|  |  |  |
| --- | --- | --- |
|  |  |  |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |

Институт искусственного интеллекта (ИИИ)

Кафедра промышленной информатики (ПИ)

**ПРАКТИЧЕСКАЯ РАБОТА №1**

по дисциплине «[Технологии и инструментарий анализа больших данных»](https://online-edu.mirea.ru/course/view.php?id=7666)

|  |  |
| --- | --- |
| Выполнил: Чурилов А.К.  Группа: ИВБО-06-20 | \_\_\_\_\_\_\_\_\_\_\_  *Подпись* |
| Проверил: Парамонов А.А. | \_\_\_\_\_\_\_\_\_\_\_  *Подпись* |

|  |  |
| --- | --- |
| Работа представлена к защите | «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2023г. |
| Работа сдана | «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2023г. |

Москва 2023

# Задание 1

Задание:

Дан текст на английском языке. Необходимо закодировать его с помощью азбуки Морзе, где каждой букве соответствует последовательность точек и тире. Например, буква «g» превратится в строку «--.». В переменной morze для удобства хранится словарь соответствия латинских букв коду Морзе.

На входе: В одной строке вам дан текст, который состоит из латинских букв и пробелов.

На выходе: Выведите каждое слово исходного текста, закодированное азбукой Морзе. Количество строк в ответе должно совпадать с количеством слов в исходном тексте. Между закодированными буквами ставится ровно один пробел. Например, слово «Help» превратится в «.... . .-.. .--.». Строчные и заглавные буквы кодируются одинаково.

Практическая часть:

*Листинг 1 – код первого задания*

|  |
| --- |
| def morze():  morze = {'a': '.-', 'b': '-…', 'c': '-.-.', 'd': '-..',  'e': '.', 'f': '..-.', 'g': '--.', 'h': '….',  'i': '..', 'j': '.---', 'k': '-.-', 'l': '.-..',  'm': '--', 'n': '-.', 'o': '---', 'p': '.--.',  'q': '--.-', 'r': '.-.', 's': '…', 't': '-',  'u': '..-', 'v': '…-', 'w': '.--', 'x': '-..-',  'y': '-.--', 'z': '--..'}  print('Введите строку - ')  str = input()  for i in str:  if i == ' ':  print('\n')  else:  print(morze[i], ' ' ,end='') |

Тесты:

На рисунке 1 предоставлен вывод программы из листинга 1.

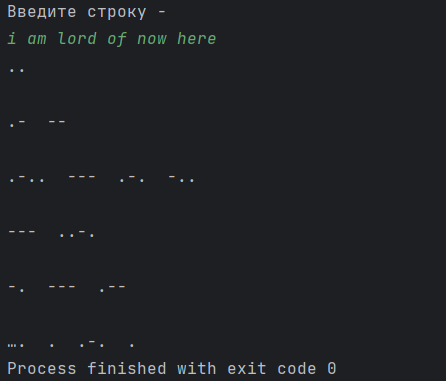


Рис. 1 – выполнение 1-ого задания

# Задание 2

Задание:

В некотором городе открывается новая служба по доставке электронных писем. Необходимо наладить систему регистрации новых пользователей. Регистрация должна работать следующим образом: если новый пользователь хочет зарегистрироваться на сайте, то он должен послать системе запрос name со своим именем. Система должна определить, существует ли уже такое имя в базе данных. Если такого имени не существует, то оно заносится в базу данных системы и пользователю возвращается ответ "ОК", подтверждающий успешную регистрацию. А если пользователь с таким именем уже существует, то система должна сформировать новое имя и выдать его пользователю в качестве подсказки, при этом сама подсказка также добавляется в базу данных. Новое имя формируется следующим образом: к name последовательно приписываются числа, начиная с 1 (name1, name2 и так далее), и среди них находят такое наименьшее i, что namei еще не содержится в системе.

Входные данные: В первой строке входных данных задано число n (1 ≤ n ≤ 100000). Следующие n строк содержат запросы к системе. Каждый запрос представляет собой непустую строку длиной не более 32 символов, состоящую только из строчных букв латинского алфавита.

Выходные данные: В выходных данных должно содержаться n строк – ответы системы на запросы: "OK" в случае успешной регистрации, или подсказка с новым именем, если запрашиваемое уже занято.

Практическая часть

*Листинг 2 – код второго задания*

|  |
| --- |
| def regis():  num = int(input())  a = 1  g = 0  res = []  che = []  for i in range(num):  names = input()  if len(res) == 0:  print('OK')  res.append(names)  a = 1  else:  if names in res:  print(names + a.\_\_str\_\_())  a = a + 1  else:  print('OK')  res.append(names)  a = 1 |

Тесты:

На рисунке 2 предоставлен вывод программы из листинга 2.

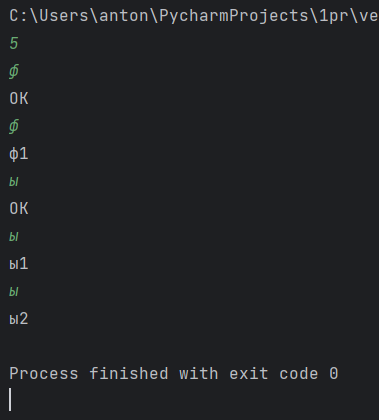


Рис. 2 - выполнение 2-ого задания

# Задание 3

Задание:

Необходимо создать программу обработки запросов пользователей к файловой системе компьютера. Над каждым файлом можно производить следующие действия: запись – w ("write"), чтение – r ("read"), запуск – x ("execute").

Входные данные: на вход программе подаются следующие параметры: число n – количество файлов в файловой системе. В следующих n строках содержится информация с именами файлов и допустимыми действиями (w, x, r), разделенных пробелами. Далее идет число m – количество запросов к файлам вида «операция файл» (обозначение операции: "write", "read", "execute").

Выходные данные: для каждого допустимого запроса программа должна возвращать OK, для недопустимого – Access denied.

Практическая часть:

*Листинг 3 – код третьего задания*

|  |
| --- |
| def check():  a = 0  names = {}  rights = {'w': 'write', 'r': 'read', 'x': 'execute'}  rig = []  num = int(input())  for i in range(num):  x = input().split()  for i in x[1:]:  rig.append(rights[i])  names[x[0]] = rig.copy()  rig.clear()  num = int(input())  for i in range(num):  comm, n = input().split()  if comm in names[n]:  print('OK')  else:  print('Access denied') |

Тесты:

На рисунке 3 предоставлен вывод программы из листинга 3.

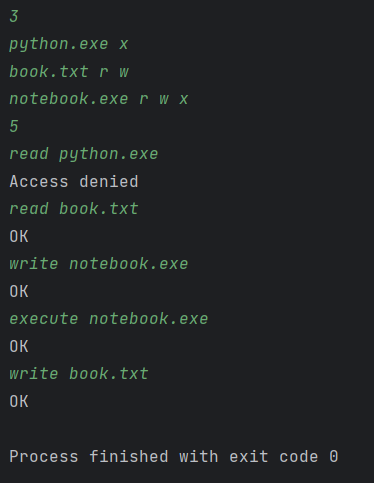


Рис. 3 - выполнение 3-ого задания