МИНОБРНАУКИ РОССИИ

РГУ НЕФТИ И ГАЗА (НИУ) ИМЕНИ И.М. ГУБКИНА

|  |  |  |
| --- | --- | --- |
| Факультет | **Автоматики и вычислительной техники** |  |
| Кафедра | **Информатики** |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Оценка комиссии: |  | | | Рейтинг: |  |
| Подписи членов комиссии: | | | | | |
|  | |  | Сидоров В.В. | | |
| (подпись) | |  | (фамилия, имя, отчество) | | |
|  | |  |  | | |
| (подпись) | |  | (фамилия, имя, отчество) | | |
|  | | | | | |
| (дата) | | | | | |
|  | |  |  | | |

**КУРСОВАЯ РАБОТА**

|  |  |
| --- | --- |
| по дисциплине | Основы алгоритмизации и программирования |
|  | |

|  |  |
| --- | --- |
| на тему | Разработка программы регистрации и аутентификации |
| пользователей | |
|  | |

|  |  |  |  |
| --- | --- | --- | --- |
| «К ЗАЩИТЕ» |  | ВЫПОЛНИЛ: |  |
|  |  | Студент группы | **АА-24-10** |
|  |  |  | (номер группы) |
|  |  | Колесников С.И. | |
| (должность, ученая степень; фамилия, и.о.) |  | (фамилия, имя, отчество) | |
|  |  |  | |
| (подпись) |  | (подпись) | |
|  |  |  | |
| (дата) |  | (дата) | |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Москва, 20 | 25 |  |

# 1. Постановка задачи

Задача состояла в разработке программы регистрации пользователя. Необходимо запросить информацию: Ф.И.О., дата рождения, пол, город, email, логин, пароль, подтверждение пароля. Проверить корректность введенных данных. Затем запомнить их в связном списке, при этом пароль должен быть закодирован с помощью битовой операции XOR (исключающее ИЛИ) (восстановление производится также с помощью этой операции). Реализовать возможность запроса логина и пароля пользователя и при корректном вводе, вывести сообщение о входе в личный кабинет. Список пользователей должен сохраняться в файле и загружаться из файла при запуске программы Программа должна предоставлять возможность запроса логина и пароля пользователя. При корректном вводе должно выводиться сообщение об успешном входе в личный кабинет, где отображается информация о пользователе. Список пользователей должен сохраняться в файле и загружаться из файла при запуске программы.

Графический интерфейс пользователя (GUI) реализован с использованием библиотеки Tkinter.

# 2. Алгоритм работы программы

Программа имеет графический интерфейс и управляется событиями (нажатия кнопок, ввод данных). Основные этапы работы:

1. **Инициализация программы:**
   * Загрузка списка зарегистрированных пользователей из файла users.data. Если файл отсутствует, он будет создан при первой регистрации.
   * Создание главного окна приложения и фреймов для различных экранов (вход, регистрация, информация о пользователе).
   * Отображение начального экрана – формы входа.
2. **Основной цикл (управляется Tkinter mainloop()):**
   * Программа ожидает действий пользователя.
3. **Процесс регистрации нового пользователя:**
   * Пользователь переходит на экран регистрации.
   * Пользователь вводит данные: Фамилия, Имя, Отчество, Дата рождения, Пол, Город, Email, Логин, Пароль, Подтверждение пароля.
   * **Валидация данных:**
     + Проверка на заполненность обязательных полей.
     + Проверка на совпадение пароля и его подтверждения.
     + Проверка на уникальность введенного логина среди уже зарегистрированных пользователей.
   * **Кодирование пароля:** Введенный пароль кодируется с помощью операции XOR с использованием предопределенного ключа.
   * **Сохранение пользователя:**
     + Данные нового пользователя (включая закодированный пароль) добавляются в общий список пользователей в памяти.
     + Обновленный список пользователей сохраняется в файл users.data, перезаписывая его.
   * Вывод сообщения об успешной регистрации.
   * Автоматический переход на экран входа.
4. **Процесс входа пользователя (аутентификация):**
   * Пользователь вводит логин и пароль на экране входа.
   * **Поиск пользователя:** Программа ищет пользователя с введенным логином в загруженном списке.
   * **Проверка пароля:**
     + Если пользователь найден, его сохраненный закодированный пароль извлекается.
     + К сохраненному закодированному паролю повторно применяется операция XOR с тем же ключом (для "дешифрования").
     + "Дешифрованный" пароль сравнивается с паролем, введенным пользователем.
   * **Результат входа:**
     + **Успешный вход:** Если пароли совпадают, выводится приветственное сообщение. Затем отображается экран с личной информацией пользователя (ФИО, дата рождения, город и т.д.).
     + **Неудачный вход:** Если пользователь не найден или пароли не совпадают, выводится сообщение об ошибке.
5. **Отображение информации о пользователе:**
   * После успешного входа открывается экран "Личный кабинет".
   * На экране отображаются данные вошедшего пользователя: ФИО, дата рождения, пол, город, email, логин.
   * Доступна кнопка "Выйти" для возврата на экран входа.
6. **Сохранение данных:**
   * Данные пользователей (список словарей) сохраняются в текстовый файл users.data в формате CSV (значения, разделенные точкой с запятой). Каждая строка файла представляет одного пользователя.
   * Пароль сохраняется в закодированном виде.
7. **Выход из программы:**
   * Закрытие главного окна приложения завершает выполнение программы.

# 3. Принцип работы XOR-шифрования

XOR (исключающее ИЛИ) — это битовая операция, которая является основой простого симметричного алгоритма шифрования.

**Принцип действия:**

1. **Представление данных:** Исходный текст (пароль) и ключ шифрования представляются в виде последовательностей битов. На практике это обычно означает работу с числовыми кодами символов (например, ASCII или Unicode).
2. **Операция XOR:** Для каждого символа исходного текста выполняется операция XOR с соответствующим символом ключа. Если ключ короче текста, он циклически повторяется.

Таблица истинности XOR (A XOR B = C):

|  |  |  |
| --- | --- | --- |
| A | B | A XOR B = C |
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Иная запись: A ^ B = (!A and B) or (A or !B)

Результаты применения XOR:

* A ^ A = 0
* A ^ 0 = A
* (A ^ B) ^ C = A ^ (B ^ C)

1. **Результат:** Полученная последовательность битов (или символов, соответствующих новым кодам) является зашифрованным текстом.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Text | A1 | A2 | A3 | A4 |
| ^ | ↓ | ↓ | ↓ | ↓ |
| Key | B1 | B2 | B3 | B4 |
| = | ↓ | ↓ | ↓ | ↓ |
| Result | C1 | C2 | C3 | C4 |
| ^ | ↓ | ↓ | ↓ | ↓ |
| Key | B1 | B2 | B3 | B4 |
| = | ↓ | ↓ | ↓ | ↓ |
| Result | A1 | A2 | A3 | A4 |

Размер ключа обычно фиксирован, а также его размер часто меньше размера текста для шифрования. Разберём на примере. Предположим, у нас имеется текст для шифрования длиной в 4 байта. Индексы i этого массива меняются от 0 до 3. Ключ для шифрования имеет длину 2 байта. Таким образом к первым двум байтам текста применяется исключающее ИЛИ с первыми двумя байтами ключа. К третьему байту текста нужно применять XOR уже первый байт ключа и т.д. Определяется порядок по формуле: остаток от деления индекса на длину ключа.

i % len(key)

0 % 2 = 0

1 % 2 = 1

2 % 2 = 0

3 % 2 = 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| i | 0 | 1 | 2 | 3 |
| Text | A1 | A2 | A3 | A4 |
| ^ | ↓ | ↓ |  |  |
| Key | B1 | B2 |  |  |
| = | ↓ | ↓ |  |  |
| Result | C1 | C2 | C3 | C4 |

1. **Дешифрование:** Уникальным свойством операции XOR является ее обратимость при повторном применении с тем же ключом:
   * C XOR B = (A XOR B) XOR B = A  
     Таким образом, для дешифрования зашифрованного текста достаточно снова применить к нему операцию XOR с тем же самым ключом, который использовался для шифрования.

**В данной программе:**

* Функция xor\_cipher(text, key) реализует этот принцип.
* Она берет каждый символ входной строки text и символ из строки key.
* Получает их числовые представления (коды Unicode) с помощью функции ord().
* Применяет операцию XOR (^ в Python) к этим кодам.
* Преобразует полученный код обратно в символ с помощью функции chr().
* Все полученные символы объединяются в итоговую строку (зашифрованную или расшифрованную).

**Важно:**

* XOR-шифрование является **симметричным**, т.е. один и тот же ключ используется как для шифрования, так и для дешифрования.
* Это очень **простой** алгоритм, но он **не является криптостойким** для защиты важной информации в реальных системах. Он уязвим для различных атак, особенно если ключ короткий или используется повторно. В данной программе он используется для демонстрации принципа и выполнения учебной задачи.
* Результатом операции XOR над кодами символов могут быть коды непечатных или управляющих символов. При сохранении в текстовый файл или выводе на экран это может приводить к некорректному отображению (пустые знаки или "красные значки").