Отчет по лабораторной работе №5.

Программа состоит из 4 файлов:

- main.cpp;
- Fynk.cpp;
- Header.h (заголовочный файл);
- Makefile.

Файл main.cpp

В файле подключены заголовочные файлы <iostream>, <cstring> и заголовочный файл Header.h, в котором содержатся функции объявленные в Fynk.cpp (Fromfile(), Tofile(), Fromtofile()) и глобальные переменные char**arg, int arc.

В файле main.cpp содержится функция int main (int argc, char ** argv). При запуске пользователь может ввести флаги “--fromfile”, “--tofile”, а также путь к файлу, сначала из которого будет считан текст, после куда будет записан. Предусмотрено также некорректное введение флагов:

- введены флаги для чтения из файла и запись в файл (проверка на количество и корректность ввода флага)
  ```
  if (argc >= 4 && argc <=5 && strcmp(argv[1], "--fromfile") == 0 && strcmp(argv[2], "--tofile") == 0) Fromtofile();
  ```
- введен флаг для чтения из файла без записи в файл (проверка на количество элементов и корректность ввода флага)
  ```
  else if (argc == 3 && strcmp(argv[1], "--fromfile") == 0 && strcmp(argv[2],"--tofile")!=0) Fromfile();
  ```
- введен флаг для чтения с консольной строки и выводом в файл (проверка на количество элементов и корректности ввода)
  ```
  else if (argc >=2 && argc <=3 && strcmp(argv[1], "--tofile") == 0) Tofile();
  ```
- в любом другом случае программа выдаст ошибку с пояснением, что введено слишком много флагов или мало или ввод некорректный

```
else std::cout << "Error! You entered the flags either
incorrectly, or in the wrong sequence, or too little." <<
std::endl;
```

Во всех, кроме последнего случая, программа переходит к выполнению нужной функции, которые находятся в файле Fynk.cpp.

Файл Header.h

Заголовочный файл включает в себя название всех функции находящихся в файле Fynk.cpp, а также глобальных переменный int agc, char** arg, хранящих в себе данные переменных функции int main() из файла main.cpp.

Файл Fynk.cpp

В файле Fynk.cpp находятся все функции для удачного выполнения программы. Подключены заголовочные файлы <iostream>, <fstream>, а также заголовочный файл Header.h.

1) Fromfile() – предназначена для чтения из файла и вывода результата в командную строку.
   a) Открытие файла. Для открытия файла для чтения используется ifstream. При неудачном открытие программа выдает ошибку с пояснением и завершается.

```
std::ifstream file;
file.open(arg[2]);
if (!file) {
     std::cout << "Error! Can not open file!" << std::endl;
return 0;
}
```

   b) Посимвольное чтения из файла производится с помощью get(). Программа выдаст ошибку, если введено слишком большое кол-во символов и завершится. При внесение значений производится замена ‘\n’ на ‘\0’, то есть замена переноса строки на пробел.

```
char stroka[100000];
```

```
char k;
int i = 1, j, h = 0, chit = 0, p;
while (file.get(k)) {
     if (i == 1000001) {
          std::cout << "Enter a shorter line" << std::endl;
          return 0;
     }
     if (k == '\n') stroka[i - 1] = '\0';
     else stroka[i - 1] = k;
     i += 1;
}
file.close();
```

c) Фиксация начала предложения и количества символов в нем с помощью точки. Также запоминается количество точек, т.е. предложений.

```
int tochki[1000000];
int tmp;
for (j = 0; j < i; j++) {
     if (stroka[j] == '.') {
          tochki[h] = j - h;
          h = j + 2;
          chit += 1;
     }
     else tochki[j] = 0;
}
```

d) Вывод предложений производится с помощью нахождения минимального количества символов и вывода строки, а после удаления информации о ней и так до того, как цикл не повторится столько раз, сколько у нас строк.

```
int nine[1000000];
int minn = 1000000, mini;
for (j = 0; j < chit; j++) {
     for (p = 0; p < i; p++) {
          if (tochki[p] != 0 && tochki[p] < minn) {
               minn = tochki[p];
               mini = p;
          }
     }
     tmp = mini;
     for (p = mini; p <= mini + minn + 1;p++) {
          if (j == chit - 1 && (stroka[p-1]==' ' || stroka[p-
          1]=='.')){
               nine[tmp] = p - tmp - 1;
               tmp = p;
          }
```

```
            else nine[p] = 0;
            std::cout << stroka[p];
        }
        std::cout << std::endl;
        }
        tochki[mini] = 0;
        minn = 1000000;
    }
```

e) Дополнительный цикл создан для вывода в файл NINE.txt самой большой строки так, что ее слова записаны в обратном порядке. Фиксируется это при помощи достижения j максимально большого номера и началом слова с помощью нахождения перед них пробела. Первое слово фиксируется при помощи mini из пункта d).

```
tmp = mini;
for (p = mini; p <= mini + minn + 1;p++) {
            if (j == chit - 1 && (stroka[p-1]==' ' || stroka[p-
    1]=='.')){
                nine[tmp] = p - tmp - 1;
                tmp = p;
            }
            else nine[p] = 0;
            std::cout << stroka[p];
}
std::cout << std::endl;
if (j == chit - 1) {
    std::ofstream file3("NINE.txt");
    for (p = i - 1;p>= 0; p--) {
            if (nine[p] != 0) {
                file3 << ' ';
                for (h=p;h<p+ nine[p]; h++) {
                    file3 << stroka[h];
                }
            }
    }
    file3.close();
}
```

2) Tofile() – предназначена для считывания текста из консольной строки и записи в файл.

a) Открытие пользовательского файла или Tofile.txt, если пользователь не ввел адрес. Выводится ошибка, если не получилось открыть нужный файл.

```
std::ofstream file;
if (arc == 3) file.open(arg[2]);
```

```
else file.open("Tofile.txt");
if (!file) {
	std::cout << "Error! Can not open file!" << std::endl;
	return 0;
}
```

b) Чтение символов из консоли осуществляется до того момента, пока пользователь не введет ‘|’, что программа воспримет как конец текста. Также при записи символов в массив перенос строки заменяется на пробел.

```
std::cout << "Enter the text and at the end, after the
period, put '|'" << std::endl;
char stroka[100000];
char k;
int i = 1, j, h = 0, chit = 0, p;
for (j = 0; j <= 100000; j++) {
	if (j == 100000) {
		std::cout << "Enter a shorter line" << std::endl;
		return 0;
	}
	k = getchar();
	if (k == '|') break;
	if (k == '\n') stroka[j] = '\0';
	else stroka[j] = k;
	i = j + 1;
}
```

c) Фиксация начала предложения производится так же, как и в функции Fromfile() (см. 1.c).

d) Запись в файл производится так же, как и в функции Fromfile(), однако std::cout заменяется на file. (см.1.d).

e) Так же, как и в функции Fromfile() присутствует запись самой больший строки в файл NINE.txt в обратном порядке, используется тот же механизм. (см. 1.e)

3) Fromtofile() – функция для считывания текста из файла и записи в файл. В ней используются те же алгоритмы, что и в функциях Fromfile() и Tofile() для считывания и записи в файле. Также присутствует алгоритм записи наибольшей строки наоборот в файл NINE.txt.

Файл Makefile

Файл создан для компиляции программы. Реализованы методы clean, который удаляет файлы с расширением .o, и distclean, который удаляет все файлы.

```
main : main.o Fynk.o
	g++ main.o Fynk.o -o main

main.o : main.cpp
	g++ -c main.cpp -o main.o

Fynk.o : Fynk.cpp
	g++ -c Fynk.cpp -o Fynk.o

clean:
	rm main.o Fynk.o

distclean: clean
	rm main
```