Object Oriented Programming

Task #4

Report

Problem Statement:

Write an object oriented C++ program for Text Editing. The program should have the following features:

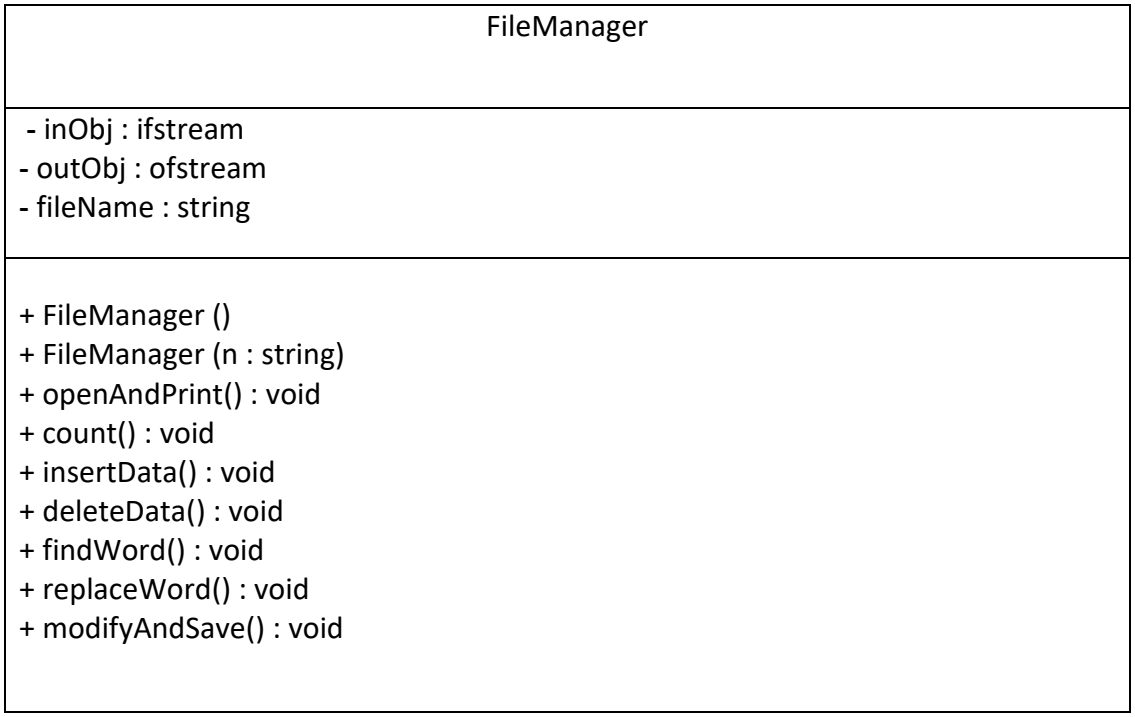
- To open a text file and show its contents on the screen.
- To count and print total number of characters, words and lines of the text in the file.
- To insert new text at a particular position in the text file.
- To delete text from a particular position in the text file.
- To search/find a word in the text file.
- To replace a word in the text file.
- To save the modified text file.

Write an interactive, command-driven program to test your Editor.

Objectives:

- To show how file handling can be achieved through OOP (classes and objects)
- To show how different operations in a problem statement can be converted into class member functions.
- To create a file editor that is completely independent of the file being modified.
- To create a file editor that allows user the complete freedom to choose which file to edit.
- To show how a string functions like **erase, replace and find** can be used to edit a text file.
- To create a file editor that also takes data security and memory consumption into consideration while working on files.
- To build a user-friendly program that allows the user to carry out whatever operations they want to.

UML diagram:


FileManager
- inObj : ifstream
- outObj : ofstream
- fileName : string
+ FileManager ()
+ FileManager (n : string)
+ openAndPrint() : void
+ count() : void
+ insertData() : void
+ deleteData() : void
+ findWord() : void
+ replaceWord() : void
+ modifyAndSave() : void

Source Code:

```
#include <iostream>
#include <fstream>
#include <string>
#include <stdlib.h>

using namespace std;

class FileManager
{
private:
	ifstream inObj;
	ofstream outObj;
	string fileName;

public:

	FileManager() : fileName("1.txt")
	{}
	FileManager(string n) : fileName(n)
	{}
	void setFileName(string n)
	{
		fileName = n;
	}

	void openAndPrint();
	void count();
	void insertData();
	void deleteData();
	void findWord();
	void replaceWord();
	void modifyAndSave();

};


int main()
{
	FileManager j;
	int option;
	string file;

	cout << "\n\t\t\t\t\t\tWelcome to Text File Manager!\n\n\n";
	cout << "\t\t\t\tPlease enter the name of the file you want to open: ";

	getline(cin, file);
	j.setFileName(file);

	cout << "\n\n\tFile contents: \n\n";
	j.openAndPrint();
	j.count();
```

```
	while (1)
	{
		cout << "\n\n\t\t\t\t\tEnter 1 to insert data into the file,\n\t\t\t\t\t   2 to
delete data from the file,\n";
		cout << "\t\t\t\t    3 to find a word in the file,\n\t\t\t\t\t   4 to replace a
word in the file,\n";
		cout << "\t\t\t\t\t       5 to modify the file,\n\t\t\t\t\t         or enter 0 to
exit\n\n\t\t\t\t\t\t\t  ";
		cin >> option;

		switch (option)
		{
		case(0):

			system("CLS");
			cout << "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\t\t\t\t\t\t
Goodbye!\n\n\n\n\n\n\n\n\n\n\n\n";
			exit(0);
		case(1):

			j.insertData();
			system("CLS");
			cout << "\n\n\tNew file contents: \n\n";
			j.openAndPrint();
			j.count();
			break;

		case(2):

			j.deleteData();
			system("CLS");
			cout << "\n\n\tNew file contents: \n\n";
			j.openAndPrint();
			j.count();
			break;

		case(3):

			system("CLS");
			j.findWord();
			break;

		case(4):

			system("CLS");
			j.openAndPrint();
			j.replaceWord();
			system("CLS");
			cout << "\n\n\tNew file contents: \n\n";
			j.openAndPrint();
			j.count();
			break;

		case(5):
			j.modifyAndSave();
			system("CLS");
			cout << "\n\n\tNew file contents: \n\n";
```

```
                        j.openAndPrint();
                        j.count();
                        break;

                }
        }
}

void FileManager::openAndPrint()
{
        string line;
        inObj.open(fileName, ios::in);
        if (inObj.good())
        {
                while (getline(inObj, line))
                {
                        cout << line << endl;
                }
                inObj.close();
        }
        else
        {
                cout << "\n   Sorry, the specified doesn't exist.";
        }

}

void FileManager::count()
{
        int characters = 0, words = 0, lines = 0;
        string tempFileData;

        inObj.open(fileName, ios::in);
        if (inObj.good())
        {
                while (getline(inObj, tempFileData))
                {
                        ++lines;
                        characters += tempFileData.length();
                        for (int i = 0; i < tempFileData.length(); ++i)
                        {
                                if (tempFileData[i] == ' ' || i == (tempFileData.length() - 1))
                                {
                                        ++words;
                                }
                        }
                }
        }
        inObj.close();
        cout << "\n\nTotal characters: " << characters;
        cout << "\nTotal words: " << words;
        cout << "\nTotal lines: " << lines << endl;

}

void FileManager::insertData()
{
        int linePos, charPos, lineCounter = 0;
```

```
	string originalLine, line, substring;

	inObj.open(fileName);
	if (inObj.good())
	{
		cout << "\n\n   Enter the line number in which you want to insert new data: ";
		cin >> linePos;
		for (int i = 0; i < linePos; ++i)
		{
			getline(inObj, line);
		}
		inObj.close();
		originalLine = line;
		cout << endl << " '" << originalLine << "'" << endl;
		cout << "\n   Enter the character position number from where you want to insert
new data: ";
		cin >> charPos;
		--charPos;
		cout << "\n   Enter the data you wish to insert: ";
		cin.ignore();
		getline(cin, substring);
		originalLine.insert(charPos, substring);

		inObj.open(fileName, ios::in);
		outObj.open("2.txt", ios::trunc);

		while (!inObj.eof())                                          //Making a copy of
file but with updated data. I could have also used strings to stora all lines of the file but
that would consume too much memory and hence would be really inefficient
		{
			++lineCounter;
			getline(inObj, line);
			if (lineCounter != linePos)
			{
				outObj << line << endl;
			}
			else
			{
				outObj << originalLine << endl;
			}
		}

		inObj.close();
		outObj.close();

		inObj.open("2.txt", ios::in);
		outObj.open(fileName, ios::trunc);                            //Erasing old data
from original file

		while (!inObj.eof())                                                  //Sending new
data back to original file from copy file because i dont want to delete and rename file in every
execution as that could create a lot of waste and the old file would also be accessible in
"Recycle bin" which is not a very good thing from data security perspective
		{
			getline(inObj, line);
			outObj << line << endl;
		}
```

```
		inObj.close();
		outObj.close();

		outObj.open("2.txt", ios::trunc);				//Erasing data from
copy file. Sound move from both data security and memory consumption perspectives.
		outObj.close();
;
	}
	else
	{
		cout << "\n   Sorry, the file could not be opened.";
	}
}

void FileManager::deleteData()
{
	int linePos, charPos, len, lineCounter = 0;
	string line, originalLine;

	inObj.open(fileName);
	if (inObj.good())
	{
		cout << "\n\n   Enter the line number from which you want to delete data: ";
		cin >> linePos;
		for (int i = 0; i < linePos; ++i)				//Picking the line
that needs data deletion
		{
			getline(inObj, line);
		}
		inObj.close();
		originalLine = line;
		cout << endl << " '" << originalLine << "'" << endl;
		cout << "\n   Enter the character position number from where you want to delete
data: ";
		cin >> charPos;
		--charPos;
		cin.ignore();
		cout << "\n   Enter the number of characters you want to delete: ";
		cin >> len;
		originalLine.erase(charPos, len);

		inObj.open(fileName);
		outObj.open("2.txt", ios::trunc);

		while (!inObj.eof())					//Making a
copy of file but with updated data
		{
			++lineCounter;
			getline(inObj, line);
			if (lineCounter != linePos)
			{
				outObj << line << endl;
			}
			else
			{
				outObj << originalLine << endl;
			}
		}
```

```
		inObj.close();
		outObj.close();

		inObj.open("2.txt", ios::in);
		outObj.open(fileName, ios::trunc);				//Erasing old data
from original file

		while (!inObj.eof())						//Moving
updated data back to original file
		{
			getline(inObj, line);
			outObj << line << endl;
		}

		inObj.close();
		outObj.close();

		outObj.open("2.txt", ios::trunc);				//Erasing data from
copy file
		outObj.close();

	}
	else
	{
		cout << "\n   Sorry, the file could not be opened.";
	}
}

void FileManager::findWord()
{
	string word, line, filler = "/";
	int linePos = 0, charPos = 0, wordLen = 0, check = 0;

	cout << "\n\n   Please enter the word you want to find: ";
	cin >> word;
	wordLen = word.length();

	inObj.open(fileName);
	if (inObj.good())
	{
		while (!inObj.eof())
		{
			getline(inObj, line);
			if (line.find(word) != -1)
			{
				check = 1;
				charPos = line.find(word);
				cout << "\n   Word found in line " << linePos + 1 << " at character
position " << charPos + 1;

				line.replace(charPos, wordLen, filler);

				while (line.find(word) != -1)
				//looking for multiple occurences
				{
					charPos = line.find(word);
```

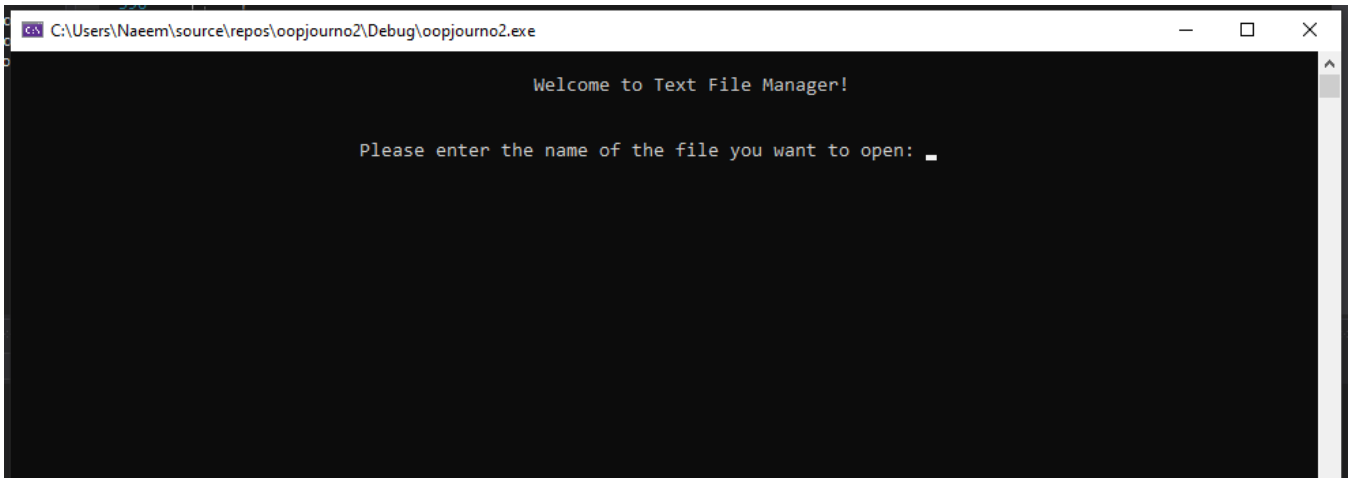
```
                                    cout << "\n   Word found in line " << linePos + 1 << " at
character position " << charPos + 1;
                                    line.replace(charPos, wordLen, filler);
                              }
                        }
                        ++linePos;
                }
                inObj.close();
                if (check == 0)
                {
                        cout << "\n\n   Sorry. The entered word does not exist in file.\n";
                }
        }
        else
        {
                cout << "\n   Sorry, the file could not be opened.";
        }
}

void FileManager::replaceWord()
{
        int wordPos, wordLen, check = 0;
        string word, replacement, line, spaces;

        inObj.open(fileName);
        if (inObj.good())
        {
                outObj.open("2.txt", ios::trunc);                              //Ensuring copy file
is empty
                outObj.close();
                cout << "\n\n   Enter the word you want to replace: ";
                cin >> word;
                wordLen = word.length();
                cout << "\n   Enter the word you want to replace '" << word << "' with: ";
                cin >> replacement;

                while (!inObj.eof())
                {
                        getline(inObj, line);

                        if (line.find(word) != -1)
                        {
                                check = 1;
                                wordPos = line.find(word);
                                while (wordPos < line.length())
                                {
                                        line.replace(wordPos, wordLen, replacement);
                                        wordPos = line.find(word);
                                }
                        }
                        outObj.open("2.txt", ios::app);                                //Creating a
separate file because storing all lines in string would consume too much memory and will be
really inefficient
                        outObj << line << endl;
                        outObj.close();
                }
                inObj.close();
```

```
		if (check == 1)
		{
			outObj.open(fileName, ios::trunc);			//Removing all data from original file
			inObj.open("2.txt", ios::in);

			while (!inObj.eof())					//Sending all data back to original file from copy file because i dont want to delete and rename file in every execution as that could create a lot of waste
			{
				getline(inObj, line);
				outObj << line << endl;
			}

			outObj.close();
			inObj.close();

			outObj.open("2.txt", ios::trunc);			//removing all data from "2.txt"
			outObj.close();
		}
		else
		{
			cout << "\n\n   Sorry. The entered word does not exist in file.\n";
		}
	}
	else
	{
		cout << "\n   Sorry, the file could not be opened.";
	}
}

void FileManager::modifyAndSave()
{
	outObj.open(fileName, ios::app);
	if (outObj.good())
	{
		int lines;
		cout << "\n\n   Please enter the number of lines of data you want to add to the file: ";
		cin >> lines;
		string* dataLines = new string[lines];

		for (int i = 0; i < lines; ++i)
		{
			cin.ignore();
			cout << "\n   Enter data for line " << i + 1 << ": ";
			getline(cin, dataLines[i]);
			outObj << dataLines[i] << endl;
		}
		delete[] dataLines;
		outObj.close();
	}
	else
	{
		cout << "\n   Sorry, the file could not be opened.";
	}
}
```

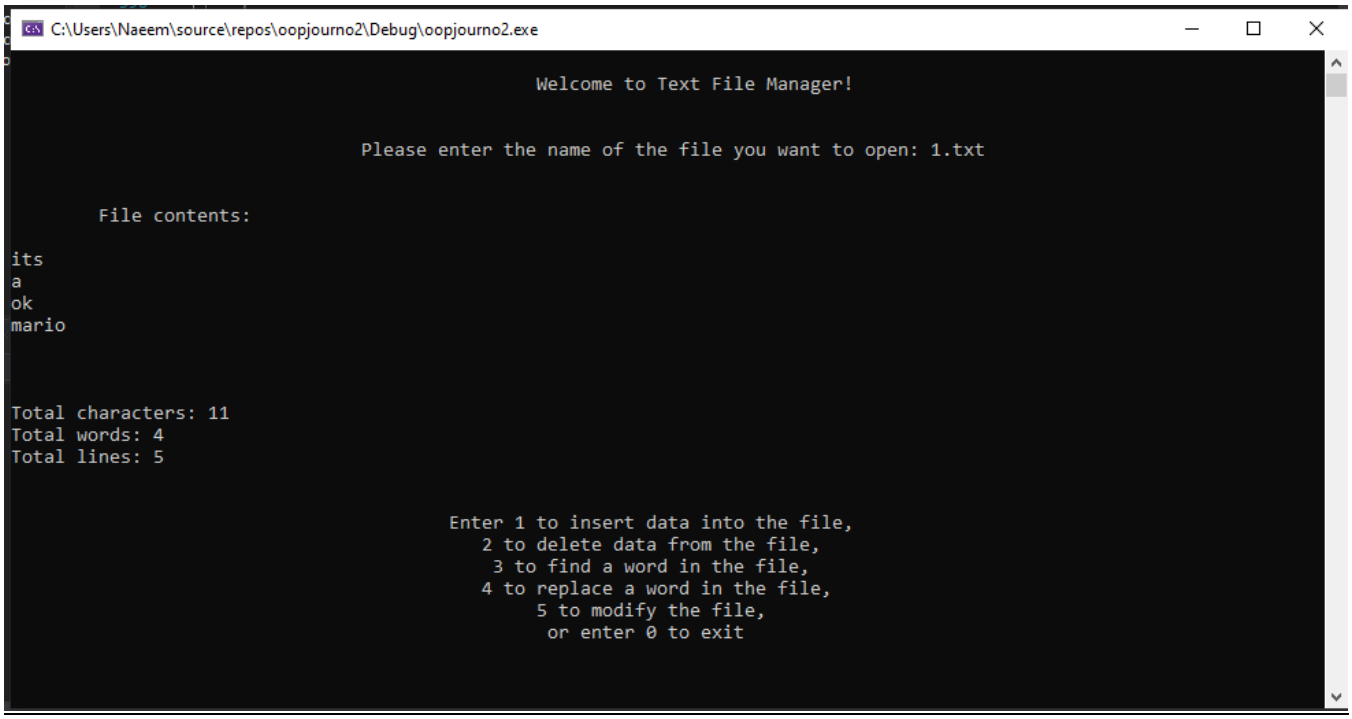
Tools Used: Visual Studio 2019, Microsoft Word

Sample Program Outputs:

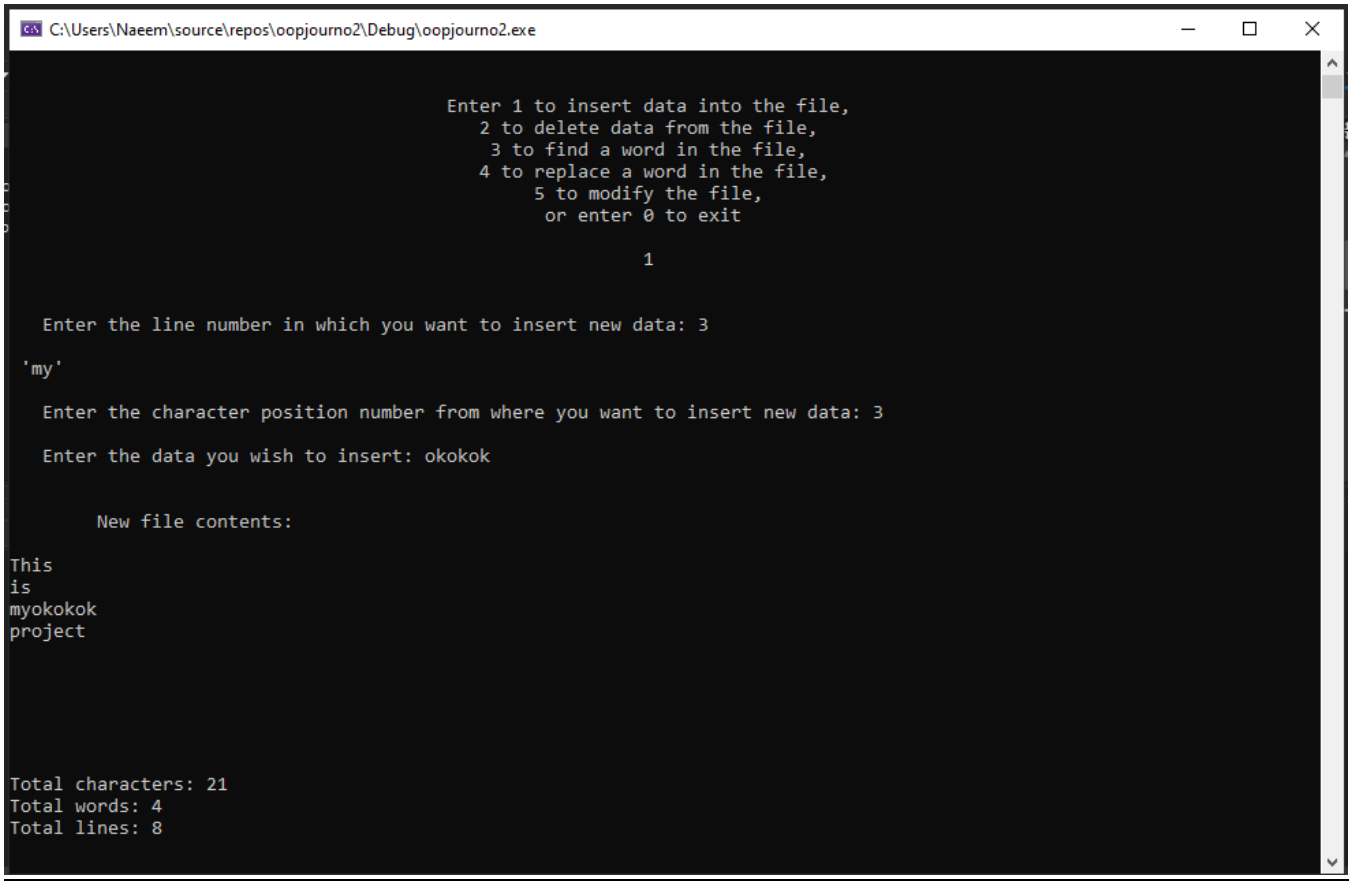
Welcome screen:



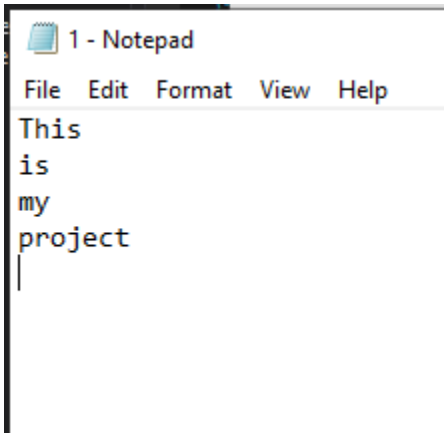
Open, print and count functions (+ Main Menu) :

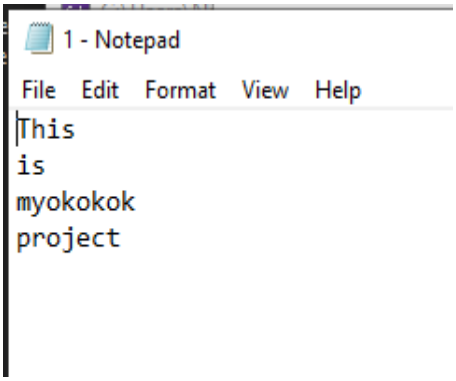


Insert function:



Text file before and after insertion

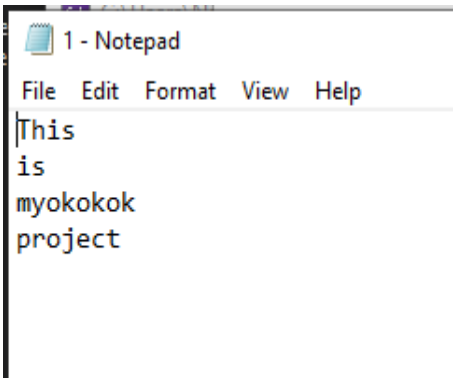


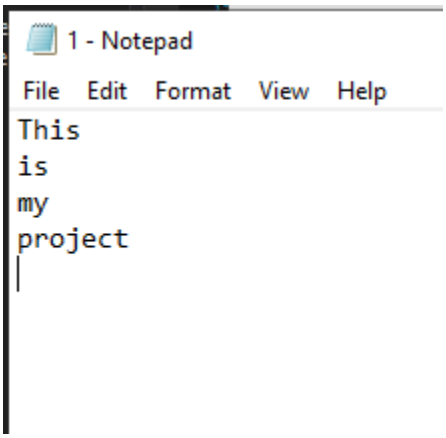


Delete function:

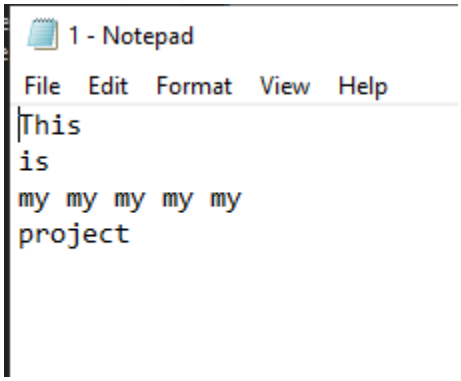
```
C:\Users\Naeem\source\repos\oopjourno2\Debug\oopjourno2.exe

                                    Enter 1 to insert data into the file,
                                       2 to delete data from the file,
                                        3 to find a word in the file,
                                       4 to replace a word in the file,
                                            5 to modify the file,
                                             or enter 0 to exit

                                                    2


   Enter the line number from which you want to delete data: 3

 'myokokok'

   Enter the character position number from where you want to delete data: 3

   Enter the number of characters you want to delete: 6


         New file contents:

This
is
my
project




Total characters: 15
Total words: 4
Total lines: 10
```

Text file before and after deletion

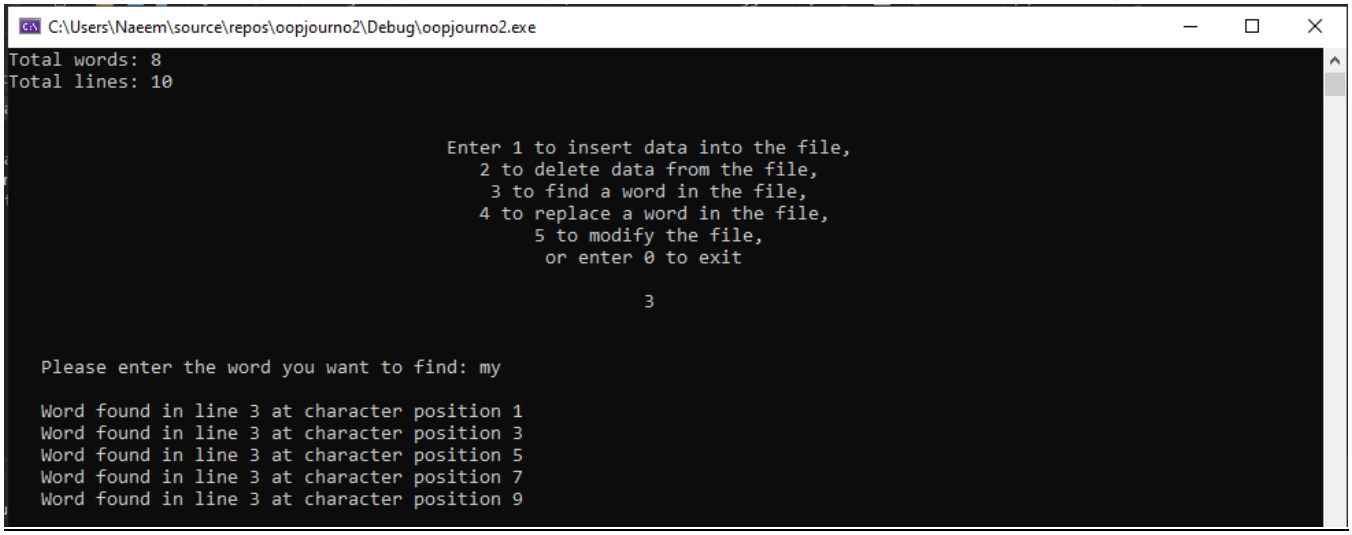




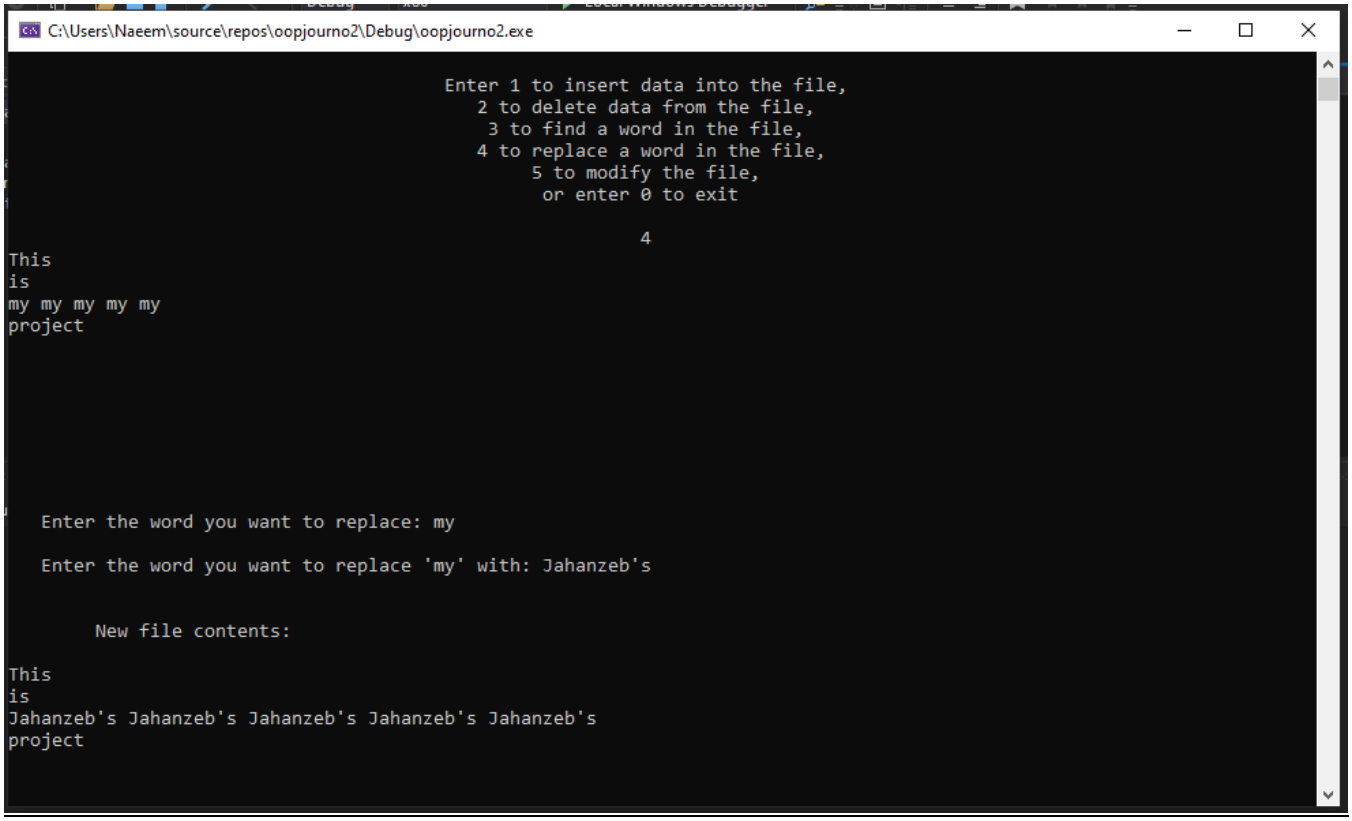
Find function:



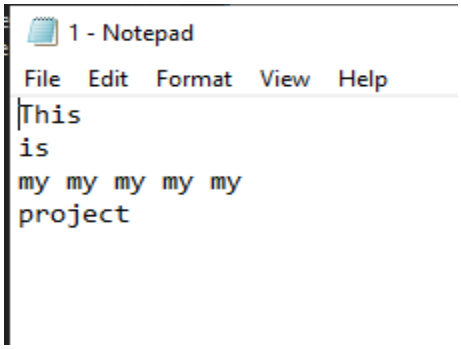
Multiple instances in same line

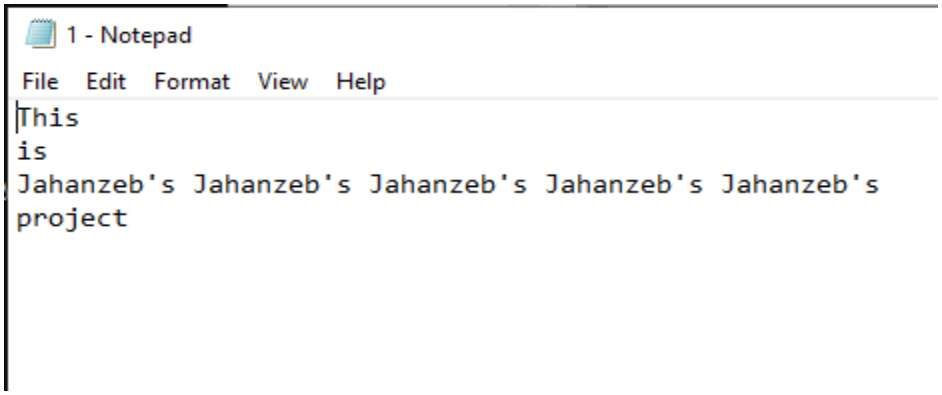


Replace function:



Text file before and after replacing

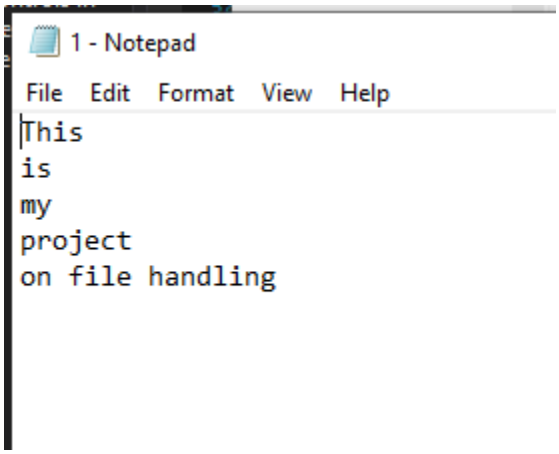




Modify and save function:

```
                                    Enter 1 to insert data into the file,
                                       2 to delete data from the file,
                                        3 to find a word in the file,
                                       4 to replace a word in the file,
                                            5 to modify the file,
                                             or enter 0 to exit

                                                      5


   Please enter the number of lines of data you want to add to the file: 1

   Enter data for line 1: on file handling


         New file contents:

This
is
my
project
on file handling
```

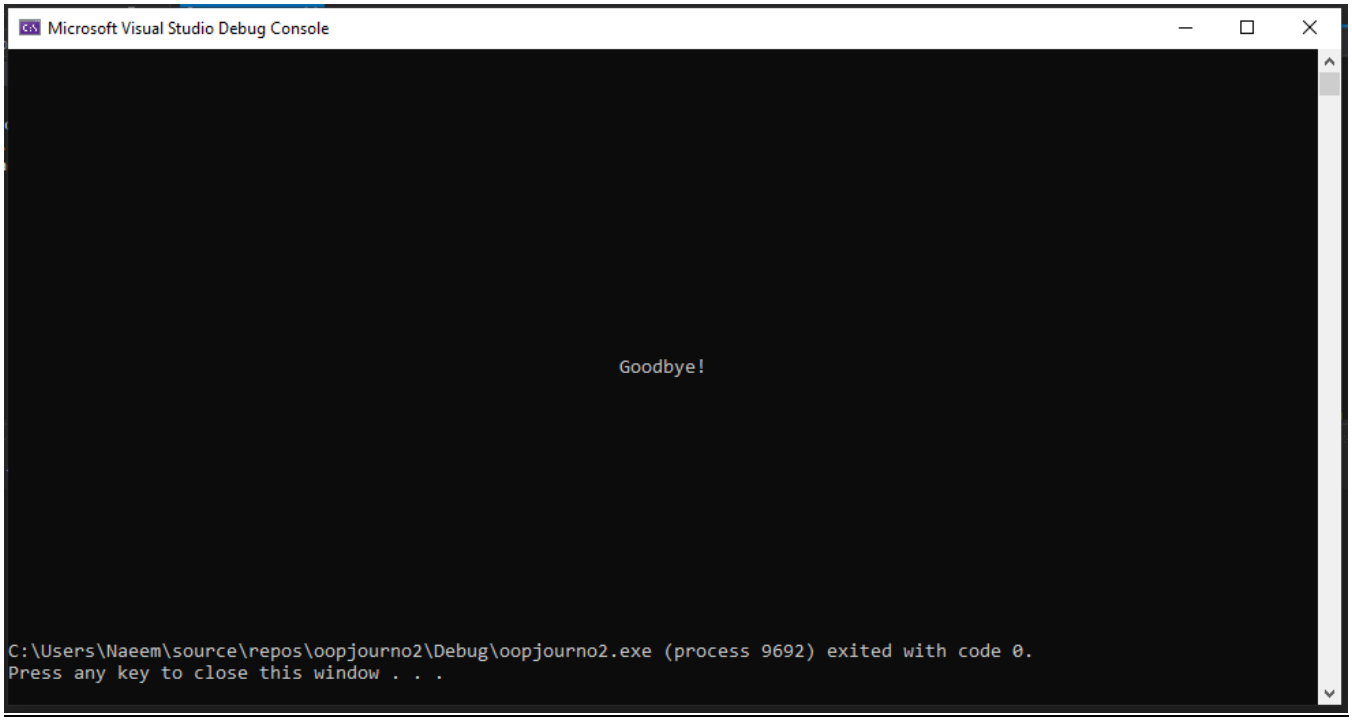
Text file after modifying



Exit:

```
                                    Enter 1 to insert data into the file,
                                       2 to delete data from the file,
                                        3 to find a word in the file,
                                       4 to replace a word in the file,
                                            5 to modify the file,
                                             or enter 0 to exit

                                                      0
```

Exit Screen:



Conclusion:

It can be seen that it is indeed possible to create such a text file editor using UML diagram, classes and text files.