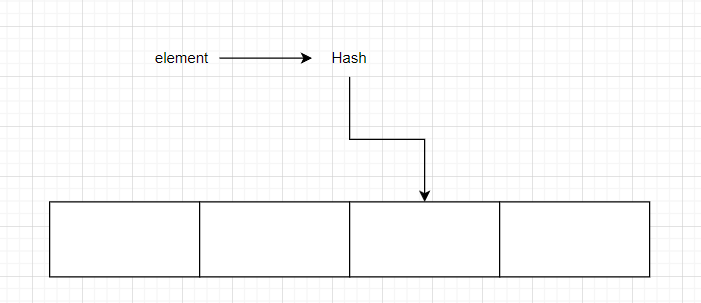
**ЛЕКЦИЯ 6**

**ХЕШ ТАБЛИЦЫ И ХЕШ ФУНКЦИИ**



1. Абстрактная структура данных с функционалом массивов и списков;
2. Hash – таблица – это два элемента (hash функция возвращающая число, которое находится в диапозоне индексов элемента);
3. Hash функция возвращает значение – целое число – индекс массива элемента;
4. Пример:

int x = hash(“John”);

//x is 4

hastable[x] = “John”;

1. Свойства хороших хеш – функций:
2. Их много;
3. Могу быть подходящими для данной структуры, которую надо обработать, или плохими;
4. Структура использует только те данные, которые обработаны хеш функцией и получили свой хеш;
5. Поскольку значения находятся в диапозоне индексов элементов, то они не должны выйти за границы хеш таблицы;
6. Для одних и тех же данных хеш совпадает (данные == данные, следовательно хеш == хеш);
7. Если данные отличаются крайне мало, необходимо подобрать такую хеш функцию, чтобы значения хешей у этих данных отличалось сильно;
8. Пример хеш функции:

usigned int hash()

{

int sum = 0;

for (int j = 0; str[j] != ‘\0’; j++)

{

Sum += str[j];

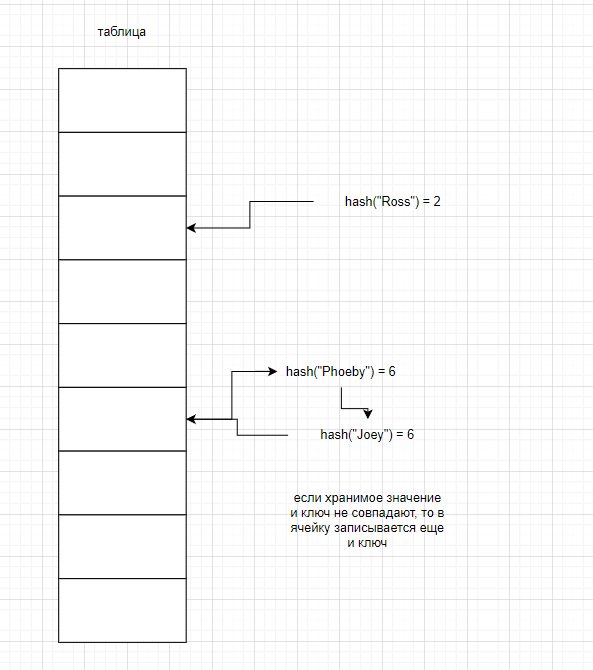
}

return sum%HASH\_MAX

}

Пробела – hash будет одинаков для строки и её перевернутого варианта (коллизия)

1. Коллизия – разные данные, прошедшие одну и туже hash функцию, возвращают одинаковые значения. Причины: плохая хеш – функция, слишком большое кол-во данных и маленькая таблица (тем меньше таблица и больше данных, тем больше коллизий);
2. Кластер – последовательность подряд идущих элементов таблицы, находящихся в одном хеше;
3. Методы разрешения коллизии:
4. Метод открытой адресации – размещения элемента с уже занятым hash в следующую ячейку, при его занятости – в hash + 2 и так далее. Плюсы – время решения остается линейным, что важно при большом числе данных. Минус – большое скопление рядом находящихся элементов и требует стартового подбора хеш функций.
5. Метод цепочек – в массиве хранятся указатели на начало связанного списка. Тобишь хеш-таблица – массив списков;



**РАЗБОР ЛАБОРАТОРНОЙ РАБОТЫ №11**

1. Дано: вариант, данные, ключ, хеш функция, использовать оба метода;
2. Для примера хеш-функция: H(k) = [M[kA mod 1]], где k – кол-во элементов, A коэффицент от 0 до 1, mod 1 – вернуть дробную часть;
3. Блок схема должна быть словестной.

ООП

1. Это способ программирования, который представляет программу в виде совокупности объектов, причем объект может быть собственным типом данных для автора. Кроме того используется понятие “класс”, которое увеличивает структурированность кода;
2. Класс – абстрактный тип данных (создается пользователем), модель реального объекта, в виде данных и функций;
3. Объект класса (переменная) – экземпляр класса, причем класс объявляется один раз, а объектов класса может быть несколько;
4. Характеристики объекта – его поля;
5. Три кита ООП: инкапсуляция, наследование, полиморфизм;
6. Классы лучше записывать в отдельные хедеры;

**Инкапсуляция**

1. Сокрытие части полей от пользователя;
2. Необходимо использовать спецификаторы доступа:
3. Public – методы и данные класса определяют его внешний интерфейс (доступны из любой части кода, даже вне класса);
4. Private – Методы и данные класса определяют его реализацию (доступно только внутри класса. Рекомендуется по умолчанию. Приватные методы используются для внутренних задач);
5. Protected – Методы и данные класса определяют его интерфейс для наследников класса (доступно внутри класса и его наследникам);

**НАСЛЕДОВАНИЕ**

1. Передача свойств методов от одного класса (родителя) к другому (наследнику). То есть принцип наследования позволяет избегать повторений при описании полей класса;

**ПОЛИМОРФИЗМ**

1. Применение одного и того же метода у разных объектов, а также получать разные результаты в зависимости от типа объекта;

**ПРИМЕР ОБЪЯВЛЕНИЯ**

1. Private – спецификатор по умолчанию;

**КОНСТРУКТОРЫ**

1. Метод класса, который исполняет инициализацию объекта. Вызывается автоматически при создании;
2. Конструктор и класс делят имя;
3. Конструктор можно перегружать (один класс может иметь несколько конструкторов);
4. Конструктор ничего не возвращает;
5. Если конструктор сам не создан, то компилятор сгенерирует его автоматически;
6. Конструктор копирования – конструктор, который в качестве параметра получает только ссылку на объект этого же типа;
7. Всегда публик.

**Деструктор**

1. Освобождает память, выделенную для этого объекта;
2. Делит имя с классом, но ~ стоит перед ним;
3. Ничего не принимает и не возвращает;
4. Вызывается автоматически когда объект выходит из области видимости (или убивается через delete);

**Статические классы**

1. Объекты типа static, которые принадлежат всему классу;
2. Существует в единственном экземпляре;
3. Инициализировать эти элементы в классе нельзя;

**Указатель this**

1. Указывает на свой объект.

**Дружественная функция**

1. Друзья имеют доступ ко всем полям класса;
2. Описывается вне класса, может даже быть глобальной;
3. Может быть методом другого класса;
4. Описывается в теле класса с ключевым словом friend;
5. Параметры – ссылка на объект класса;

**Дружественны класс**

1. Имеет доступ ко всем полям класса, которому является другом.

**ЛАБОРАТОРНЫЕ РАБОТЫ**