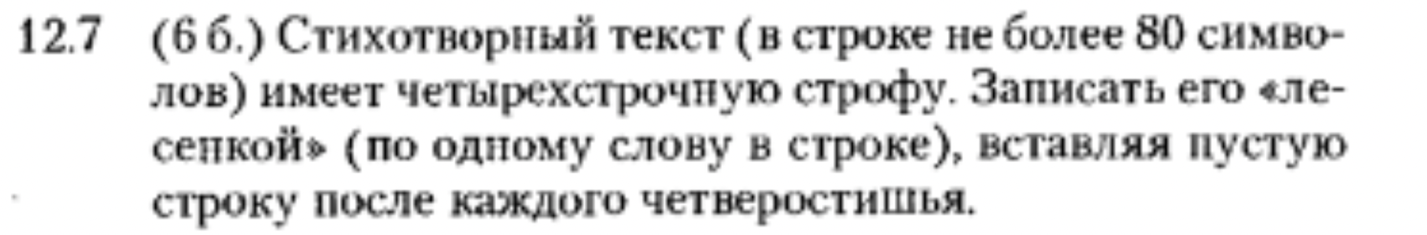
**Отчет курсанта Громова Григория Андреевича группы 22.Б05 о выполнении практического задания на 24.11.2022**

****

Алгоритм (в комментариях к тесту программы алгоритм представлен более детально):

1. Открываем файл
2. Пока остались несчитанные строки, последовательно и поочередно копируем строки в массив line
3. Пока строка не закончилась, последовательно продвигаемся по всем символам строки: если это символ пробела, то переводим курсор на новую строку, во всех остальных случаях печатаем этот символ на экран
4. Закрываем файл

Текст программы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h> //библиотка для работы со строками

int main(){

char line[80]; //сюда будем поочередно помещать строки

FILE \*file = fopen("stih.txt", "r"); //открываем файл в режиме чтения

int i;

int k = 0;

int len;

while (fgets(line, 100, file)){ //пока удается считать строку

len = strlen(line); //длина строки

for (i = 0; i < len; i++){ //пока мы не дошли до конц строки

if (line[i] != ' '){ //пока мы не дошли до пробела (то есть пока не закончилось слово)

printf("%c", line[i]); //посимвольно выписываем слово

}

else{ //в противном случае (когда слово закончилось)

printf("\n"); //переставляем курсор на следующую строку

}

}

k++;

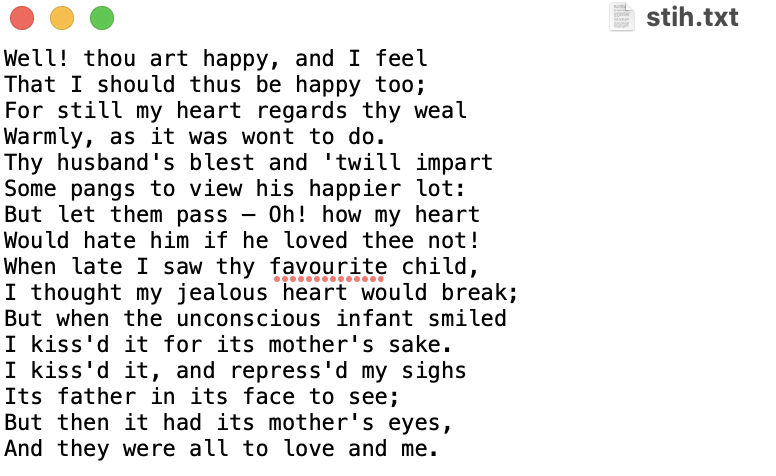
if (k % 4 == 0){ //через каждые 4 строки, одну пропускаем

printf("\n");

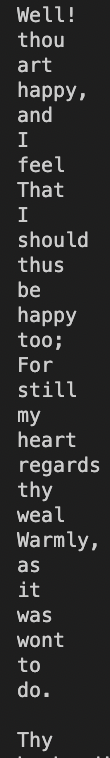
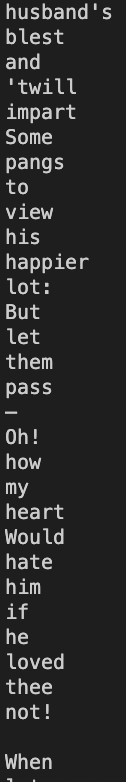
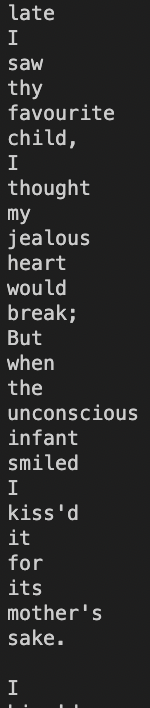
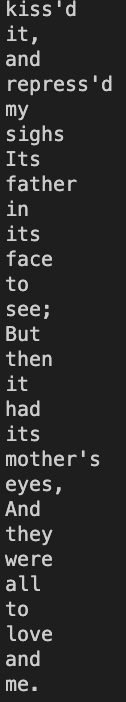
}

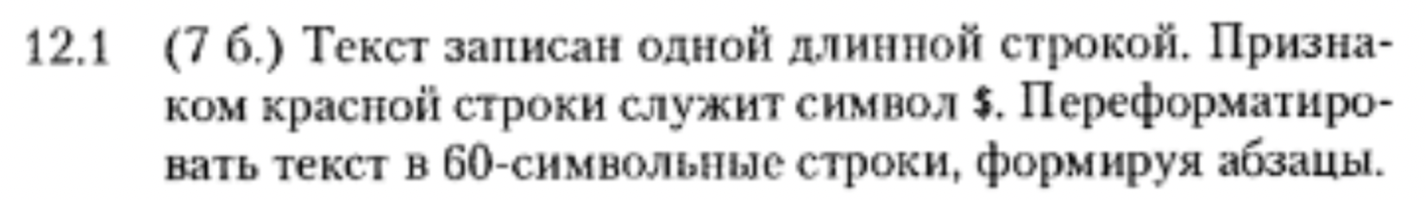
}

Проверим работу программы на стихотворении лорда Байрона “Well! Thou Art Happy”:



Программа работает корректно:



Алгоритм:

1. Открываем файл, в котором лежит одна строка
2. Считываем строку из файла в заранее заготовленный длинный массив
3. Посимвольно выводим строку, каждые 60 символов, переводя курсор на новую строку. Однако, если встречается символ «$», мы сразу переводим курсор на новую строку и ставим 8 пробелов.
4. Закрываем файл

Текст программы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h> //библиотка для работы со строками

int main(){

char line[4000]; //сюда поместим строку

FILE \*file = fopen("text2.txt", "r"); //открываем файл в режиме чтения

int i;

int k = 0;

int len;

fgets(line, sizeof(line), file); //считываем строку и записываем

len = strlen(line); //длина строки

int counter = 0;

for (i = 0; i < len; i++){ //пока мы не дошли до конца строки

if (line[i] != '$'){ //пока мы не дошли до $

printf("%c", line[i]); //посимвольно выписываем слово

counter++;

if (counter > 60){

printf("\n");

counter = 0;

}

}

else{ //в противном случае (когда $ встретился)

printf("\n"); //переставляем курсор на следующую строку

printf("\t"); //красная строка

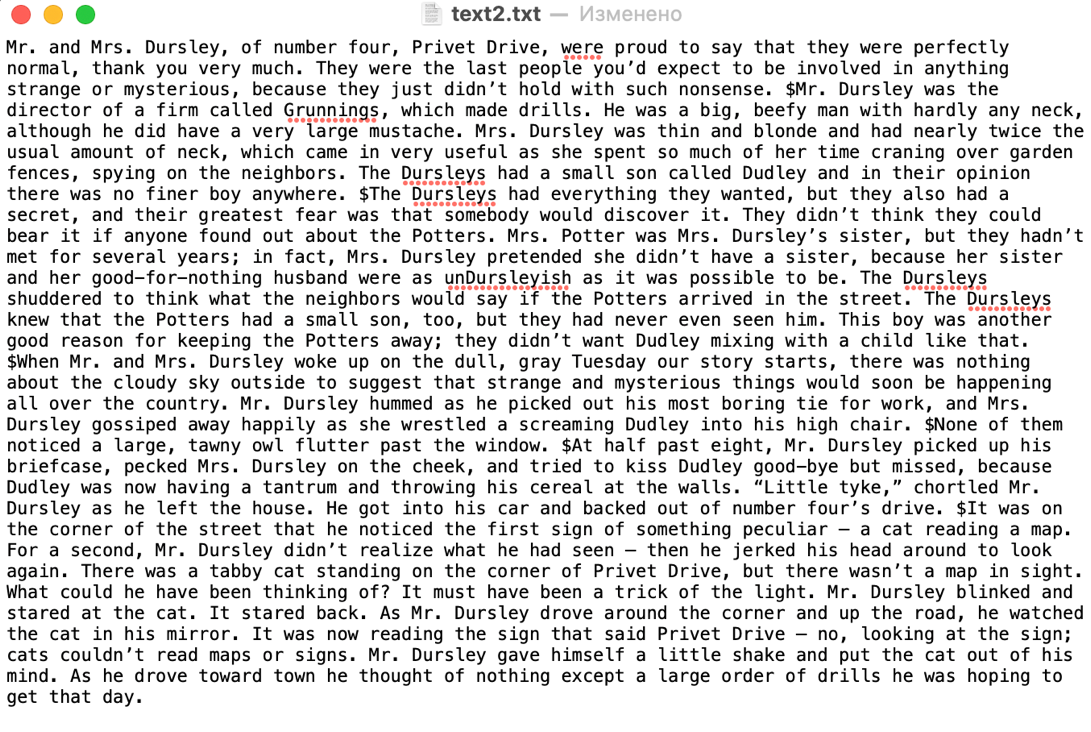
counter = 8; //компенсируем длину красной строки

}

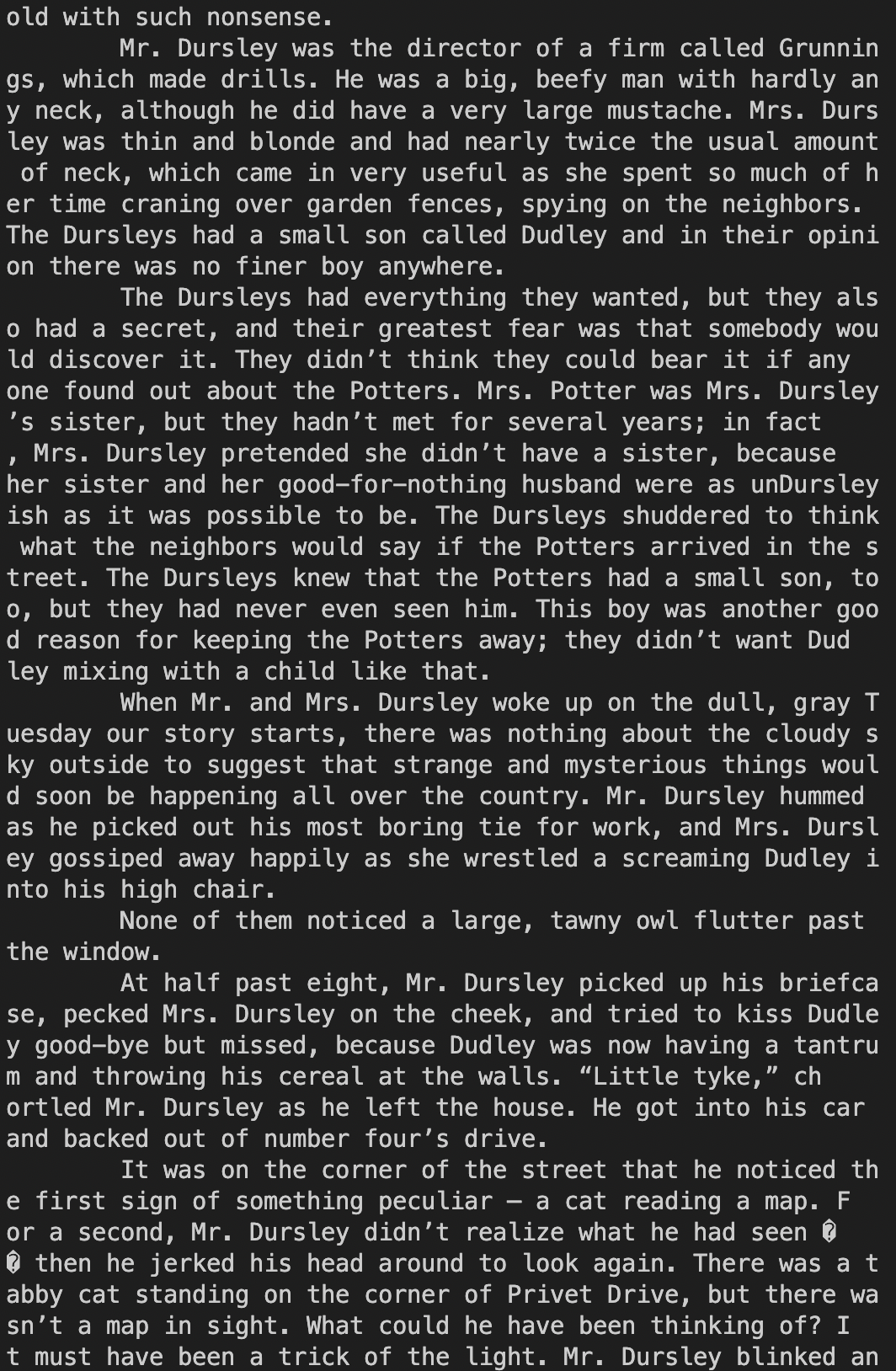
}

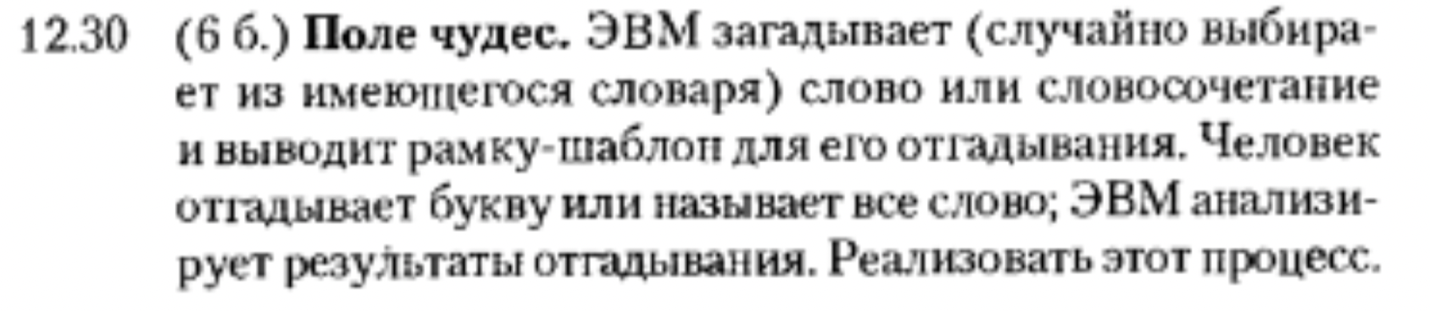
}

Протестируем программу на тексте из книги «Гарри Поттер»:



С помощью сервиса <https://lingojam.com/TexttoOneLine> я перевел этот фрагмент в одну строку. После этого вручную расставил символы “$” на месте абзацев. Программа выдала корректный результат:





Алгоритм:

1. Создаем словарь
2. Случайным образом выбираем оттуда одно слово
3. Выводим на экран его маску и предлагаем пользователю угадать его буквы
4. Если пользователь ввел букву, содержащуюся в слове, выводим измененную маску, в которой на месте пропуска стоит угаданная буква
5. Если буквы нет, говорим об этом пользователю
6. Когда он угадал все буквы, поздравляем его с победой, и завершаем прорамму

Текст программы:

#include <stdio.h>

#include <stdlib.h>

#include <string.h> //библиотека для работы со строками

#include <time.h> //библиотека, которая будет работать для улучшения функции rand()

int size(char word[]){ //функция, определяющая количество букв в слове из словаря

int i = 0;

while (word[i] != '$'){ //пока не встретим символ конца слова, увеличиваем значение длины

i++;

}

return i;

}

int main(){

//словарь введен весьма специфически, потому что пока я не знаю, как это сделать правильней

char \*slov[] = {"example$", "horse$", "paper$", "window$", "street$", "house$", "void$", "density$", "amount$", "enigma$"};

int x;

srand(time(NULL)); //это функция делает функцию rand() зависимой от времени

x = (rand()) % 10; //выбираем случайное слово из нашего "словаря"

int len = size(slov[x]); //записываем сюда количество букв в этом слове

int i;

printf("This word contains %d letters\n", len);

for (i = 0; i < len; i++){ //выводим число пропусков, соответствующее количеству букв в данной слове

printf("\_ ");

}

printf("\n\n");

char l;

char M[len]; //создаем массив длиной равной количесеству букв в слове

for (i = 0; i < len; i++){ //заполняем его "\_"

M[i] = '\_';

}

int flag = 0;

while(1){

printf("Add the letter: "); //просим пользователя ввести букву

scanf("%c", &l); //записываеме ее в перемнную l

while(getchar() != '\n'); //очистим строку от ввода \n, противном случае, каждый второй раз scanf игнорируется

//если введенная буква есть в нашем слове, в массиве заменим "\_" на эту букву, в месте, соответствующим порядку

//этой буквы в слове:

int co = 0;

for (i = 0; i < len; i++){

if (slov[x][i] == l){ //если такая замена произошла

M[i] = l;

co = 1; //запишем, что она произошла

}

}

if (co == 1){ //если замена произошла

printf("You are right! Now you have:\n");

for (i = 0; i < len; i++){ //выводим модефицированный массив

printf("%c ", M[i]);

}

printf("\n\n");

}

else{ //если нет

printf("Letter %c is not in this word\n", l);

printf("\n");

}

//Проверка на заполненность массива:

flag = 0;

for (i = 0; i < len; i++){

if (M[i] == '\_'){ //если пропусков нет, значит мы отгадали все буквы

flag = 1;

}

}

if (flag == 0){ //если пользователь отгадал все буквы

printf("Congratulations, you guessed the word!\n"); //нужно поздравить его с этим

break; //и выйти из цикла

}

}

}

Тест:

