**Задание:**

**Вариант: 13**  
Написать программу, которая читая текст из файла, выводит на экран только предложения, состоящие из заданного количества слов.

**Таблица с результатами тестирования программы:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Область тестирования** | **Вариант тестирования** | **Результат тестирования** | **Результат на экране** |
| Наличие файла | Файл есть | Программа работает правильно | Результат работы программы: … Время выполнения: … |
| Файла нет | Программа определяет, что файла по такому пути нет | Файл \*.txt не обнаружен. Время выполнения: … |
| Размер файла | Пустой файл | Программа определяет, что файл пустой | Файл является пустым! Время выполнения: … |
| Средний файл | Программа работает правильно | Результат работы программы: … Время выполнения: … |
| Большой файл | Результат тестирования зависит от объема ОЗУ | Результат работы программы: … Время выполнения: … |
| Путь к файлу | Задан верно | Программа работает правильно | Результат работы программы: … Время выполнения: … |
| Задан не верно | Программа определяет, что файла по такому пути не существует | Файл \*.txt не обнаружен. Время выполнения: … |
| Требуемое количество слов в предложении | Задано корректно | Программа работает правильно | Результат работы программы: … Время выполнения: … |
| Задано не корректно | Программа определяет, что параметр задан не корректно | Вы ввели некорректное число. Время выполнения: … |
| Знаки препинания | Текст содержит знаки препинания, предполагающие конец предложения | Программа работает правильно | Результат работы программы: … Время выполнения: … |
| Текст не содержит знаки препинания, предполагающие конец предложения | Программа проверяет наличие текста, в результате чего делает вывод: выводить ошибку или проверить количество слов в единственном предложении с требуемым | Результат работы программы: … Время выполнения: …  **или**  Файл не содержит знаков препинания, предполагающих конец предложения!  Время выполнения: … |

|  |  |
| --- | --- |
| **Качество работы программы** | |
| **Количество символов** | **Время работы** |
| 1000 символов | 0.0625 seconds |
| 5000 символов | 0.0724 seconds |
| 10000 символов | 0.075 seconds |
| 50000 символов | 0.109 seconds |
| 100000 символов | 0.192 seconds |
| 500000 символов | 0.663 seconds |

**Контрольная оценка работы программы:**

Основываясь на полученные в ходе тестирования данные, сделаем вывод, что программа работает правильно при любом наборе исходных данных (поставленная задача полностью реализована), а также верно анализирует всевозможные исключения. Сложность данной программы линейная - O(n).

**Приложение 1:**

import time  
import re  
  
try:  
 filePath = input('Введите путь к файлу: ') # Получаем путь к файлу.  
 numWords = int(input('Введите требуемое количество слов: ')) # Получаем требуемое количество слов в предложении.  
 with open(filePath, 'r', encoding='utf8') as file: # Безопасно открываем файл.  
 print("Результат работы программы:")  
 separators = ['.', '!', '?'] # Разделители текста на предложения.  
 punctuationFlag = False # Флаг для отслеживания наличия знаков пунктуации в текстовом документе.  
 text = file.read() # Чтение файла.  
 if text == '': # Проверка файла на наличие символов.  
 print('Файл является пустым!')  
 print("Время выполнения: " + str(time.process\_time()))  
 for el in separators:  
 if (text.find(el) != -1):  
 punctuationFlag = True  
 break  
 if (punctuationFlag == False): # Проверка на содержание в тексте знаков препинания.  
 if (len(text.split()) == numWords): # Проверка совпадения количества слов с требуемым количеством.  
 print(text)  
 print("Время выполнения: " + str(time.process\_time()))  
 else: # Если не совпадает, тогда выводим ошибку.  
 print("Файл не содержит знаков препинания, предполагающих конец предложения!")  
 print("Время выполнения: " + str(time.process\_time()))  
 else:  
 sentences = re.split('([.|!|?])', text) # Получение списка предложений и разделителей.  
 sentences2 = [] # Создание отдельного списка.  
 for i in range(1, len(sentences), 2): # Прохождение списка 'sentences' для заполнения 'sentences2'.  
 sentences2.append(sentences[i - 1].strip() + sentences[i].strip()) # Заполнение списка 'sentences2'.  
 for el in sentences2: # Прохождение списка 'sentences' для проверки на совпадение количества слов.  
 if (len(el.split()) == numWords): # Проверка совпадения количества слов с требуемым количеством.  
 print(el)  
 print("Время выполнения: " + str(time.process\_time()))  
# Исключения  
except FileNotFoundError: # Файл не обнаружен.  
 print("Файл \*.txt в директории проекта не обнаружен.")  
 print("Время выполнения: " + str(time.process\_time()))  
except ValueError(): # Параметр 'numWords' некорректен.  
 print("Вы ввели некорректное число.")  
 print("Время выполнения: " + str(time.process\_time()))