Министерство образования Республики Беларусь

Учреждение образования

Белорусский государственный университет информатики и

Радиоэлектроники

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

Отчет по лабораторной работе №1

по курсу “Средства и методы защиты информации в интеллектуальных системах”

Вариант 2

Выполнил:

Студент гр. 221701 Телица И. Д.

Проверил: Захаров В. В.

Минск

2024

**Тема:** Генерация паролей

**Задание для самостоятельного выполнения:**

1. Разработать программу на языке C++, реализующую следующие функции:

- генерация строки с заданной пользователем длиной, состоящей из символов алфавита в соответствии с вариантом задания (использовать функции rand( ), srand( ) и инициализацию от таймера);

- поверка равномерности распределения символов путем визуализации частотного распределения;

- вычисление среднего времени подбора пароля, выбираемого из сгенерированной строки.

- Использовать строчные и заглавные буквы латинского алфавита

2. Построить график зависимости среднего времени подбора пароля от его длины.

3. Дать практические рекомендации по выбору пароля исходя из предположений об алфавите пароля; ценности информации, доступ к которой защищается с помощью этого пароля; производительности вычислительного средства атакующего и времени атаки.

**Выполнение:**

**Основные функции программы:**

**1. char get\_char(int number)**

**Назначение:** Преобразует числовое значение в символ английского алфавита (как заглавную, так и строчную букву).

**Параметры:**

* number (int): Целое число, представляющее индекс в алфавите.

**Возвращаемое значение:**

* char: Символ, соответствующий данному числу.

**Описание:**

* Если number меньше 26 (количество букв в английском алфавите), функция возвращает заглавную букву (от 'A' до 'Z').
* Если number больше или равен 26, функция возвращает строчную букву (от 'a' до 'z').

### **2. int get\_char\_index(char c)**

**Назначение:** Определяет числовой индекс символа в алфавите.

**Параметры:**

* c (char): Символ, индекс которого нужно определить.

**Возвращаемое значение:**

* int: Числовой индекс символа.

**Описание:**

* Если символ находится в диапазоне заглавных букв ('A' до 'Z'), функция возвращает его индекс в диапазоне от 0 до 25.
* Если символ находится в диапазоне строчных букв ('a' до 'z'), функция возвращает его индекс в диапазоне от 26 до 51 (сдвиг по сравнению с заглавными буквами).

### **3. void generate\_password(char\* password, int\* frequencies, int length)**

**Назначение:** Генерирует случайный пароль и обновляет частоты символов в массиве.

**Параметры:**

* password (char\*): Массив символов, куда будет записан сгенерированный пароль.
* frequencies (int\*): Массив, который будет содержать частоты появления символов.
* length (int): Длина пароля.

**Возвращаемое значение:** Нет.

**Описание:**

* Генерирует случайные символы для пароля, используя std::rand(), и записывает их в массив password.
* Обновляет массив frequencies, учитывая, сколько раз каждый символ появляется в пароле.

### **4. void increase\_string(char\* string, int length, int index = 0)**

**Назначение:** Увеличивает строку в "лексикографическом" порядке, аналогично инкременту числа.

**Параметры:**

* string (char\*): Массив символов, который будет изменяться.
* length (int): Длина строки.
* index (int, по умолчанию 0): Текущий индекс символа для увеличения.

**Возвращаемое значение:** Нет.

**Описание:**

* Функция рекурсивно увеличивает символ в строке. Если символ в текущем индексе равен 'z', он заменяется на 'A', и увеличение происходит на следующем индексе.
* Если символ не равен 'z', он заменяется на следующий символ в алфавите.

### **5. char\* find\_password(const char\* target\_password, int length)**

**Назначение:** Ищет пароль, совпадающий с заданным target\_password, перебирая все возможные комбинации.

**Параметры:**

* target\_password (const char\*): Пароль, который нужно найти.
* length (int): Длина пароля.

**Возвращаемое значение:**

* char\*: Найденный пароль (или nullptr, если не найден).

**Описание:**

* Инициализирует строку found\_password как строку, состоящую только из символов 'A'.
* Перебирает все возможные комбинации, увеличивая строку с помощью increase\_string, пока не найдет совпадение с target\_password.

**Результат программы:**

**Задание 1.1:**

****

Рисунок 1. Генерация пароля

**Задание 1.2:**

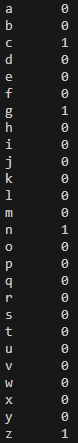
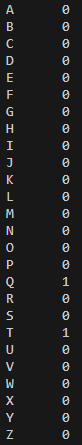
****

Рисунок 2. Вывод частотного распределения символов

**Задание 1.3:**

****

Рисунок 3. Подбор пароля и определение времени подбора

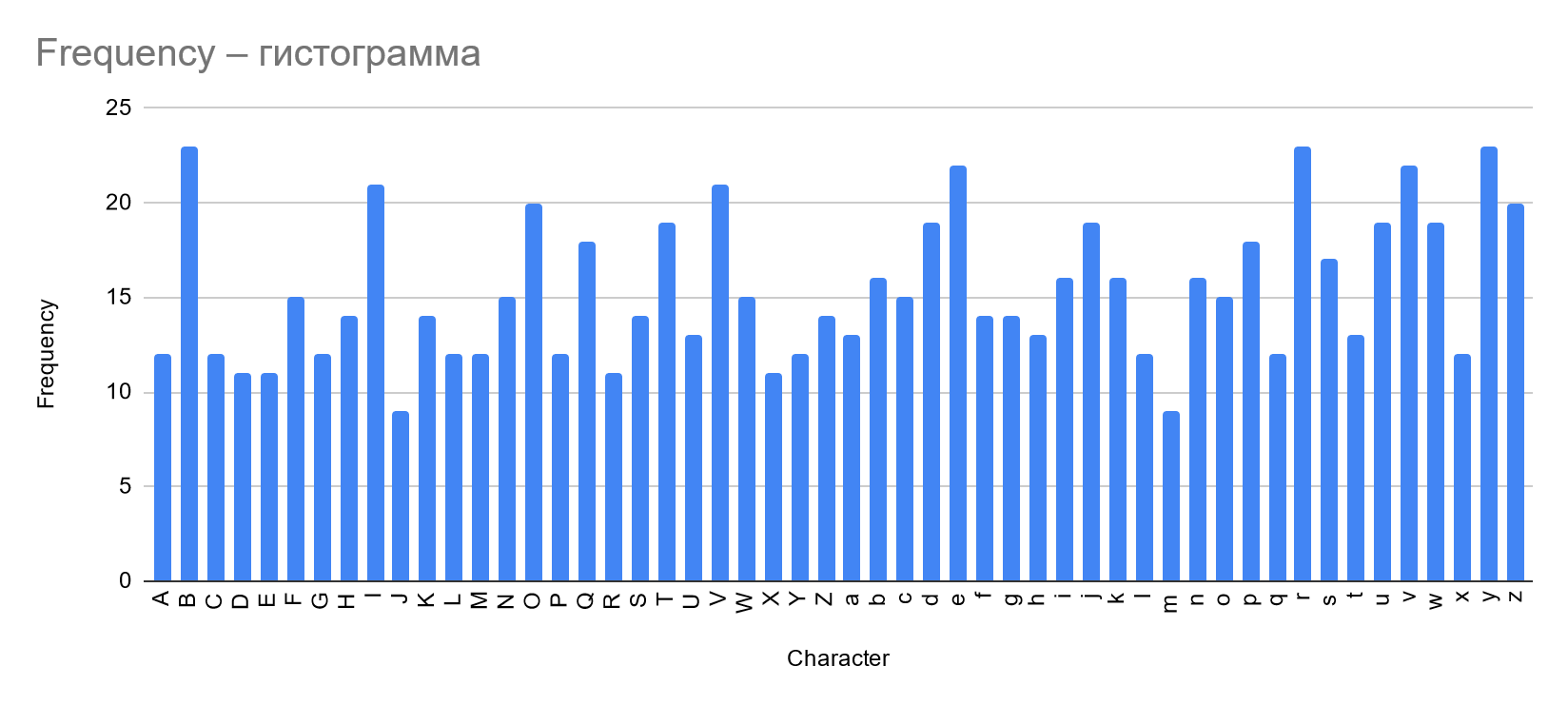
****

Рисунок 4. График частотного распределения символов в пароле длиной 200 символов

**Задание 2:**

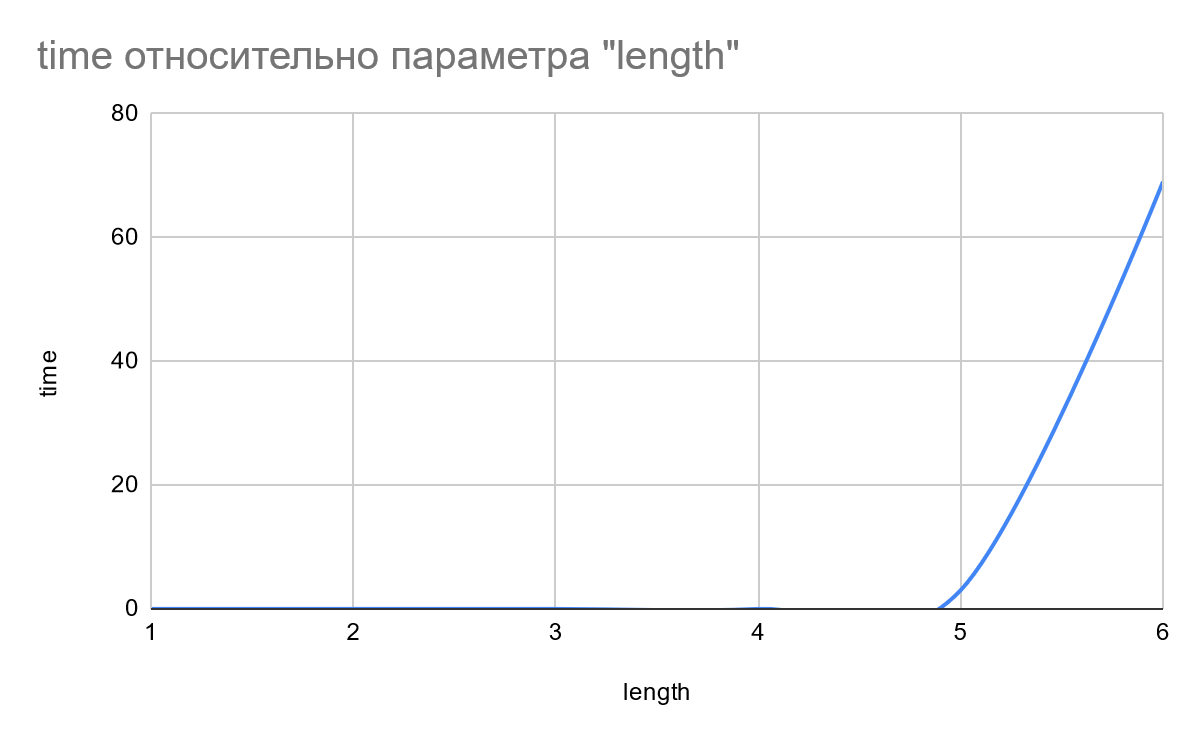
****

Рисунок 5. График среднего времени подбора пароля

**Задание 3:**

Если ваш пароль может включать только строчные и прописные буквы латинского алфавита (всего 52 символа), его безопасность значительно возрастает по сравнению с паролем, состоящим только из строчных букв.

Советы по рекомендации пароля:

1. **Увеличение длины пароля**:
   * **Пароль длиной 8 символов**:
     + С учетом 52 возможных символов, число возможных комбинаций составит 52^8, что примерно равно 53 триллионам комбинаций.
   * **Пароль длиной 12 символов**:
     + Для пароля из 12 символов, число возможных комбинаций будет 522^12, что составляет около 3.13 триллиона триллионов комбинаций.
   * **Рекомендуется использовать пароли длиной не менее 12 символов** для обеспечения надежной защиты. Это особенно важно при использовании латинских букв, чтобы повысить стойкость пароля.
   * **Если ваша информация особенно ценна**, рекомендуется использовать пароли длиной 16 символов или более, чтобы значительно повысить безопасность.
2. **Выбор начальных и конечных символов**:
   * **Выбирайте начальные и конечные символы пароля из середины диапазона доступных символов**. Это снизит вероятность подбора пароля путем перебора символов от начала или конца алфавита, что повысит защиту вашего пароля.
3. **Использование генераторов паролей**:
   * **Используйте генераторы случайных паролей** для создания сложных и непредсказуемых комбинаций. Это поможет создать надежные пароли без необходимости их ручного создания.
   * **Избегайте использования личных данных** (таких как даты рождения, имена или простые слова), которые могут быть легко угаданы или взломаны.
4. **Использование дополнительных символов**:
   * **Для дополнительной безопасности добавляйте цифры и специальные символы** (например, !, @, #, $, %, &, \*) в ваш пароль. Это значительно увеличит количество возможных комбинаций и усложнит задачу злоумышленникам.

**Вывод:**

В ходе выполнения лабораторной работы была разработана программа на языке программирования C++, которая позволяет генерировать случайные пароли заданной длины, состоящие из строчных и прописных латинских букв. Программа также отображает частотное распределение символов в пароле и может подбирать пароль, определяя время, необходимое для подбора пароля методом прямого перебора.

Тестирование программы показывает, что среднее время подбора пароля увеличивается с ростом его длины. Теоретически время подбора пароля можно рассчитать по следующей формуле:

где:

* T — время подбора пароля;
* R — скорость подбора (количество проверяемых паролей в секунду);
* A — размер алфавита (количество возможных символов, которые могут быть использованы в пароле);
* L — длина пароля.

В этой лабораторной работе также изучены важные концепции, такие как субъекты и объекты доступа, а также идентификация и аутентификация.

**Идентификация** выполняет следующие функции:

* **Установление подлинности** и определение полномочий субъекта при входе в систему;
* **Контроль полномочий** в процессе сеанса работы;
* **Регистрация и учет действий** субъекта.

**Аутентификация** включает три основных метода:

1. **Аутентификация с использованием паролей**: наиболее распространенный метод, который может использовать одноразовые и многоразовые пароли. Многоразовые пароли часто используются в повседневной практике, в то время как одноразовые пароли (например, в банковских системах) применяются для повышения безопасности.
2. **Аутентификация с использованием уникальных предметов**: метод, при котором субъекты предъявляют различные предметы (например, ключи или файлы данных) для подтверждения своей личности.
3. **Аутентификация с использованием биометрических данных**: способ, при котором для аутентификации используются уникальные биометрические характеристики субъекта, такие как отпечатки пальцев, рисунок радужной оболочки глаз или лицо.

Тем не менее, ни один из этих методов не обеспечивает абсолютную безопасность. Аутентификация с использованием паролей может быть уязвима к атакам, если пароли перехвачены или угаданы. Аутентификация с использованием уникальных предметов подвержена риску потери или кражи предметов. Аутентификация на основе биометрических данных имеет свои ограничения, такие как ошибки распознавания или возможность подделки.