Частное учреждение образования

«Колледж бизнеса и права»

|  |  |
| --- | --- |
|  | УТВЕРЖДАЮ  Ведущий методист колледжа  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Е.В. Паскал  « \_\_\_\_ » \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 года |
| Специальность 2-40 01 01 «Программное обеспечение информационных технологий» | Учебная дисциплина «Основы алгоритмизации и программирование» |

**Лабораторная работа № 32**

**Инструкционно-технологическая карта**

Тема: Реализация простейших функций хеширования. Разрешение конфликтов.

Цель: Научиться разрабатывать алгоритмы и программы с использованием функций хеширования и разрешать конфликты при совпадении хеш-значений.

Время выполнения: 2 часа.

1. **Порядок выполнения работы**
2. Изучить теоретические сведения к лабораторной работе.
3. Разработать на языке С++ программу вывода на экран решения задачи в соответствии с вариантом индивидуального задания, указанным преподавателем.
4. Отлаженную, работающую программу сдать преподавателю. Работу программы показать с помощью самостоятельно разработанных тестов.
5. Ответить на контрольные вопросы.
6. **Теоретические сведения**

**Хеш-функция**

**Хеш-функция** – функция, преобразовывающая входную последовательность данных произвольного размера в выходную последовательность фиксированного размера. Процесс преобразования данных называется хешированием. Результат хеширования – **хеш-код** (хеш-значение, хеш-сумма, хеш).

При решении класса практических задач выбирается такая хеш-функция, которая является наиболее оптимальной именно для данного класса. В общем случае следует использовать «хорошую» функцию. Хеш-функцию называют «хорошей», если она:

1. вычисляется достаточно быстро;

2. сводит к минимуму число коллизий. **Коллизия** (конфликт) – это совпадающие посчитанные хеш-значения для различающихся данных, то есть для значений «эволюция» и «революция» хеш-функция выдала одинаковые хеш-суммы, например, 15 и 15, хотя сами данные не совпадают полностью.

Предотвратить коллизии могут далеко не все хеш-функции, но «хорошие» способны минимизировать вероятность их появления. При определенных обстоятельствах (известна некоторая информация о ключах), можно найти «идеальную» хеш-функцию, т.е. такую, которая полностью исключает возможность появления коллизий.

Использовать хеш-функцию или нет, зависит от того, насколько целесообразно применение ее свойств, а также свойств алгоритма, по которому она может быть реализована на ЯП. В одних ситуациях наиболее важна высокая скорость работы, в других равномерное распределение хеш-кодов и т.п. Рассмотрим два наиболее известных метода хеширования.

**Метод деления**: пусть k – ключ (значение, которое необходимо хешировать), а N– максимально возможное число хеш-кодов. Тогда метод хеширования посредством деления будет заключаться во взятии остатка от целочисленного деления k на N: h(k)=k mod N, где mod – операция взятия остатка от деления. Например, на вход подаются следующие ключи: 3, 6, 7, 15, 32, 43, 99, 100, 133, 158.

Определим N равным 10, из чего последует, что возможные значения хешей лежат в диапазоне 0…9. Используя данную функцию, получим следующие значения хеш-кодов:

h(3)=3, h(6)=6, h(7)=7, h(15)= 5, h(32)=2, h(43)=3, h(99)=9, h(100)=0, h(133)=3, h(158)=8. Поскольку на разные значения 3, 43, 133 вычисляется одинаковый хеш-код 3, то коллизии у такой простой хеш-функции есть.

На C++ программу, выполняющую хеширование методом деления можно записать так:



Во избежание большого числа коллизий рекомендуется выбирать N простым числом, и не рекомендуется степенью с основанием 2 и показателем m (2m). Вообще, по возможности, следует выбирать N, опираясь на значения входящих ключей. Так, например, если все или большинство k=10m (m – натуральное число), то неудачным выбором будет N=10\*m и N=10m.

**Метод умножения**: получить из исходной последовательности ключей последовательность хеш-кодов, используя метод умножения (мультипликативный метод), то есть воспользоваться хеш-функцией:

h(k)=⌊N\*({k\*A})⌋

Здесь A – рациональное число, по модулю меньшее единицы (0<A<1), а k и N обозначают то же, что и в предыдущем методе: ключ и размер хеш-таблицы. Также правая часть функции содержит три пары скобок:

( ) – скобки приоритета;

⌊ ⌋ – скобки взятия целой части;

{ } – скобки взятия дробной части.

Аргумент хеш-функции k (k≥0) в результате даст значение хеш-кода h(k)=x, лежащие в диапазоне 0…N-1. Для работы с отрицательными числами можно число x взять по модулю. От выбора A и N зависит то, насколько оптимальным окажется хеширование умножением на определенной последовательности. Не имея сведений о входящих ключах, в качестве N следует выбрать одну из степеней двойки, т. к. умножение на 2m равносильно сдвигу на m разрядов, что компьютером производиться быстрее. Неплохим значением для A (в общем случае) будет (√5-1)/2≈0,6180339887. Оно основано на свойствах золотого сечения: **Золотое сечение** – такое деление величины на две части, при котором отношение большей части к меньшей равно отношению всей величины к ее большей части.

Отношение большей части к меньшей, выраженное квадратичной иррациональностью:

**φ=(√5+1)/2≈1,6180339887**

Для мультипликативной хеш-функции было приведено обратное отношение:

**1/φ=(√5-1)/2≈0,6180339887**

При таком A, хеш-коды распределятся достаточно равномерно, но многое зависит от начальных значений ключей.

Для демонстрации работы мультипликативного метода, возьмём N=13, A=0,618033. В качестве ключей возьмем числа: 25, 44 и 97. Подставим их в функцию:

1. h(k)=⌊13\*({25\*0,618033})⌋=⌊13\*{15,450825}⌋=⌊13\*0,450825⌋=⌊5,860725⌋=5.
2. h(k)=⌊13\*({44\*0,618033})⌋=⌊13\*{27,193452}⌋=⌊13\*0,193452⌋=⌊2,514876⌋=2.
3. h(k)=⌊13\*({97\*0,618033})⌋=⌊13\*{59,949201}⌋=⌊13\*0,949201⌋=⌊12,339613⌋=12.

Реализация метода на C++ с использованием оговоренных N и A:

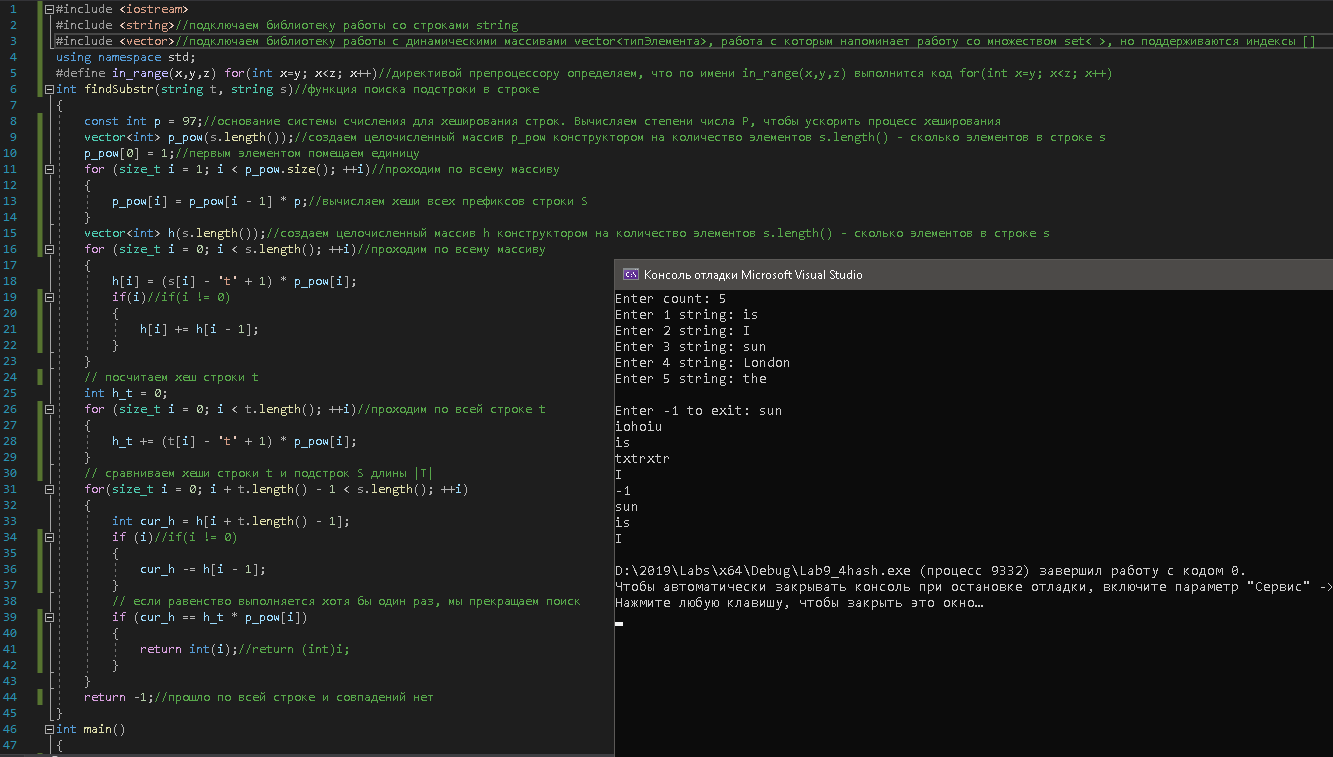


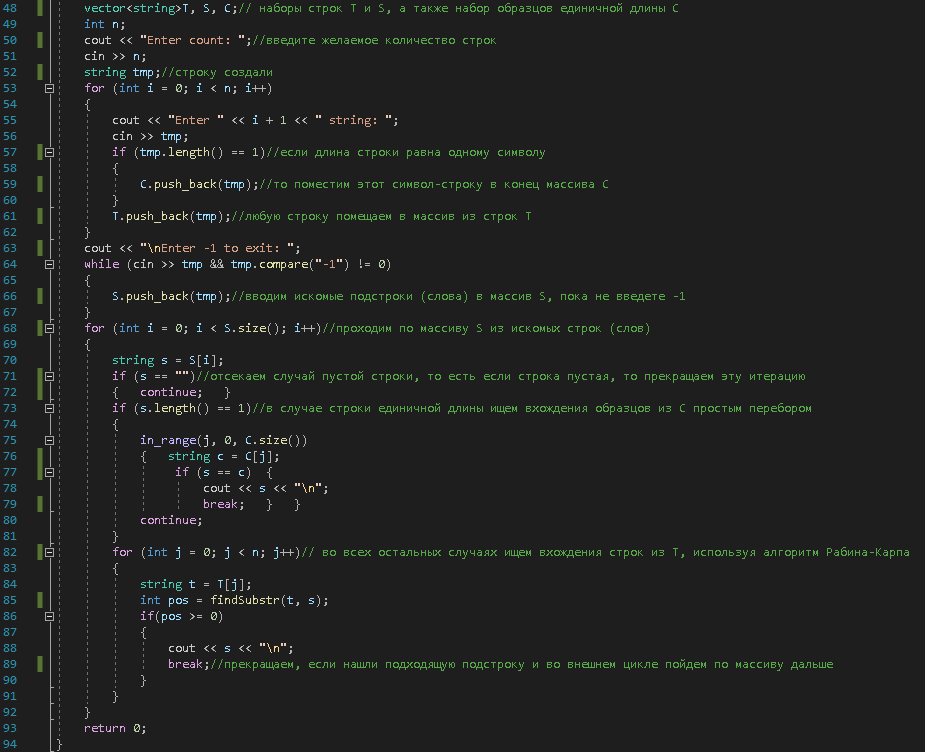
Функция fmod возвращает остаток от деления как целочисленного, так и нецелочисленного (вещественного). Например, для значений fmod(5.1, 2.4) вернет результат 0.3. Мы получаем вещественный результат, но возвращаем его из HashFunction как целочисленный, в результате чего берем только целочисленную часть числа, как и требуется по алгоритму. Также в программу вводим набор чисел из первого примера и видим, что коллизии есть на числа 44, 99, 133, чей хеш-код вычислился как 2, а числам 32 и 100 вычислился одинаковый хеш-код 10.

Если хеш-функция выдает недостаточно разбросанные посчитанные хеш-значения или их разрешенный диапазон небольшой, из-за чего они часто совпадают, то это создает коллизии (конфликты). Посчитанные значения обычно помещают в хеш-таблицу, чаще это динамический список, причем лучше когда он отсортированный, поэтому может подойти и множество set< > или мультимножество. Если посчитанные хеш-значения совпадают, а помещаются они, например, в массив, то место для совпадающего хеш-значения в массиве может оказаться занятым ранее посчитанным хеш-значением. **Способы разрешения коллизии**: 1) хеширование методом открытой адресации с линейным опробыванием, когда мы будем идти по массиву и искать ближайший пустой элемент, чтобы поместить в него очередной хеш, но расположение хешей при таком подходе будет уже неотсортированным; 2) хеширование методом цепочек (списков), или так называемое многомерное хеширование, когда в элементе с хешем у нас есть поле-указатель на другой элемент, обычно там адрес NULL, но если хеши совпали, то к первому элементу с таким хешем мы добавляем следующий элемент с таким же хешем, получая список списков.

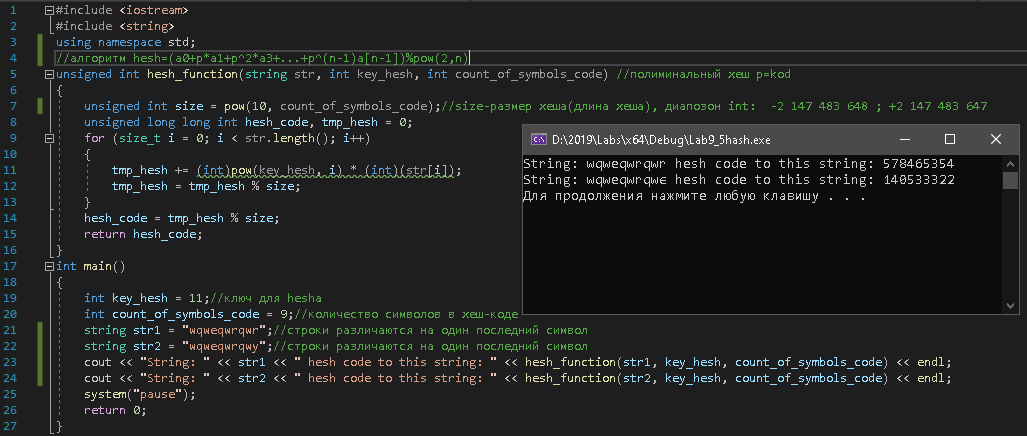
1. **Пример выполнения программы**

Написать программу, которая для каждой строки из заданного набора S проверяет, верно ли, что она содержит как подстроку одну из строк из набора T.

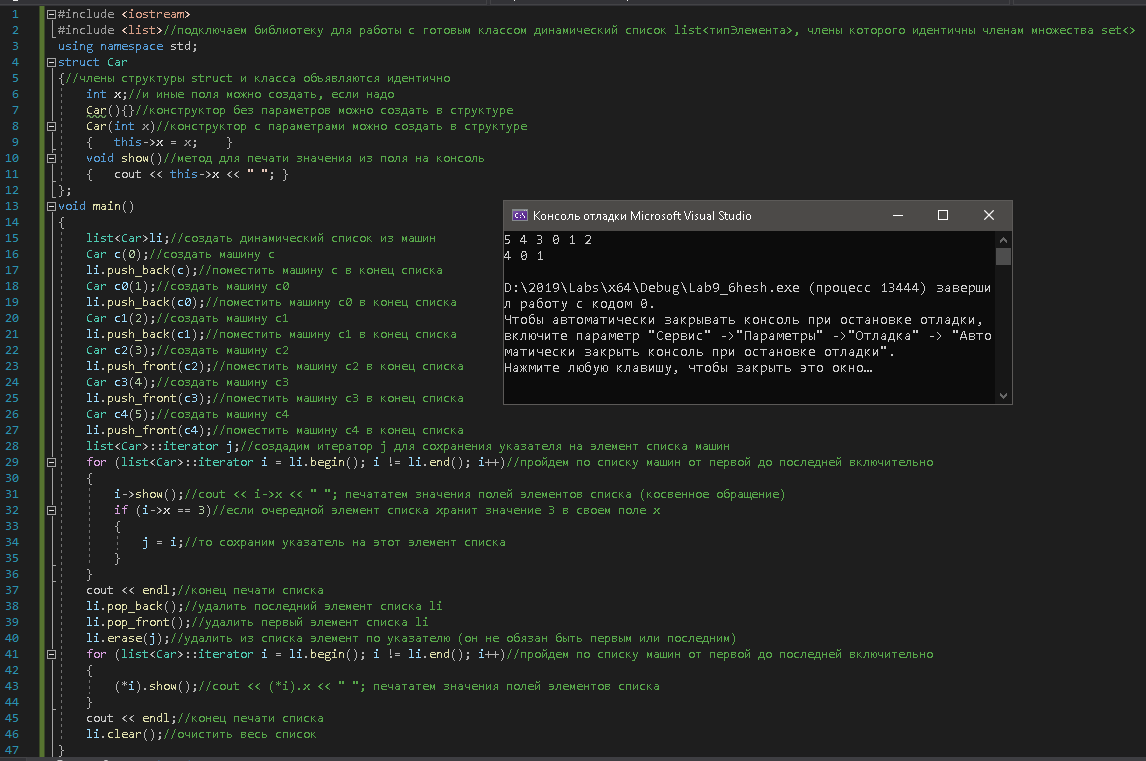




Написать программу с хеш-функцией, которая для строки находит хеш-код, принимая строку, длину хеш-кода (количество символов в хеш-коде), ключ хеш-кода).



Пример кода по работе с контейнером – динамическим списком list< >.



1. **Задания по вариантам**

В вашей программе должна быть реализована хеш-таблица list**<**Структура struct – хеш-таблица**>** или свой динамический список. Вместо структуры struct можно создать класс.

Cтруктура хеш-таблицы:

{

поле структура данных;

hesh-переменая; //она высчитывается по хеш-функции

указатель на следующий элемент хеш-таблицы (Структуру хеш-таблицы), у которой значение hesh-переменой такое же (равное);

};

Хеширование происходит по нескольким из ваших переменных (из поля структура данных). Если в вашем list'e есть одинаковые элементы (по значению рассчитанной hesh-переменой), то в list’e образуется список с одинаковыми hesh-переменными. Добавить консольное меню в вашу программу с функционалом: добавить новое значение в список, удалить значение из списка, изменить значение в списке, распечатать значения (и иное, если потребуется).

Варианты индивидуальных заданий:

1. Сформировать структуру, содержащую сведения о количестве изделий, собранных сборщиками цеха за неделю. Структурный тип содержит поля: фамилия сборщика, количество изделий, собранных им ежедневно в течение шестидневной недели, т.е. раздельно в понедельник, вторник и т.д. Написать программу, выдающую на печать: фамилию сборщика и общее количество деталей, собранных им за неделю.
2. Создать структуру, содержащую информацию о покупателе: ФИО, домашний адрес покупателя, суммы покупок, размер предоставляемой скидки. Вывести информацию о покупателе и сумме покупок с учетом скидки.
3. Создать структуру, содержащую информацию о складе, которая включает в себя наименование товара, количество единиц товара, цену единицы и дату поступления товара на склад. Вывести информацию о товаре и его стоимость.
4. Создать структуру, содержащую информацию о рейсе, где указаны его номер, тип автобуса, пункт назначения, время отправления и прибытия. Вывести информацию о рейсе и разницу между прибытием в пункт назначения и заданным временем.
5. Создать структуру, содержащую информацию междугородной АТС, которая содержит дату разговора, код и название города, время разговора, тариф, номер телефона в этом городе и номер телефона абонента. Вывести: город, общее время разговоров с ним и стоимость разговора.
6. Сформировать структуру данных с информацией об участнике спортивных соревнований, которая содержит: наименование страны, название команды, ФИО игрока, игровой номер, возраст, рост и вес. Вывести информацию об ИМТ игрока.
7. Создать структуру, содержащую информацию о сотруднике фирмы: ФИО, табельный номер, количество отработанных часов за месяц, почасовой тариф. Рабочее время свыше 144 часов считается сверхурочным и оплачивается в двойном размере. Вывести размер заработной платы сотрудника фирмы за вычетом подоходного налога, который составляет 12 % от суммы заработка.
8. Создать структуру, содержащую информацию о книге: регистрационный номер книги, автор, название, год издания, издательство, количество страниц. Вывести возраст книги.
9. Создать структуру Сотрудник предприятия, которая содержит: ФИО, номер отдела, должность, дату начала работы. Вывести стаж сотрудника.
10. Создать структуру Рейс о полетах самолетов. Для рейса указаны: номер рейса, тип самолета, пункт назначения, время вылета. Вывести информацию о рейсе и разницу между отправлением в пункт назначения и заданным временем.
11. Создать структуру Ремонт радиоаппаратуры. В радиоателье хранятся квитанции о сданной в ремонт радиоаппаратуре. Квитанция содержит следующую информацию: наименование группы изделий (телевизоры, радиоприемники и т.п.), марку изделия, дату приемки и время ремонта. Вывести информацию о состоянии заказа.
12. Создать структуру, содержащую информацию о горожанине: фамилия И.О., дата рождения, адрес. Посчитать возраст.
13. Создать структуру, содержащую информацию о постояльцах гостиницы: паспортные данные, даты приезда и отъезда, номер. Посчитать количество дней пребывания.
14. Создать структуру, содержащую информацию о клиенте банка: фамилия И.О., номер счета, сумма на счете, процент. Посчитать сумму после начисления процентов.
15. Создать структуру, содержащую информацию об ученике: фамилия, группа, отметки по физике, информатике, истории. Определить средний бал отметок по всем предметам.
16. **Контрольные вопросы**
17. Что за структура данных хеш-таблица и для чего она нужна?
18. Для чего используется хеширование? Дайте определение понятию.
19. Дайте определение понятию хеш-функция.
20. Какие есть функции хеширования? Перечислите их разновидности.
21. Дайте определение понятию коллизия?

**Литература**

**Дейтел,** Х.М. Как программировать на С++ / Х.М. Дейтел, П.Дж. Дейтел . – М. : Бином-Пресс , 2018 . – 1456 с.

**Павловская**, Т. А. С++. Объектно-ориентированное программирование : практикум / Т.А. Павловская, Ю.А. Щупак . – СПб. : Питер , 2019 . – 265 с.

**Страуструп**, Б. Язык программирования С++ / Б. Страуструп . – СПб. : Бином-Пресс , 2019 . – 1054 с.

Преподаватель Шаляпин Ю.В.

|  |
| --- |
| Рассмотрено на заседании цикловой  комиссии ПОИТ № 10  Протокол №\_\_\_\_от «\_\_\_»\_\_\_\_\_\_\_\_\_\_2021 г.  Председатель ЦК \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |