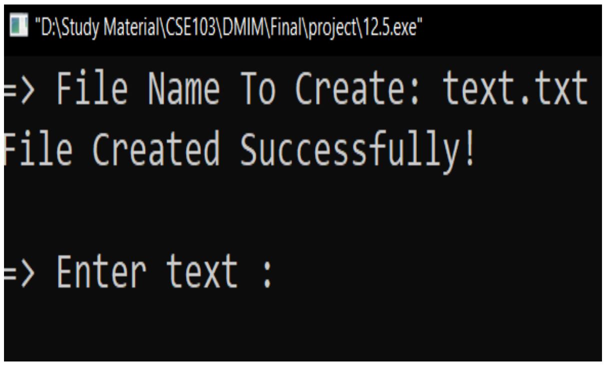
```c
    printf("=> Enter text :");  //display message to enter text
    uint8_t a = 0; //initialize unsigned integer variable 'a' to zero
        do    //execute the following block of code at least once
        {
            ch = getch(); //get character input from user
            if (isprint(ch))//check if character is printable
            {
                text[a] = ch;  //store character in text array
                a++;          //increment value of 'a'
                printf("%c", ch);//display the character on console
            }
            else if (ch == 13)/*check if character is Enter key '\n' or '\r'*/
            {
                text[a] = '\0'; /* set the last character in text array to null character */
                break; //break out of the loop
            }
            else if (ch == 8 && a > 0)   /* check if character is Backspace key and 'a' is greater than zero */
            {
                a--;   //decrement value of 'a'
                printf("\b \b");   /*delete the previous character from console*/
            }
        } while (1);   //execute the loop infinitely
        fputs(text, fp);   //write the text to the file
        fclose(fp);   //close the file
        printf("\n*File saved successfully!\n");/*display message that file is saved successfully */
        do   //execute the following block of code at least once
        {
            printf("-------   Menu   -------\n");
            printf("\n\t1. Read \n");//option to read the file
            printf("\t2. Delete \n");//option to Delete the file
            printf("\t3. Insert \n");//option to modify the file
            printf("\t4. Exit \n");//option to exit the program
            printf("=>Enter your choice: ");/*display message to enter choice*/
            scanf("%d", &choice); //get integer input from user
            switch (choice)   //check the value of choice
            {
                case 1:     //if choice is 1
                fp = fopen(file_name, "r");//open file in read mode
                if (fp == NULL)//check if file is successfully opened
                    {
                      printf("Error opening file.\n");//display error
                    exit(1);
                }
                printf("\n- Text in %s:[ ", file_name);
                while ((ch = fgetc(fp)) != EOF)
                {
                    printf("%c", ch);
                }
                printf(" ]\n");
                fclose(fp);       //close the file
                break;
            case 3:
                for(int b=0;text[b]!='\0';b++)
                {
                    if(text[b]==' ')
                    {
                     printf("\t Position[%d]=' '\t\n",b); //space
                     b++;
                    }
                    else if(text[b]!=' ');
                    {
                      printf("\t Position[%d]=%c\t\n",b,text[b]);
                    }
                }
                printf("\nEnter position to insert text: ");
                scanf("%d",&position);

                printf("Enter text: ");
                a = 0;
                do
                {
                    ch = getch();

                    if (isprint(ch))/*checks whether a character is a printable character or not*/
                    {
                        text[a] = ch;
                        a++;
                        printf("%c", ch);
                    }
                    else if (ch == 13) // Enter key
                    {
                        text[a] = '\0';
                        break;
                    }
                    else if (ch == 8 && a > 0) // Backspace key
                    {
                        a--;
```

```c
                         printf("\b \b"); /*\b= It is used to move the cursor backward.*/
                        }
                    } while (1);
                    fp = fopen(file_name, "r");// to read the file
                    if (fp == NULL)
                    {
                        printf("Error opening file.\n");
                        exit(1);
                    }
                    a = 0;
                    while ((ch = fgetc(fp)) != EOF)//EOF= End of file
                    {
                        a++;
                    }
                    fclose(fp);
                    fp = fopen(file_name, "r+");/*r+ = opens a file in both read and write mode*/
                    if (fp == NULL)
                    {
                        printf("Error opening file.\n");
                        exit(1);
                    }
                    fseek(fp, position, SEEK_SET);/*fseek() is used to move file pointer associated with a given file to a specific position*/
                    fputs(text, fp);
                    fclose(fp);
                    printf("\nText inserted successfully!\n");
                    break;
                case 2:
                    {
                        for(int b=0;text[b]!='\0';b++)
                    {
                        if(text[b]==' ')
                        {
                            printf("\t Index[%d]=' '\t\n",b); //space
                             b++;
                        }
                        else if(text[b]!=' ');
                        {
                            printf("\t Index[%d]=%c \t\n",b,text[b]);
                        }
                    }
                    printf(">Starting Index: ");
                    scanf("%d",&x);
                    printf(">Ending   Index: ");
                    scanf("%d",&y);
                    printf("-Before: %s\n", text);
                    deleteChars(text, x,y);      //function to delete
                    printf("-After Delete: %s\n", text);
                    fp=fopen(file_name,"w");
                     {
                     fputs(text,fp);
                     fputs("\n",fp);
                     fclose(fp);
                     }
                     break;
                    }
                    case 4:
                        {
                            printf("\nExiting...\n");
                            exit(0);
                        }
                        default:
                            {
                                printf("\nInvalid choice.\n");
                            }
            }
        } while (1);
        return 0;
    }
    void deleteChars(char* text, int index, int count)
    {
        int i, j;
        int len = strlen(text);/* Check if the index is within the string's bounds*/
        if (index >= 0 && index < len)
            {
            int deleteCount = (count < (len - index)) ? count : (len - index);    // Calculate the number of characters to delete
         for (i = index, j = index + deleteCount; j <= len; i++, j++)
                {
                text[i] = text[j];//Shift remaining characters toLeft
                }
            }
    }
```

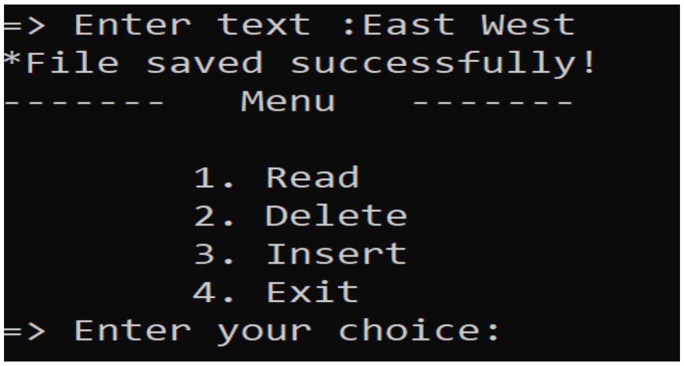
1.



```
"D:\Study Material\CSE103\DMIM\Final\project\12.5.exe"
=> File Name To Create: text.txt
File Created Successfully!

=> Enter text :
```

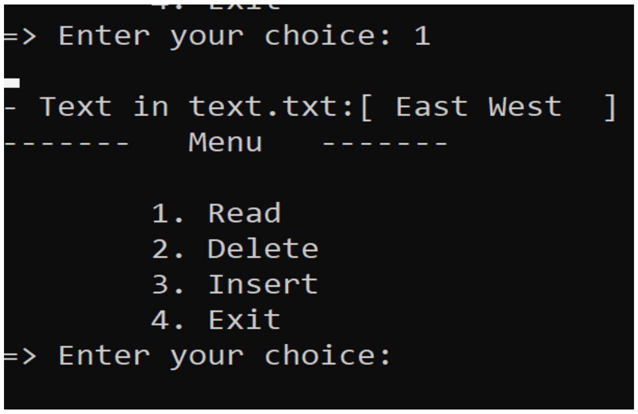
2.



```
=> Enter text :East West
*File saved successfully!
--------    Menu    --------

        1. Read
        2. Delete
        3. Insert
        4. Exit
=> Enter your choice:
```

3.

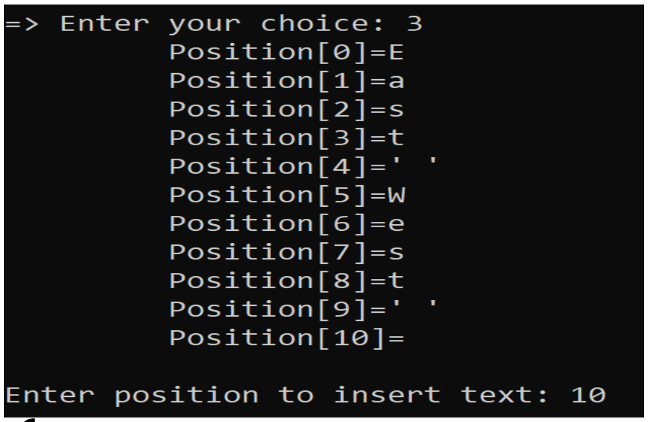


```
=> Enter your choice: 1

- Text in text.txt:[ East West  ]
-------    Menu    -------

        1. Read
        2. Delete
        3. Insert
        4. Exit
=> Enter your choice:
```

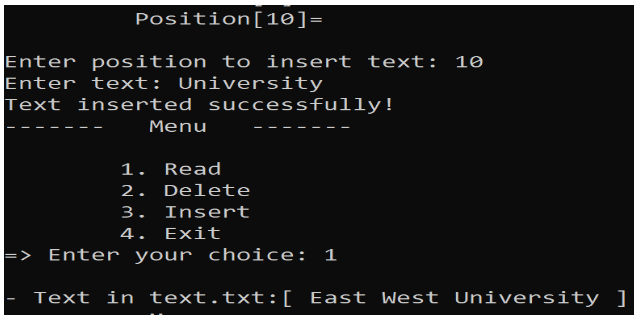
4.



```
=> Enter your choice: 3
        Position[0]=E
        Position[1]=a
        Position[2]=s
        Position[3]=t
        Position[4]=' '
        Position[5]=W
        Position[6]=e
        Position[7]=s
        Position[8]=t
        Position[9]=' '
        Position[10]=

Enter position to insert text: 10
```

5.



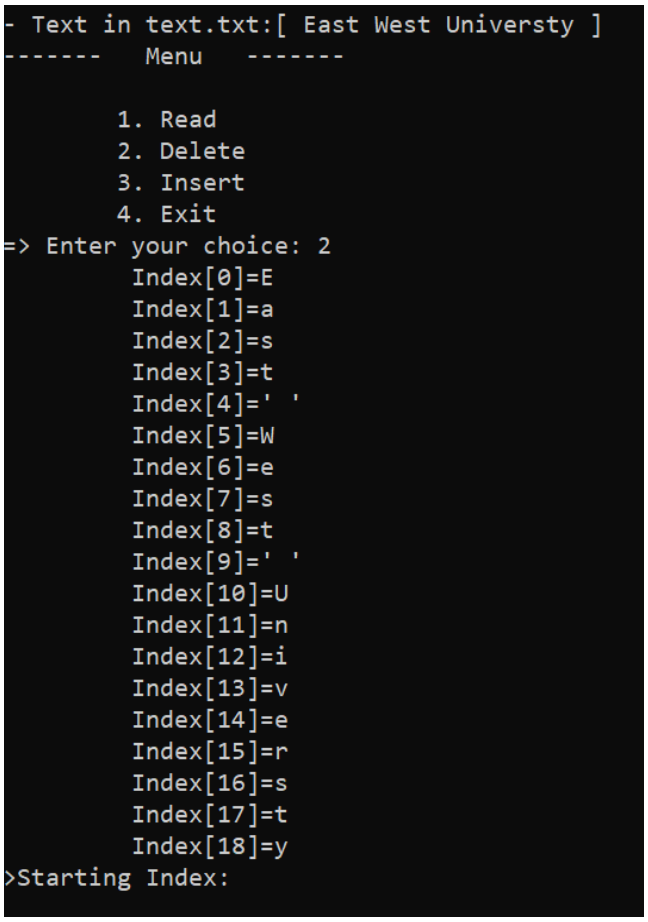
```
        Position[10]=

Enter position to insert text: 10
Enter text: University
Text inserted successfully!
-------    Menu    -------

        1. Read
        2. Delete
        3. Insert
        4. Exit
=> Enter your choice: 1

- Text in text.txt:[ East West University ]
```

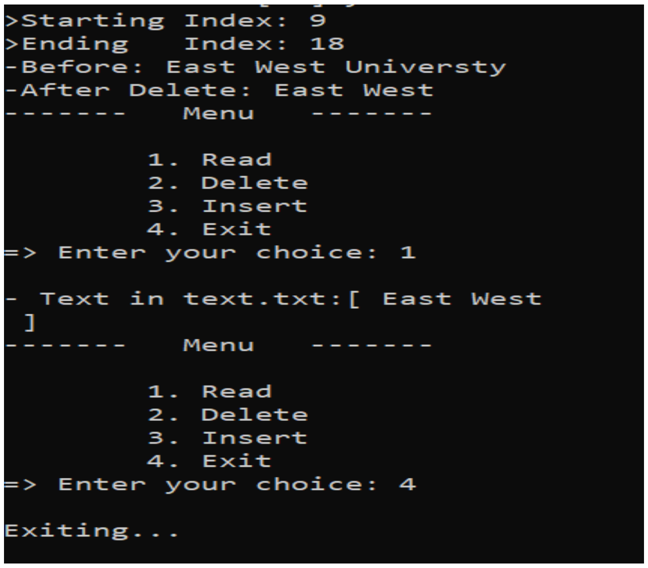
6.



```
- Text in text.txt:[ East West Universty ]
-------    Menu    -------

        1. Read
        2. Delete
        3. Insert
        4. Exit
=> Enter your choice: 2
        Index[0]=E
        Index[1]=a
        Index[2]=s
        Index[3]=t
        Index[4]=' '
        Index[5]=W
        Index[6]=e
        Index[7]=s
        Index[8]=t
        Index[9]=' '
        Index[10]=U
        Index[11]=n
        Index[12]=i
        Index[13]=v
        Index[14]=e
        Index[15]=r
        Index[16]=s
        Index[17]=t
        Index[18]=y
>Starting Index:
```

7.



```
>Starting Index: 9
>Ending   Index: 18
-Before: East West Universty
-After Delete: East West
-------    Menu    -------

        1. Read
        2. Delete
        3. Insert
        4. Exit
=> Enter your choice: 1

- Text in text.txt:[ East West
 ]
-------    Menu    -------

        1. Read
        2. Delete
        3. Insert
        4. Exit
=> Enter your choice: 4

Exiting...
```

Explanation :

1. The necessary header files are included: **stdio.h** for standard input/output functions, **stdlib.h** for standard library functions, **conio.h** for console input/output functions, **ctype.h** for character classification functions, and **stdint.h** for integer data types.
2. A macro constant MAX_TEXT_LENGTH is defined with a value of 1000.
3. Global variables are declared:
    a. file_name: a character array to store the file name.
    b. x and y: integer variables.
    c. ch: a character variable. choice, a, and position: unsigned integer variables. text: a character array of size MAX_TEXT_LENGTH to store the text.

The main() function starts:

- A file pointer fp is declared.
- The user is prompted to enter the file name using printf().
- The gets() function is used to read the file name input from the user.
- The fopen() function is called to open the file in write mode ("w"). If the file cannot be opened, an error message is displayed, and the program exits.
- If the file is opened successfully, a success message is displayed.
- The user is prompted to enter text using printf().
- A do-while loop is used to read characters from the user using getch(). Inside the loop:
    - If the character is printable (`isprint()`), it is stored in the `text` array, and the character is displayed on the console.
    - If the Enter key is pressed (`ch == 13`), the loop breaks.
    - If the Backspace key is pressed (`ch == 8`) and there are characters in the `text` array, the last character is removed from both the array and the console.
- After the loop, the text array is written to the file using fputs().

- The file is closed using fclose().
- A success message is displayed.
- A do-while loop is used to present a menu of options to the user. The loop continues indefinitely until the user chooses to exit (choice == 4). Inside the loop:
    - The user is prompted to enter their choice.
    - Based on the choice, different cases are executed using switch-case.

Case 1: Read the file:

- The file is opened in read mode ("r").
- If the file cannot be opened, an error message is displayed, and the program exits.
- The contents of the file are read character by character using fgetc(), and they are displayed on the console.
- The file is closed.

Case 2: Delete text:

- A loop is used to iterate through the text array and display the indexes of characters. If a space character is encountered, it is displayed as "Index[index]=' '". For other characters, it is displayed as "Index[index]=character".
- The user is prompted to enter the starting and ending indexes of the text they want to delete.
- The deleteChars() function is called, passing the text array, starting index, and the number of characters to delete.
- The deleteChars() function performs the deletion by shifting the remaining characters to the left.
  The conditional (ternary) operator in C. The conditional operator has the following syntax: Condition ? value_if_true : value_if_false
  In the case of the line int deleteCount = (count < (len - index)) ? count : (len - index);, it can be understood as follows:
    1. count < (len - index) is the condition being evaluated.
    2. If the condition is true, the value of count is assigned to deleteCount.

    3. If the condition is false, the value of (len - index) is assigned to deleteCount.

  Essentially, the line is checking if count is less than the remaining characters in the text (len - index). If it is, deleteCount will be assigned the value of count, meaning that count characters will be deleted. If count is greater than or equal to the remaining characters, deleteCount will be assigned the value of (len - index), indicating that all the remaining characters will be deleted.
  The purpose of this line is to determine the actual number of characters to delete, ensuring that it does not exceed the remaining characters in the text.

- The updated text array is displayed.
- The file is reopened in write mode ("w").
- The updated text array is written to the file using fputs().
- The file is closed.

  The deleteChars() function is defined separately outside the main() function and takes the text array, starting index, and count as parameters. It performs the deletion of characters by shifting the remaining characters to the left based on the given index and count."Position[index]=character".

Case 3: Insert text:

1. A loop is used to iterate through the text array and display the positions of characters. If a space character is encountered, it is displayed as "Position[index]=' '". For other characters, it is displayed as Case 3: Insert text:
2. A loop is used to iterate through the text array and display the positions of characters. If a space character is encountered, it is displayed as "Position[index]=' '". For other characters, it is displayed as "Position[index]=character".
3. The user is prompted to enter the position at which they want to insert the text.
4. The user is prompted to enter the text they want to insert.
5. A do-while loop is used to read characters from the user using getch(). Inside the loop:

   a. If the character is printable (`isprint()`), it is stored in the `text` array, and the character is displayed on the console.
   b. If the Enter key is pressed (`ch == 13`), the loop breaks.
   c. If the Backspace key is pressed (`ch == 8`) and there are characters in the `text` array, the last character is removed from both the array and the console.
6. The file is reopened in read mode ("r") to determine the length of the file by counting the characters using fgetc() until the end of the file (EOF) is reached.
7. The file is closed.
8. The file is reopened in read and write mode ("r+").
9. The file pointer is moved to the specified position using fseek() with the position value as the offset and SEEK_SET as the starting point.
10. The text is inserted into the file at the specified position using fputs().
11. The file is closed.
12. A success message is displayed.

Case 4: Exit:

1. The program displays the message "Exiting..." to indicate that the program is about to terminate.
2. The exit(0) function is called with an argument of 0. This function terminates the program execution and returns the value 0 to the operating system, indicating successful termination.
3. Since the program execution is terminated by calling exit(0), the subsequent code in the do-while loop will not be executed.
4. If none of the previous case blocks match the user's choice, the program enters the default block.
5. In the default block, the program displays the message "Invalid choice." to inform the user that the entered choice is not valid.
6. After executing the default block, the program proceeds to the beginning of the do-while loop, where the user is prompted again for their choice.

The do-while loop continues to execute indefinitely (while(1)) until the user chooses to exit the program by selecting option 4

A few notes on the code:(On Going)

fseek(fp, pos, SEEK_SET) is used to move the file pointer to the position pos from the beginning of the file. This is done before inserting the text entered by the user into the file at the specified position.If the user chooses to read the file, the program opens the file in read mode using fopen() and reads the text using fgetc(), and displays it on the screen.

The gets() function used to read the filename is considered unsafe and has been deprecated. It is recommended to use fgets() instead, which allows you to specify a maximum length for the input.

The getch() function used to read input from the user is not part of the standard C library and may not be available on all platforms. It is recommended to use scanf() or fgets() instead.The program does not check if the specified position to insert the text is within the bounds of the file, which could result in undefined behavior or a crash. It would be a good idea to add some input validation for this.

some logical error with delete

Project[Text Editor]