**НЕГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ ЧАСТНОЕ** **УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ** **«МОСКОВСКИЙ ФИНАНСОВО-ПРОМЫШЛЕННЫЙ УНИВЕРСИТЕТ** **“СИНЕРГИЯ”»**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Факультет/Институт** | | |  | Информационных технологий | |
|  | | |  | (наименование факультета/ Института) | |
| **Направление/специальность** | |  | 09.02.07 Информационные системы и программирование | | |
| **подготовки:** | | |  | (код и наименование направления /специальности подготовки) | |
| **Форма обучения:** | | |  | Очная | |
|  | | |  | (очная, очно-заочная, заочная) | |
|  | | |  |  | |

**Отчет по лабораторному практикуму №2**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **на тему** |  | **Разработка тестовых пакетов** | | |
|  |  | (наименование темы) | | |
|  |  |  | | |
| **по дисциплине** | | |  | Тестирование информационных систем |
|  | | |  | (наименование дисциплины) |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Обучающийся** |  | Хайдуров Николай Николаевич |  |  |
|  |  | (ФИО) |  | (подпись) |
| **Группа** |  | ДКИП-312 |  |  |
|  |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Преподаватель** |  | **Сибирев Иван Валерьевич** |  |  |
|  |  | (ФИО) |  | (подпись) |

**Москва 2024**

**Лабораторная работа № 2: «Разработка тестовых пакетов»:**

**Цель:** получить навыки разработки тестовых пакетов.

**Ход работы:**

1) Была написана программа, которая шифрует и дешифрует текст по алгоритму квадрата Полибия.

**Код:**

def main():

  """

  Функция main - это точка входа в программу.

  """

  while True:

    # Меню выбора действия

    print("1. Шифрование")

    print("2. Дешифрование")

    print("3. Выход")

    choice = input("Выберите действие: ")

    if choice == "1":

      # Шифрование текста

      text = input("Введите текст для шифрования: ").upper()

      encrypted\_text = encrypt\_text(text)

      print(f"Зашифрованный текст: {encrypted\_text}")

      save\_text\_to\_file(encrypted\_text, "Encrypted.txt")

    elif choice == "2":

      # Дешифрование текста

      print("1. Дешифровать из файла Encrypted.txt")

      print("2. Дешифровать введённый текст")

      decryption\_choice = input("Выберите способ дешифрования: ")

      if decryption\_choice == "1":

        # Дешифрование из файла

        try:

          with open("Encrypted.txt", "r") as f:

            cipher = f.read()

          decrypted\_text = decrypt\_text(cipher)

          print(f"Дешифрованный текст: {decrypted\_text}")

          save\_text\_to\_file(f"Шифр: {cipher}\nДешифрованный текст: {decrypted\_text}", "Decrypted.txt")

        except FileNotFoundError:

          print("Файл Encrypted.txt не найден.")

      elif decryption\_choice == "2":

        # Дешифрование введённого текста

        cipher = input("Введите зашифрованный текст: ")

        decrypted\_text = decrypt\_text(cipher)

        print(f"Дешифрованный текст: {decrypted\_text}")

        save\_text\_to\_file(f"Шифр: {cipher}\nДешифрованный текст: {decrypted\_text}", "Decrypted.txt")

      else:

        print("Неверный выбор. Пожалуйста, выберите 1 или 2.")

    elif choice == "3":

      # Выход из программы

      print("Завершение работы программы.")

      break

    else:

      # Неверный выбор

      print("Неверный выбор. Пожалуйста, выберите из 1, 2 или 3.")

def encrypt\_text(text):

  """

  Функция шифрования текста.

  """

  encrypted\_text = ""

  for char in text:

    if char == " ":

      encrypted\_text += "99"

    elif char == ",":

      encrypted\_text += "89"

    elif char == '"':

      encrypted\_text += "79"

    elif char == ".":

      encrypted\_text += "69"

    else:

      row, col = get\_coordinates(char)

      encrypted\_text += f"{row}{col}"

  return encrypted\_text

def decrypt\_text(cipher):

  """

  Функция дешифрования текста.

  """

  if len(cipher) % 2 != 0:

    print("Ошибка: Шифр имеет нечётную длину. Текст должен быть зашифрован парами цифр.")

    return

  decrypted\_text = ""

  for i in range(0, len(cipher), 2):

    row = int(cipher[i])

    col = int(cipher[i + 1])

    if cipher[i] == '9' and cipher[i + 1] == '9':

      char = " "

    elif cipher[i] == '8' and cipher[i + 1] == '9':

      char = ","

    elif cipher[i] == '7' and cipher[i + 1] == '9':

      char = '"'

    elif cipher[i] == '6' and cipher[i + 1] == '9':

      char = "."

    else:

      char = get\_char(row, col)

    decrypted\_text += char

  return decrypted\_text

def get\_coordinates(char):

  """

  Функция получения координат буквы в квадрате Полибия.

  """

  alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

  index = alphabet.index(char)

  row = index // 5 + 1

  col = index % 5 + 1

  return row, col

def get\_char(row, col):

  """

  Функция получения буквы из квадрата Полибия по координатаm.

  """

  alphabet = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"

  index = (row - 1) \* 5 + col - 1

  if 0 <= index < len(alphabet):

    return alphabet[index]

  else:

    return "?"

def save\_text\_to\_file(text, filename):

  """

  Функция сохранения текста в файл.

  """

  with open(filename, "w") as f:

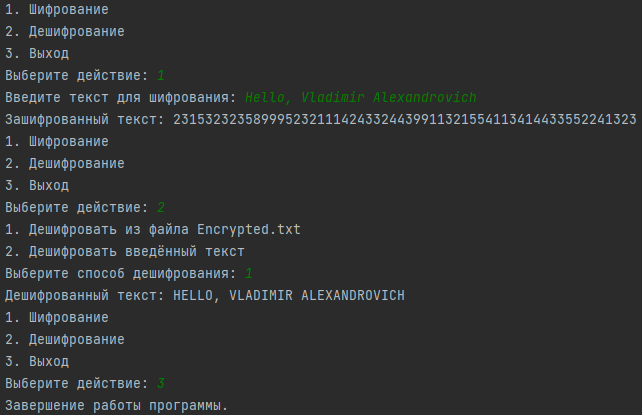
    f.write(text)

if \_\_name\_\_ == "\_\_main\_\_":

  main()

**2) Результаты тестирования:**

|  |  |  |  |
| --- | --- | --- | --- |
| № | Вводные данные | Выводные 1 | Выводные 2 |
| 1 | Hello, Vladimir Alexandrovich | 2315323235899952321114243324439911321554113414433552241323 | HELLO, VLADIMIR ALEXANDROVICH |
| 2 | ",". | 79897969 | ",". |
| 3 | @!? | Error | Error |
| 4 | A program was written that encrypts and decrypts text using the Polybius square algorithm. | 119941433522431133995311449953432445451534994523114599153413435541454499113414991415134355414544994515544599514424342299452315994135325512245144994442511143159911322235432445233369 | A PROGRAM WAS WRITTEN THAT ENCRYPTS AND DECRYPTS TEXT USING THE POLYBIUS SQUARE ALGORITHM. |
| 5 | При ручном вводе шифра: 1155778 | Ошибка: Шифр имеет нечётную длину. Текст должен быть зашифрован парами цифр. | — |
| 6 | При выборе действий:  4 | Неверный выбор. Пожалуйста, выберите из 1, 2 или 3. | — |

* 3-ая срока демонстрирует, что этот код не сможет зашифровать (следовательно, и дешифровать) символы, которые отсутствуют в его алфавите [алфавите кода]. Что можно довольно просто исправить.
* 1-ая, 2-ая и 4-ая строка демонстрируют устойчивость системы (как в шифровке, так и дешифровке) к словам, прописанным через пробел и со знаками пунктуации. Разве что, алгоритм переводит нижний регистр в верхний на выводе дешифрованного текста.
* Скриншот первого теста:

**Вывод:** Методом ручного перебора различных сочетаний символом, были выявлены некоторые ошибки (невосприимчивость к спец. символам, как следствие – незапланированное завершение работы программы, хотя добавить новые символы в запас алфавита программы довольно просто), устойчивость кода к неверным данным (выбор несуществующего действия, или попытка дешифровать нечётный код) и стабильная работа при остальных вводных.