Белорусский государственный технологический университет

Факультет информационных технологий

Кафедра Программной Инженерии

Лабораторная работа 15

По дисциплине «Основы алгоритмизации и программирования»

На тему «Хеш-таблицы c цепочками»

Выполнил:

Студент 1 курса 3 группы

Шатерник Г.И.

Преподаватель: асс. Андронова М.В.

2023, Минск

5. В соответствии со своим вариантом разработать проект для условия из таблицы, представленной ниже. Построить хеш-таблицы разного размера с коллизиями.

Для вариантов с 1 по 8 вычисление хеш-функции произвести по методу универсального хеширования. Для вариантов с 9 по 16 при вычислении хеш-функции использовать алгоритм на основе исключающего ИЛИ для поля строки данных.

Исследовать время поиска информации.

**Паркинг**.Реализовать хеш-таблицу со следующими полями: № места, владельца места. Ключ – № места.

|  |
| --- |
| Программный код  #include <iostream>  #include <vector>  #include <string>  #include <ctime>  using namespace std;  // Структура для хранения информации о парковочном месте  struct ParkingSpot {  int number;  string owner;  };  // Функция вычисления хеша на основе исключающего ИЛИ  int hashFunction(int key, int tableSize) {  int hash = 0;  for (int i = 0; i < sizeof(int); ++i) {  hash ^= (key >> (i \* 8));  }  return hash % tableSize;  }  // Функция добавления элемента в хеш-таблицу  void insert(std::vector<ParkingSpot>& table, const ParkingSpot& spot) {  int index = hashFunction(spot.number, table.size());  while (table[index].number != -1) {  index = (index + 1) % table.size();  }  table[index] = spot;  }  // Функция поиска элемента в хеш-таблице  ParkingSpot\* search(std::vector<ParkingSpot>& table, int number) {  int index = hashFunction(number, table.size());  while (table[index].number != -1) {  if (table[index].number == number) {  return &table[index];  }  index = (index + 1) % table.size();  }  return nullptr;  }  int main() {  setlocale(LC\_ALL, "RUSSIAN");  const int tableSize = 10;  vector<ParkingSpot> parkingTable(tableSize, ParkingSpot{ -1, "" });  // Добавление парковочных мест  insert(parkingTable, ParkingSpot{ 1, "Глеб" });  insert(parkingTable, ParkingSpot{ 5, "Алексей" });  insert(parkingTable, ParkingSpot{ 9, "Сергей" });  // Поиск информации о парковочном месте  int searchNumber = 5;  clock\_t start = clock();  ParkingSpot\* result = search(parkingTable, searchNumber);  clock\_t end = clock();  if (result) {  cout << "Парковочное место " << searchNumber << " занято " << result->owner <<endl;  }  else {  cout << "Парковочное место " << searchNumber << " не найдено" <<endl;  }  // Вывод времени поиска информации  double searchTime = static\_cast<double>(end - start) / CLOCKS\_PER\_SEC;  cout << "Время поиска: " << searchTime << " сек" << endl;  return 0;  } |
| Вывод: |