Міністерство освіти і науки України
Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського"
Факультет інформатики та обчислювальної техніки

Кафедра інформатики та програмної інженерії

Звіт

з лабораторної роботи № 1 з дисципліни
«Основи програмування 2. Модульне програмування»
«Текстові файли»
Варіант 7

Виконав студент ІП-15, Гуменюк Олександр Володимирович

(шифр, прізвище, ім'я, по батькові)

Перевірила Вєчерковська Анастасія Сергіївна

(прізвище, ім'я, по батькові)

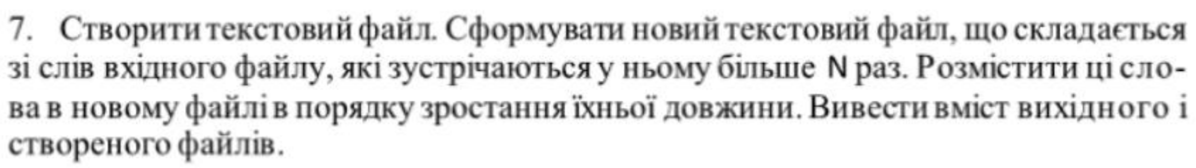
Київ 2022

Лабораторна робота 1

Текстові файли

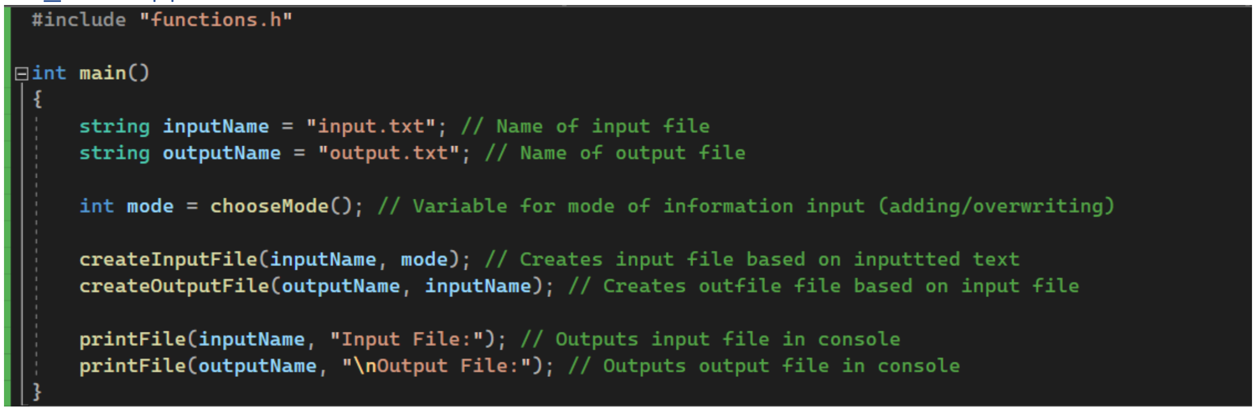
Індивідуальне завдання

Варіант 7

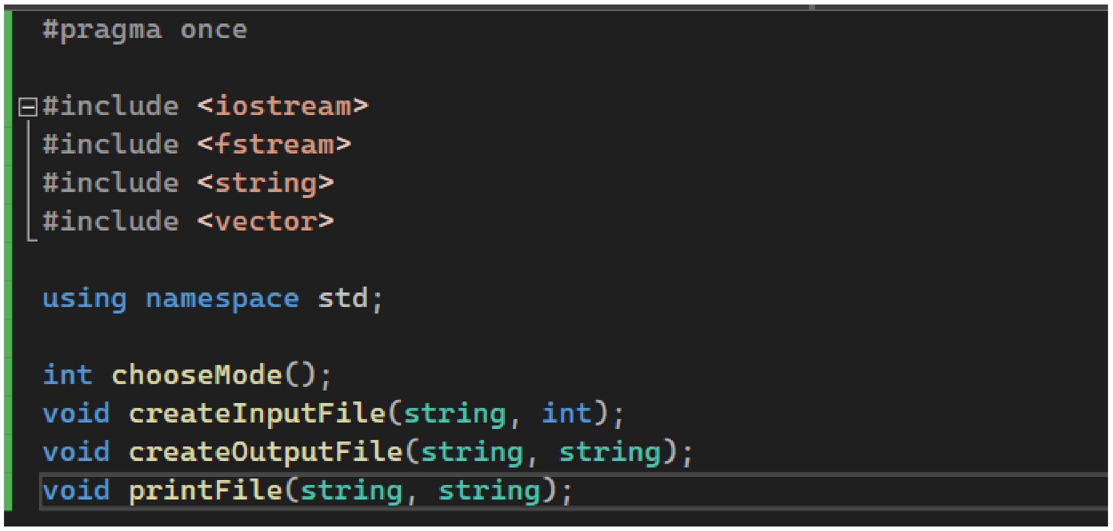

7. Створити текстовий файл. Сформувати новий текстовий файл, що складається зі слів вхідного файлу, які зустрічаються у ньому більше N раз. Розмістити ці слова в новому файлі в порядку зростання їхньої довжини. Вивести вміст вихідного і створеного файлів.

Код C++

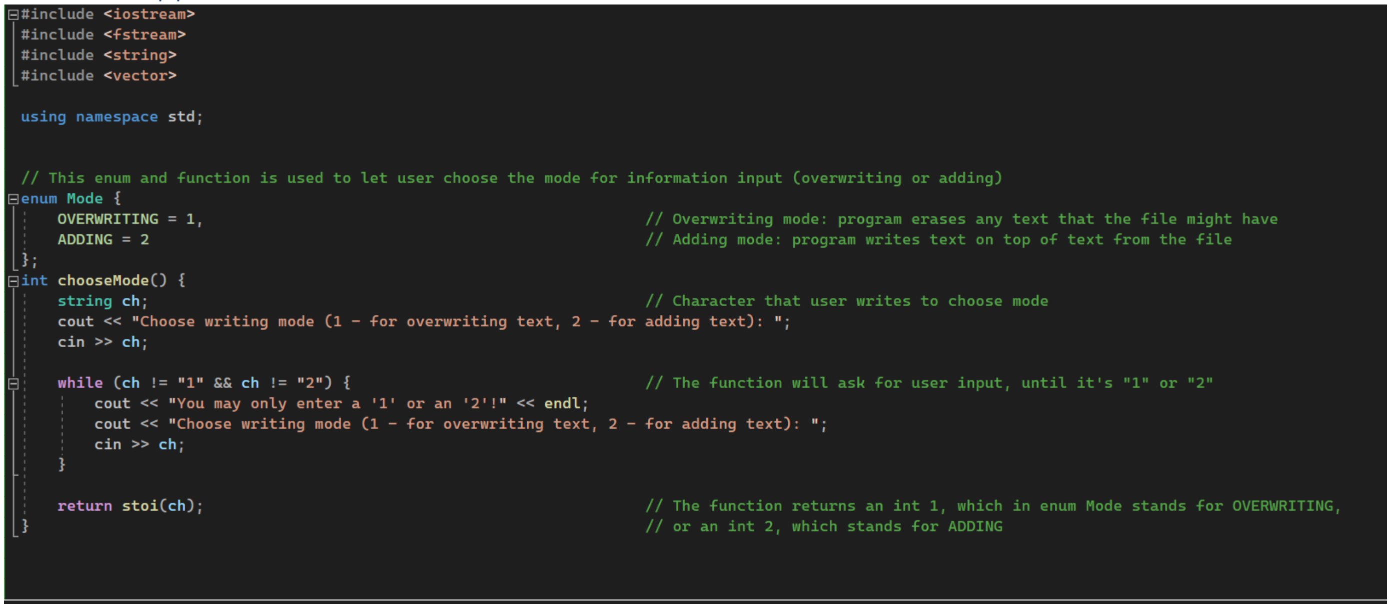
OP_Lab1.cpp


```cpp
#include "functions.h"

int main()
{
    string inputName = "input.txt"; // Name of input file
    string outputName = "output.txt"; // Name of output file

    int mode = chooseMode(); // Variable for mode of information input (adding/overwriting)

    createInputFile(inputName, mode); // Creates input file based on inputtted text
    createOutputFile(outputName, inputName); // Creates outfile file based on input file

    printFile(inputName, "Input File:"); // Outputs input file in console
    printFile(outputName, "\nOutput File:"); // Outputs output file in console
}
```

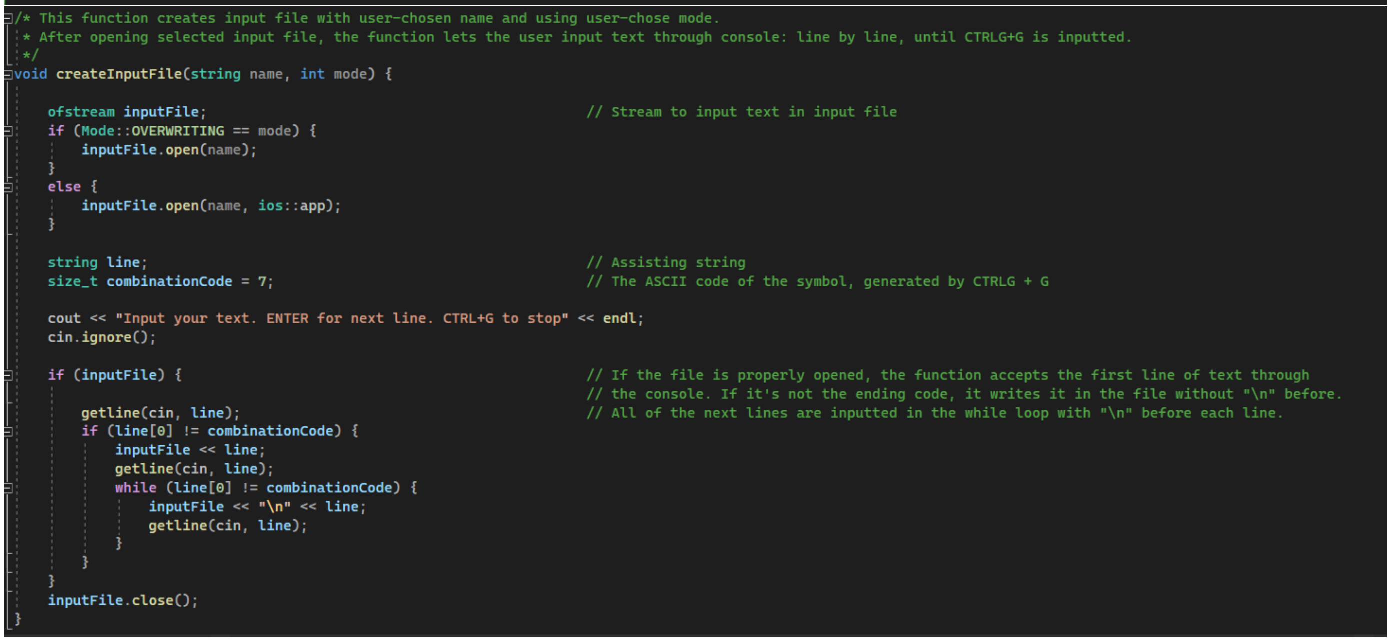
functions.h


```cpp
#pragma once

#include <iostream>
#include <fstream>
#include <string>
#include <vector>

using namespace std;

int chooseMode();
void createInputFile(string, int);
void createOutputFile(string, string);
void printFile(string, string);
```

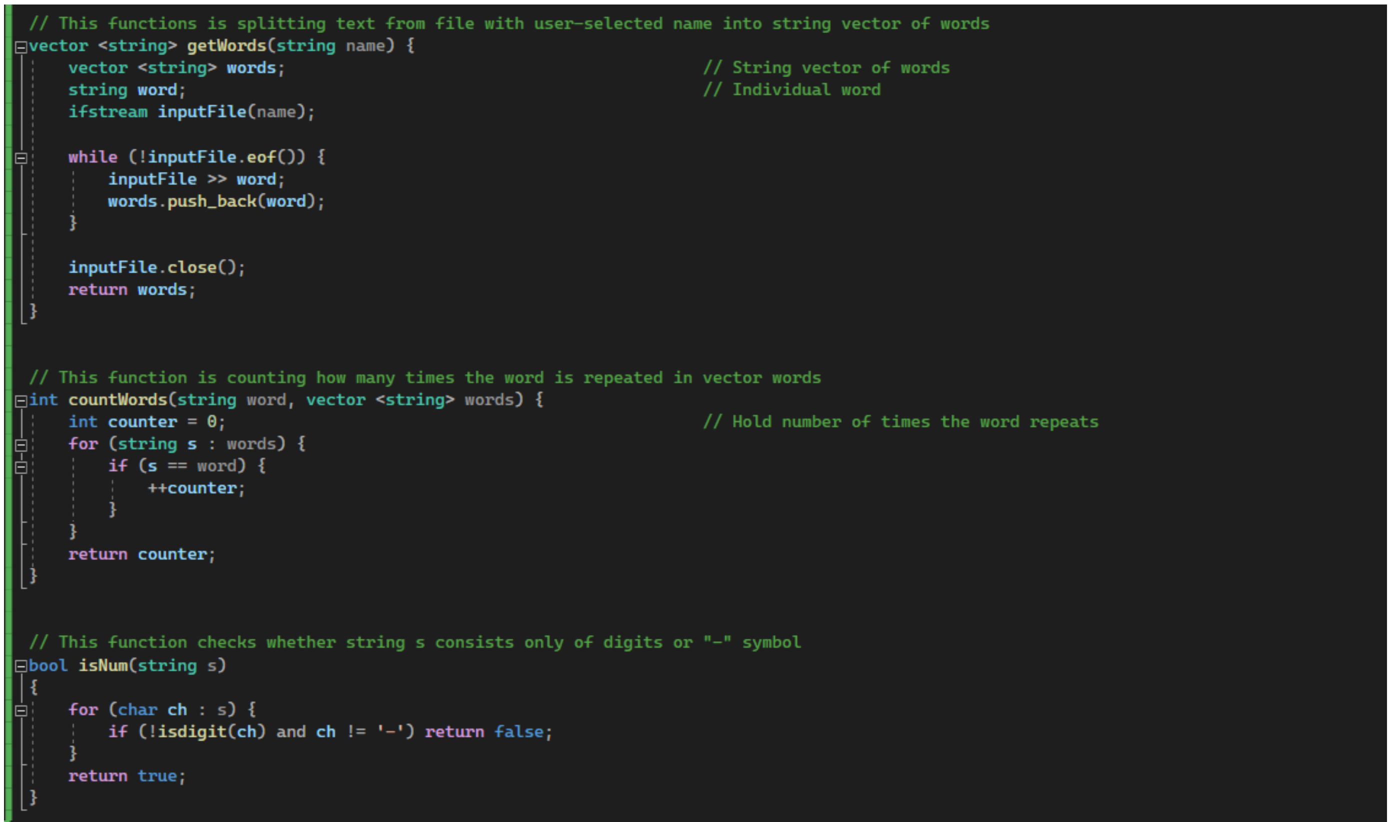
functions.cpp



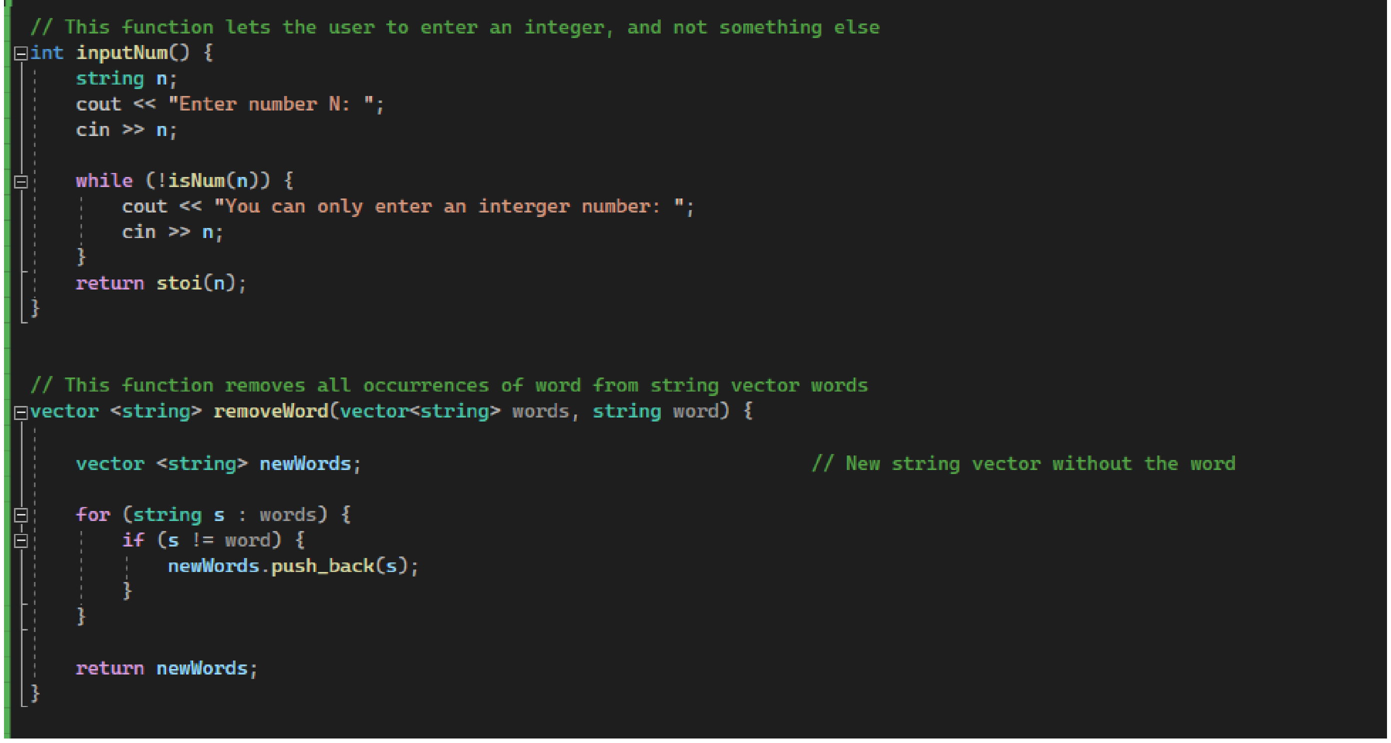
```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>

using namespace std;


// This enum and function is used to let user choose the mode for information input (overwriting or adding)
enum Mode {
    OVERWRITING = 1,                                                // Overwriting mode: program erases any text that the file might have
    ADDING = 2                                                      // Adding mode: program writes text on top of text from the file
};
int chooseMode() {
    string ch;                                                      // Character that user writes to choose mode
    cout << "Choose writing mode (1 - for overwriting text, 2 - for adding text): ";
    cin >> ch;

    while (ch != "1" && ch != "2") {                                // The function will ask for user input, until it's "1" or "2"
        cout << "You may only enter a '1' or an '2'!" << endl;
        cout << "Choose writing mode (1 - for overwriting text, 2 - for adding text): ";
        cin >> ch;
    }

    return stoi(ch);                                                // The function returns an int 1, which in enum Mode stands for OVERWRITING,
}                                                                   // or an int 2, which stands for ADDING
```



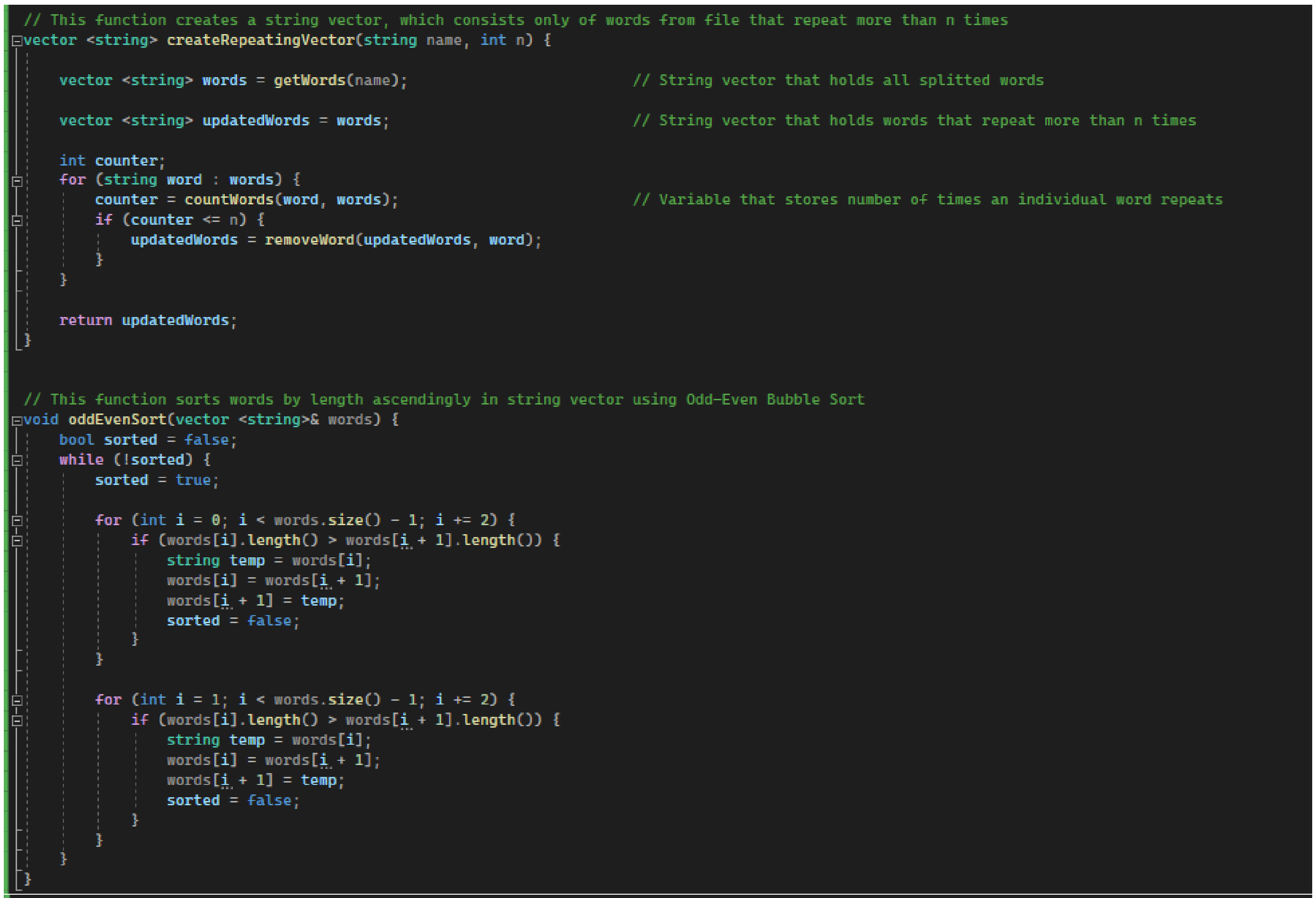
```cpp
/* This function creates input file with user-chosen name and using user-chose mode.
 * After opening selected input file, the function lets the user input text through console: line by line, until CTRLG+G is inputted.
 */
void createInputFile(string name, int mode) {

    ofstream inputFile;                                                  // Stream to input text in input file
    if (Mode::OVERWRITING == mode) {
        inputFile.open(name);
    }
    else {
        inputFile.open(name, ios::app);
    }

    string line;                                                         // Assisting string
    size_t combinationCode = 7;                                          // The ASCII code of the symbol, generated by CTRLG + G

    cout << "Input your text. ENTER for next line. CTRL+G to stop" << endl;
    cin.ignore();

    if (inputFile) {                                                     // If the file is properly opened, the function accepts the first line of text through
                                                                         // the console. If it's not the ending code, it writes it in the file without "\n" before.
        getline(cin, line);                                              // All of the next lines are inputted in the while loop with "\n" before each line.
        if (line[0] != combinationCode) {
            inputFile << line;
            getline(cin, line);
            while (line[0] != combinationCode) {
                inputFile << "\n" << line;
                getline(cin, line);
            }
        }
    }
    inputFile.close();
}
```



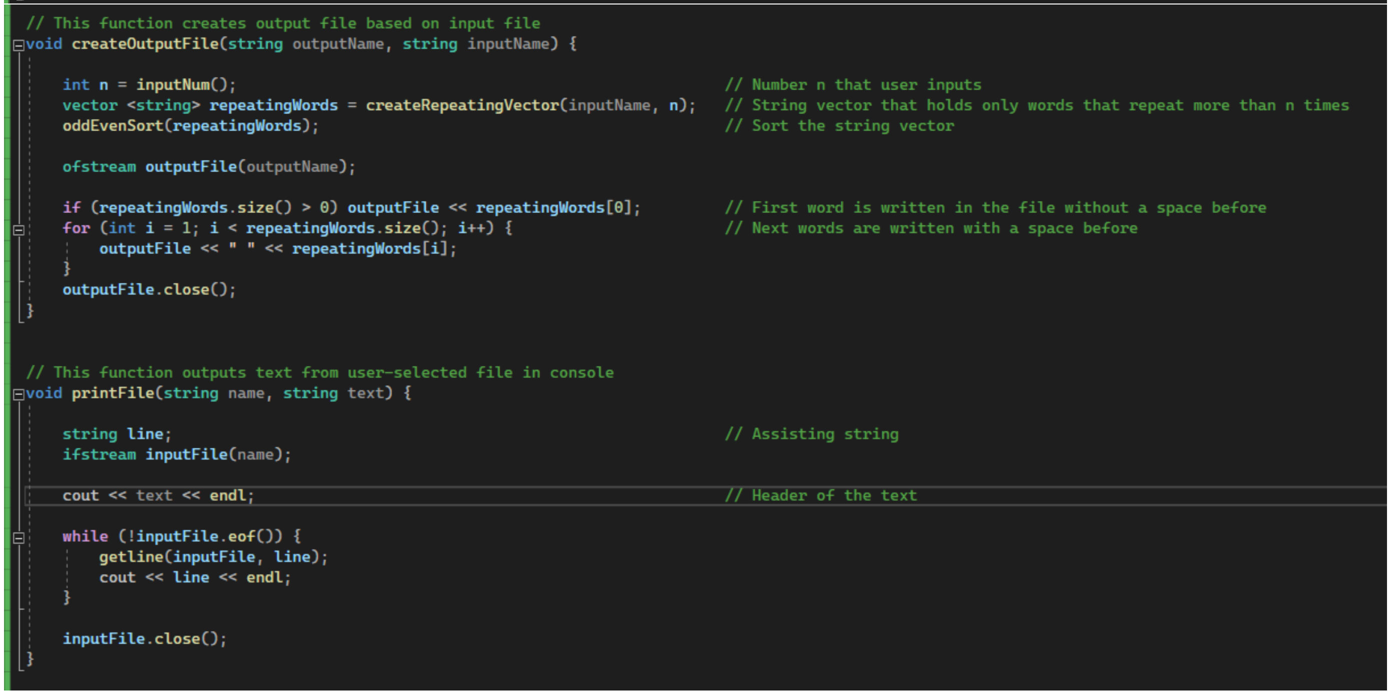
```
// This functions is splitting text from file with user-selected name into string vector of words
vector <string> getWords(string name) {
    vector <string> words;                                      // String vector of words
    string word;                                                // Individual word
    ifstream inputFile(name);

    while (!inputFile.eof()) {
        inputFile >> word;
        words.push_back(word);
    }

    inputFile.close();
    return words;
}


// This function is counting how many times the word is repeated in vector words
int countWords(string word, vector <string> words) {
    int counter = 0;                                            // Hold number of times the word repeats
    for (string s : words) {
        if (s == word) {
            ++counter;
        }
    }
    return counter;
}


// This function checks whether string s consists only of digits or "-" symbol
bool isNum(string s)
{
    for (char ch : s) {
        if (!isdigit(ch) and ch != '-') return false;
    }
    return true;
}
```



```
// This function lets the user to enter an integer, and not something else
int inputNum() {
    string n;
    cout << "Enter number N: ";
    cin >> n;

    while (!isNum(n)) {
        cout << "You can only enter an interger number: ";
        cin >> n;
    }
    return stoi(n);
}


// This function removes all occurrences of word from string vector words
vector <string> removeWord(vector<string> words, string word) {

    vector <string> newWords;                                   // New string vector without the word

    for (string s : words) {
        if (s != word) {
            newWords.push_back(s);
        }
    }

    return newWords;
}
```



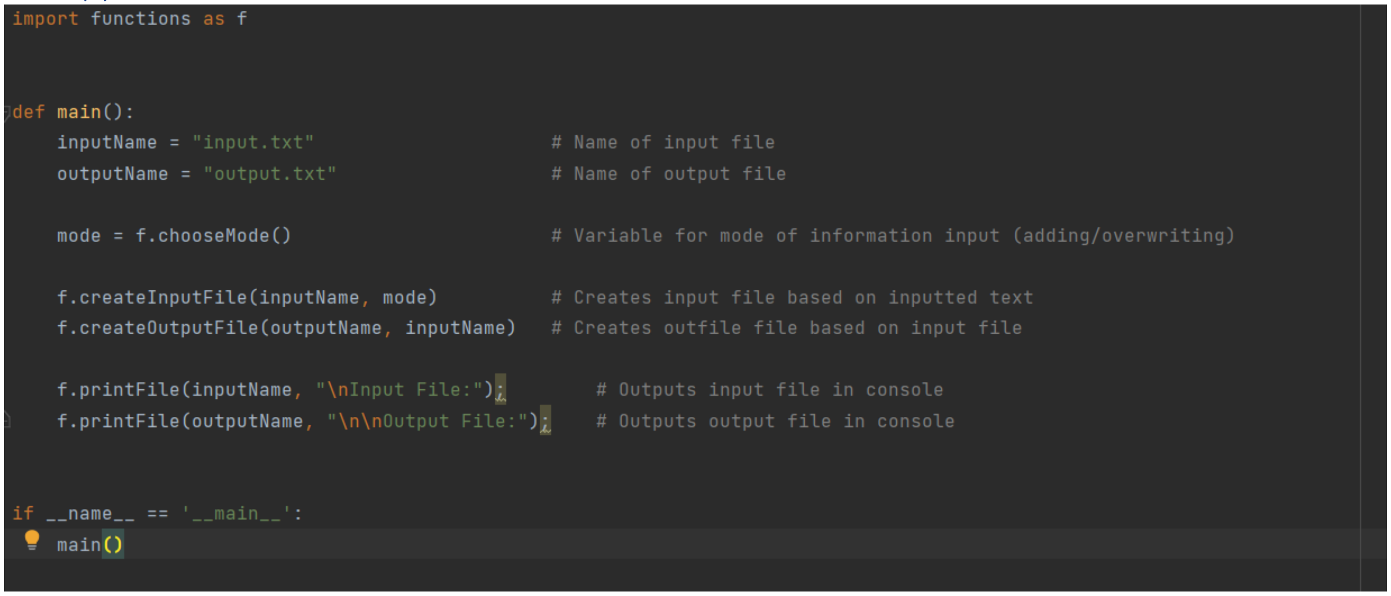
```cpp
// This function creates a string vector, which consists only of words from file that repeat more than n times
vector <string> createRepeatingVector(string name, int n) {

    vector <string> words = getWords(name);                         // String vector that holds all splitted words

    vector <string> updatedWords = words;                           // String vector that holds words that repeat more than n times

    int counter;
    for (string word : words) {
        counter = countWords(word, words);                          // Variable that stores number of times an individual word repeats
        if (counter <= n) {
            updatedWords = removeWord(updatedWords, word);
        }
    }

    return updatedWords;
}


// This function sorts words by length ascendingly in string vector using Odd-Even Bubble Sort
void oddEvenSort(vector <string>& words) {
    bool sorted = false;
    while (!sorted) {
        sorted = true;

        for (int i = 0; i < words.size() - 1; i += 2) {
            if (words[i].length() > words[i + 1].length()) {
                string temp = words[i];
                words[i] = words[i + 1];
                words[i + 1] = temp;
                sorted = false;
            }
        }

        for (int i = 1; i < words.size() - 1; i += 2) {
            if (words[i].length() > words[i + 1].length()) {
                string temp = words[i];
                words[i] = words[i + 1];
                words[i + 1] = temp;
                sorted = false;
            }
        }
    }
}
```



```cpp
// This function creates output file based on input file
void createOutputFile(string outputName, string inputName) {

    int n = inputNum();                                                 // Number n that user inputs
    vector <string> repeatingWords = createRepeatingVector(inputName, n);   // String vector that holds only words that repeat more than n times
    oddEvenSort(repeatingWords);                                        // Sort the string vector

    ofstream outputFile(outputName);

    if (repeatingWords.size() > 0) outputFile << repeatingWords[0];     // First word is written in the file without a space before
    for (int i = 1; i < repeatingWords.size(); i++) {                   // Next words are written with a space before
        outputFile << " " << repeatingWords[i];
    }
    outputFile.close();
}


// This function outputs text from user-selected file in console
void printFile(string name, string text) {

    string line;                                                        // Assisting string
    ifstream inputFile(name);

    cout << text << endl;                                               // Header of the text

    while (!inputFile.eof()) {
        getline(inputFile, line);
        cout << line << endl;
    }

    inputFile.close();
}
```

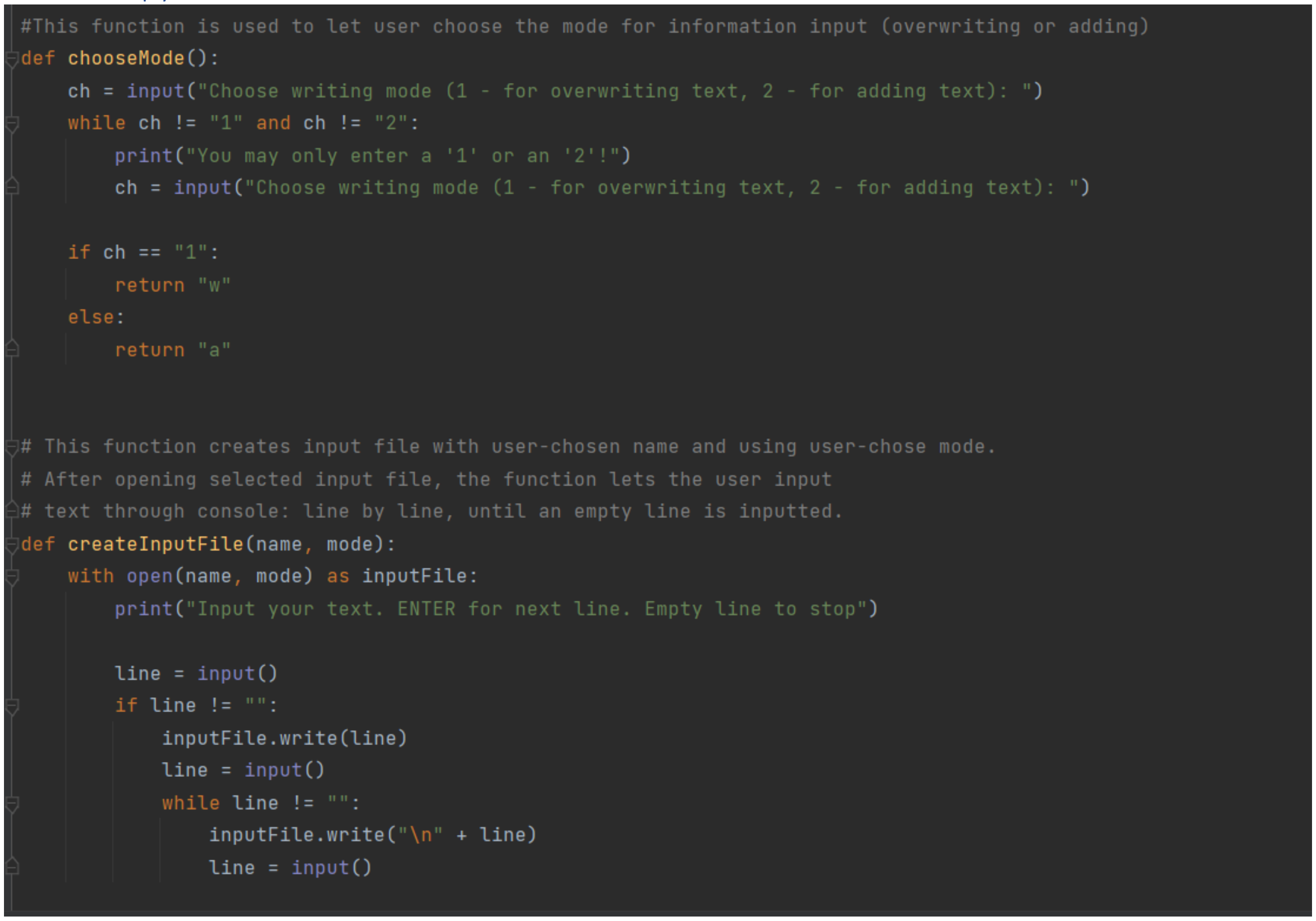
Код Python

main.py

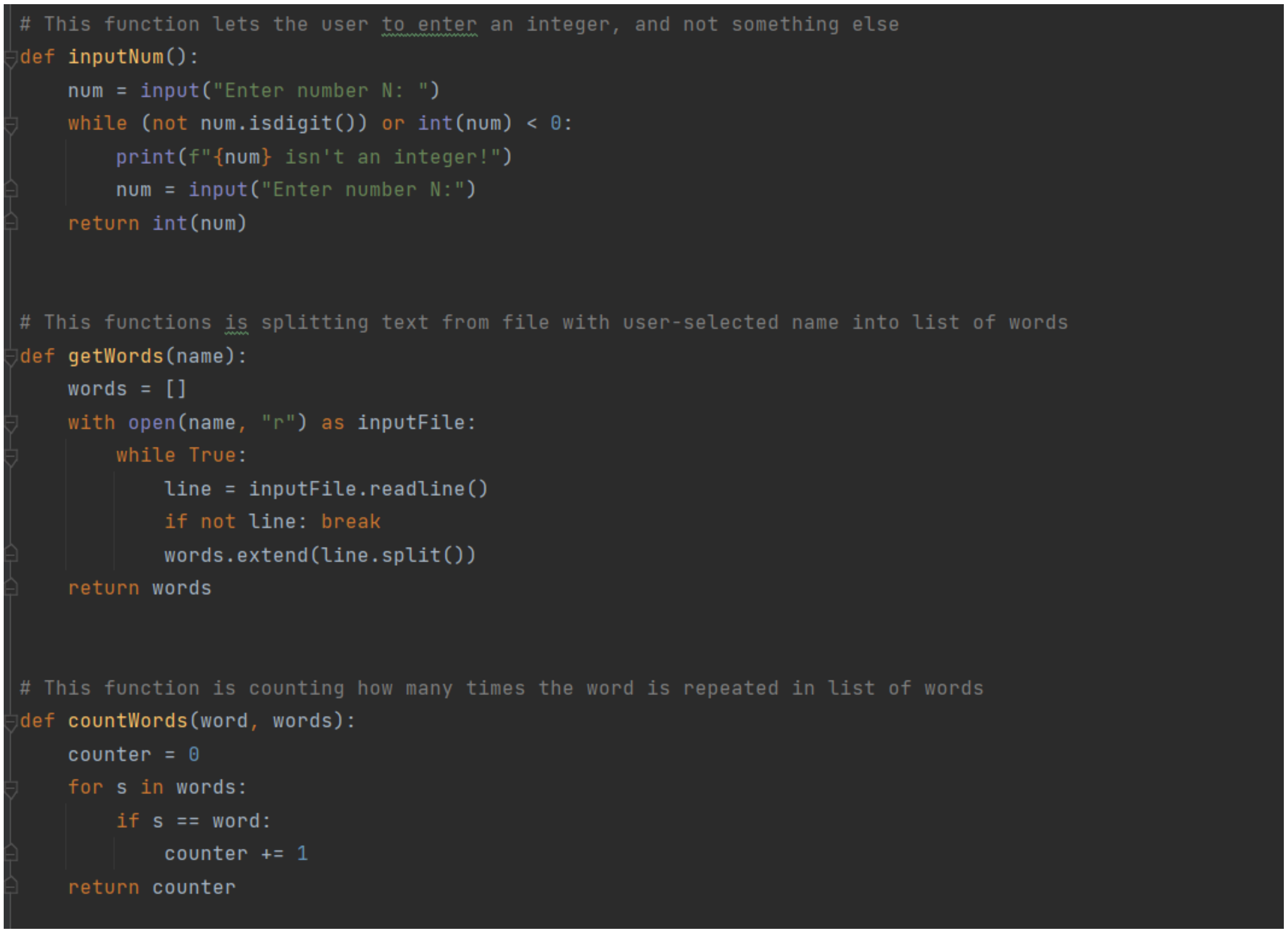


```python
import functions as f


def main():
    inputName = "input.txt"                     # Name of input file
    outputName = "output.txt"                   # Name of output file

    mode = f.chooseMode()                       # Variable for mode of information input (adding/overwriting)

    f.createInputFile(inputName, mode)          # Creates input file based on inputted text
    f.createOutputFile(outputName, inputName)   # Creates outfile file based on input file

    f.printFile(inputName, "\nInput File:");        # Outputs input file in console
    f.printFile(outputName, "\n\nOutput File:");    # Outputs output file in console


if __name__ == '__main__':
    main()
```

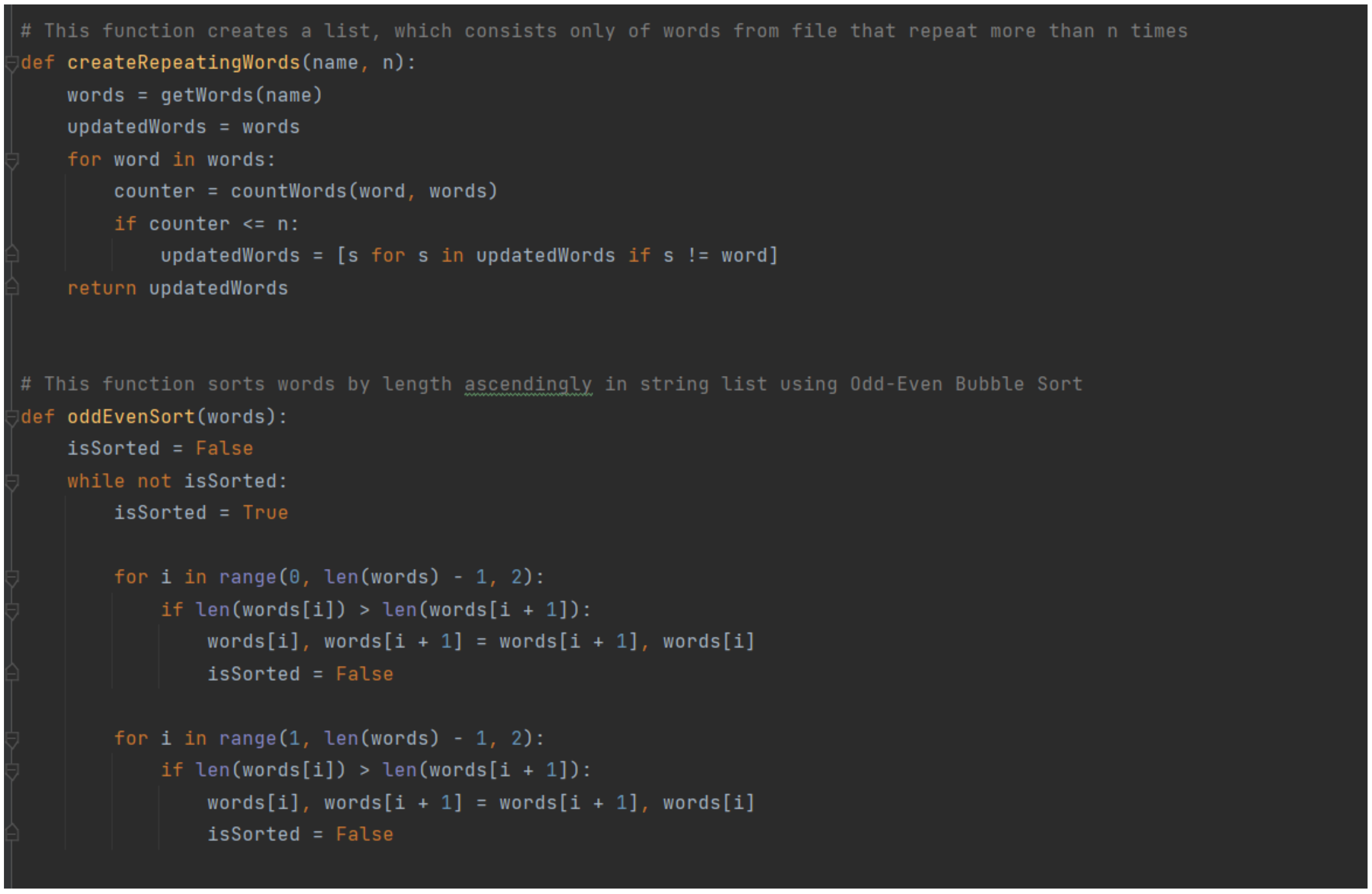
functions.py



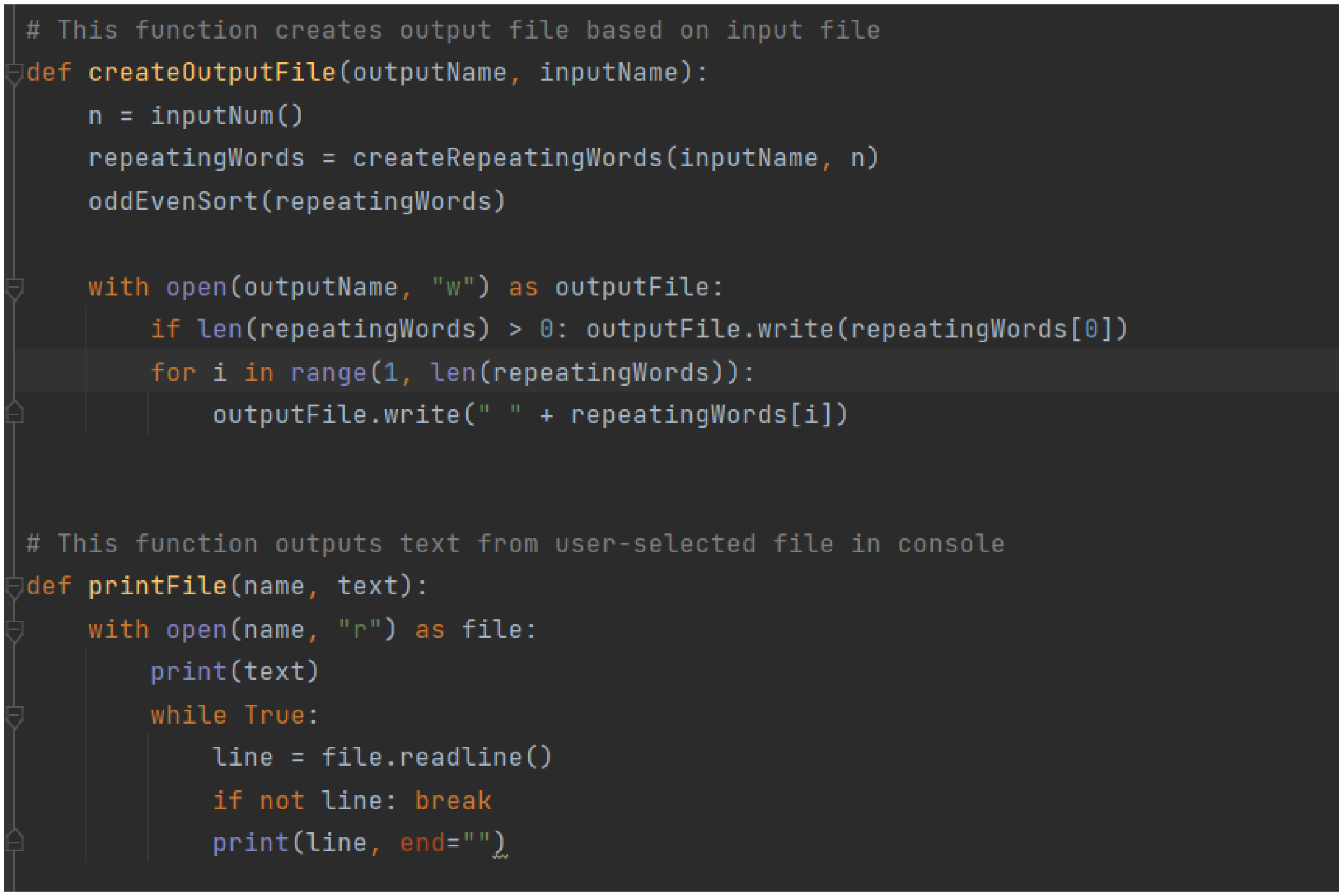
```python
#This function is used to let user choose the mode for information input (overwriting or adding)
def chooseMode():
    ch = input("Choose writing mode (1 - for overwriting text, 2 - for adding text): ")
    while ch != "1" and ch != "2":
        print("You may only enter a '1' or an '2'!")
        ch = input("Choose writing mode (1 - for overwriting text, 2 - for adding text): ")

    if ch == "1":
        return "w"
    else:
        return "a"


# This function creates input file with user-chosen name and using user-chose mode.
# After opening selected input file, the function lets the user input
# text through console: line by line, until an empty line is inputted.
def createInputFile(name, mode):
    with open(name, mode) as inputFile:
        print("Input your text. ENTER for next line. Empty line to stop")

        line = input()
        if line != "":
            inputFile.write(line)
            line = input()
            while line != "":
                inputFile.write("\n" + line)
                line = input()
```



```python
# This function lets the user to enter an integer, and not something else
def inputNum():
    num = input("Enter number N: ")
    while (not num.isdigit()) or int(num) < 0:
        print(f"{num} isn't an integer!")
        num = input("Enter number N:")
    return int(num)


# This functions is splitting text from file with user-selected name into list of words
def getWords(name):
    words = []
    with open(name, "r") as inputFile:
        while True:
            line = inputFile.readline()
            if not line: break
            words.extend(line.split())
    return words


# This function is counting how many times the word is repeated in list of words
def countWords(word, words):
    counter = 0
    for s in words:
        if s == word:
            counter += 1
    return counter
```

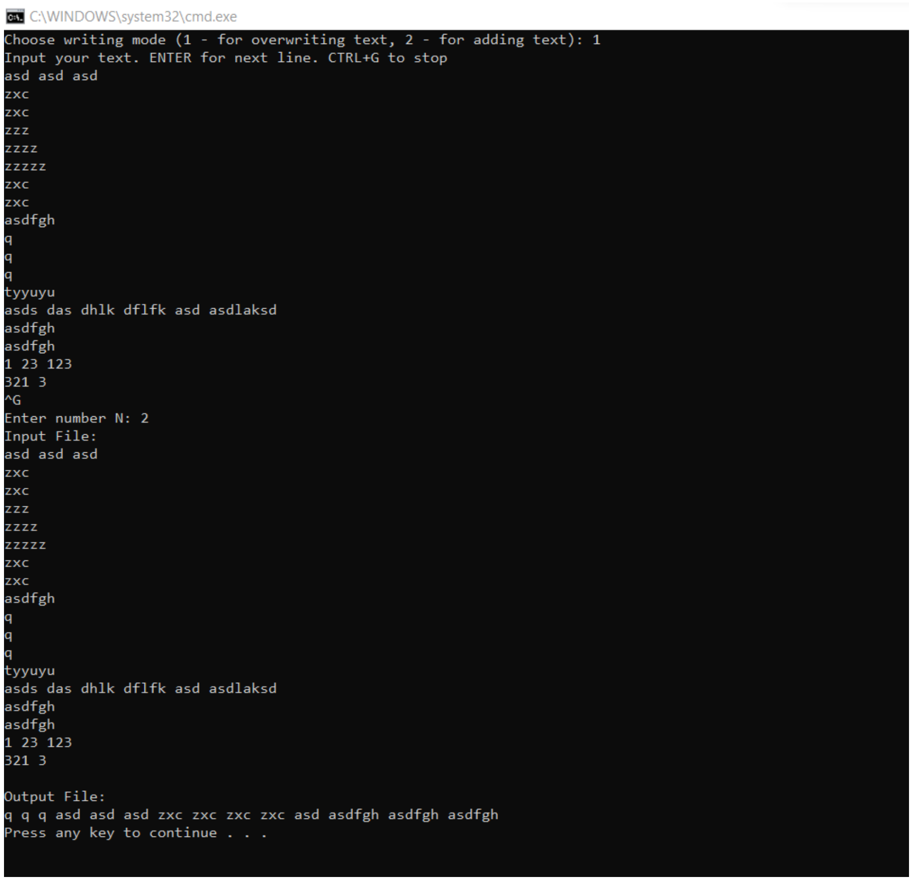


```python
# This function creates a list, which consists only of words from file that repeat more than n times
def createRepeatingWords(name, n):
    words = getWords(name)
    updatedWords = words
    for word in words:
        counter = countWords(word, words)
        if counter <= n:
            updatedWords = [s for s in updatedWords if s != word]
    return updatedWords


# This function sorts words by length ascendingly in string list using Odd-Even Bubble Sort
def oddEvenSort(words):
    isSorted = False
    while not isSorted:
        isSorted = True

        for i in range(0, len(words) - 1, 2):
            if len(words[i]) > len(words[i + 1]):
                words[i], words[i + 1] = words[i + 1], words[i]
                isSorted = False

        for i in range(1, len(words) - 1, 2):
            if len(words[i]) > len(words[i + 1]):
                words[i], words[i + 1] = words[i + 1], words[i]
                isSorted = False
```


```python
# This function creates output file based on input file
def createOutputFile(outputName, inputName):
    n = inputNum()
    repeatingWords = createRepeatingWords(inputName, n)
    oddEvenSort(repeatingWords)

    with open(outputName, "w") as outputFile:
        if len(repeatingWords) > 0: outputFile.write(repeatingWords[0])
        for i in range(1, len(repeatingWords)):
            outputFile.write(" " + repeatingWords[i])


# This function outputs text from user-selected file in console
def printFile(name, text):
    with open(name, "r") as file:
        print(text)
        while True:
            line = file.readline()
            if not line: break
            print(line, end="")
```

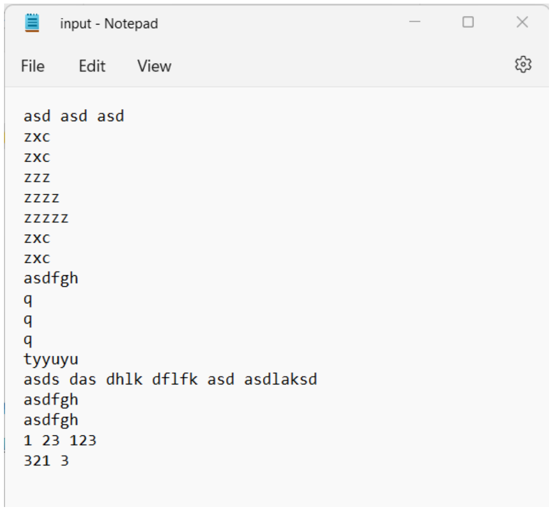
Тестування коду

C++

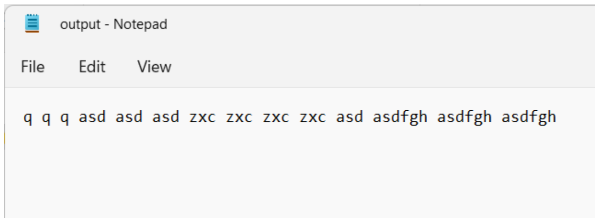


C:\WINDOWS\system32\cmd.exe

```
Choose writing mode (1 - for overwriting text, 2 - for adding text): 1
Input your text. ENTER for next line. CTRL+G to stop
asd asd asd
zxc
zxc
zzz
zzzz
zzzzz
zxc
zxc
asdfgh
q
q
q
tyyuyu
asds das dhlk dflfk asd asdlaksd
asdfgh
asdfgh
1 23 123
321 3
^G
Enter number N: 2
Input File:
asd asd asd
zxc
zxc
zzz
zzzz
zzzzz
zxc
zxc
asdfgh
q
q
q
tyyuyu
asds das dhlk dflfk asd asdlaksd
asdfgh
asdfgh
1 23 123
321 3

Output File:
q q q asd asd asd zxc zxc zxc zxc asd asdfgh asdfgh asdfgh
Press any key to continue . . .
```

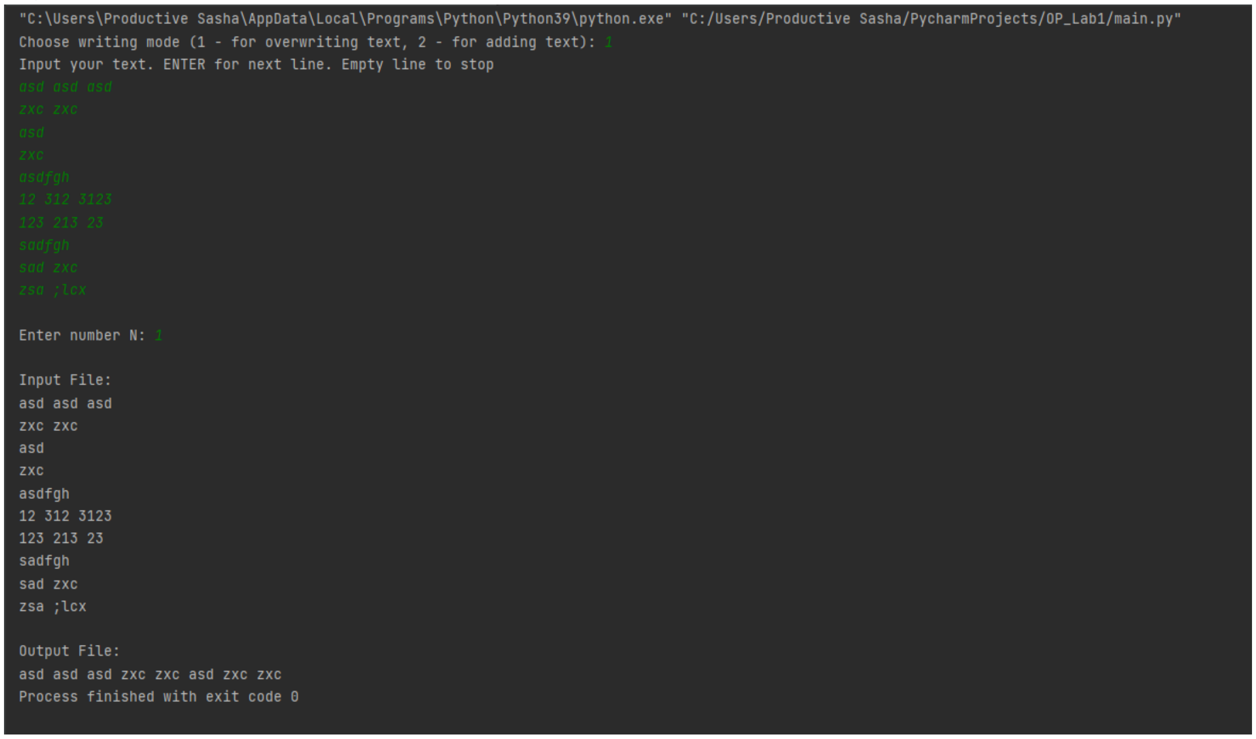


```
asd asd asd
zxc
zxc
zzz
zzzz
zzzzz
zxc
zxc
asdfgh
q
q
q
tyyuyu
asds das dhlk dflfk asd asdlaksd
asdfgh
asdfgh
1 23 123
321 3
```

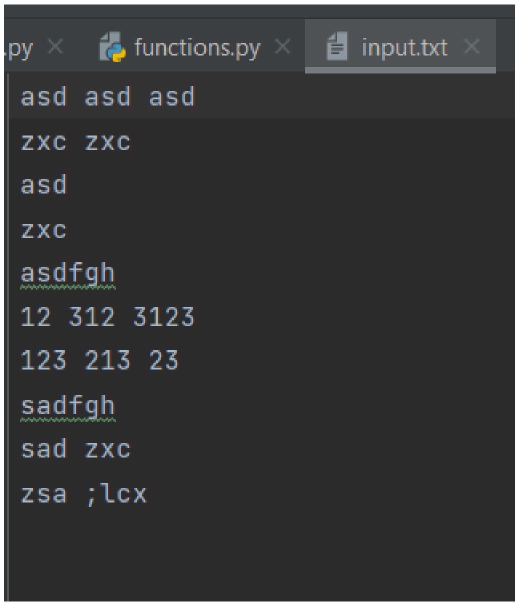


```
q q q asd asd asd zxc zxc zxc zxc asd asdfgh asdfgh asdfgh
```

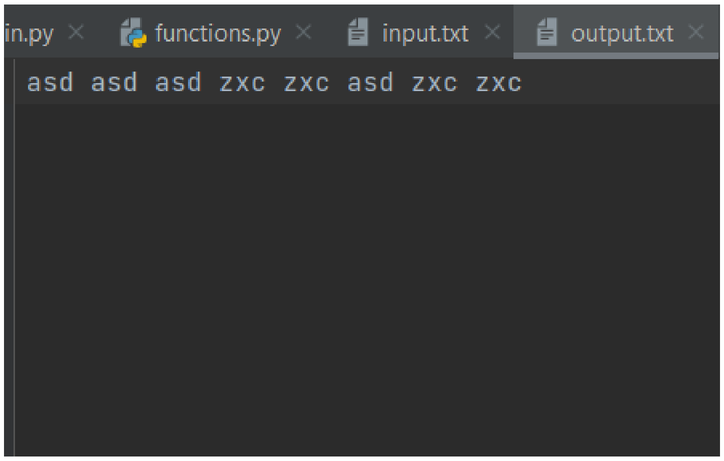
Python



```
"C:\Users\Productive Sasha\AppData\Local\Programs\Python\Python39\python.exe" "C:/Users/Productive Sasha/PycharmProjects/OP_Lab1/main.py"
Choose writing mode (1 - for overwriting text, 2 - for adding text): 1
Input your text. ENTER for next line. Empty line to stop
asd asd asd
zxc zxc
asd
zxc
asdfgh
12 312 3123
123 213 23
sadfgh
sad zxc
zsa ;lcx

Enter number N: 1

Input File:
asd asd asd
zxc zxc
asd
zxc
asdfgh
12 312 3123
123 213 23
sadfgh
sad zxc
zsa ;lcx

Output File:
asd asd asd zxc zxc asd zxc zxc
Process finished with exit code 0
```



```
asd asd asd
zxc zxc
asd
zxc
asdfgh
12 312 3123
123 213 23
sadfgh
sad zxc
zsa ;lcx
```



```
asd asd asd zxc zxc asd zxc zxc
```