山东大学 计算机科学与技术 学院

数据结构与算法 课程实验报告

|  |  |  |  |
| --- | --- | --- | --- |
| 学号：201700140056 | 姓名：李港 | | 班级：18.2 |
| 实验题目：实验三 数组描述线性表 | | | |
| 实验学时：2h | | 实验日期：2019.09.26 | |
| 实验目的：   * 1. 掌握线性表结构、数组描述方法（顺序存储结构）、数组描述线性表的实现。   2. 掌握线性表应用。 | | | |
| 软件开发工具：  Virtual Studio 2019 | | | |
| 1. **实验内容**    1. 创建线性表类：线性表的存储结构使用数组描述，提供操作: 插入、删除、查找等。    2. 设通讯录中每一个联系人的内容有：姓名、电话号码、班级、宿舍。    3. 由键盘输入或文件录入的通讯录信息建立通讯录表，使用线性表中操作实现通讯录管理功能，包括：插入、删除、编辑、查找（按姓名查找）；键盘输入一班级，输出通讯录中该班级所有人信息。 2. **数据结构与算法描述（整体思路描述，所需要的数据结构与算法）**    1. 本次实验建立vector.h头文件，编写vector类。    2. vector类使用泛型技术。    3. 枚举常见错误，如下标越界，并使用throw抛出错误，便于调试。    4. 提供增删改查，获取长度等功能。       1. 增：insert与push       2. 删：delete       3. 改：重载“[]”运算符。       4. 查：find函数线性搜索。    5. 在需要添加元素的函数中实现越界判断，并适时调用私有函数进行长度拓展。    6. 最后使用自己编写的vector实现通讯录功能，同样实现通讯录的增删改查。   **数据结构：**本实验是线性表实验，采用动态申请的泛型数组。  **算法：**本实验较为关键的部分是数组的动态扩展，其他部分难度不大，且无需特殊算法。   1. **测试结果（测试输入，测试输出）**   操作反馈正确，未发现错误。  本程序输入输出交替。  total:5  index: 0 name: name classnum: 0 dom: 0 phone: 1575332828  index: 1 name: name classnum: 1 dom: 1 phone: 1575332828  index: 2 name: name classnum: 0 dom: 2 phone: 1575332828  index: 3 name: name classnum: 1 dom: 3 phone: 1575332828  index: 4 name: name classnum: 0 dom: 4 phone: 1575332828  1: name  2: classnum  3: dom  4: phone  5: show all  6: find by class  7: show by index  8: insert by index  9: delete by index  Please input function number:  8  Please input new index:3  Please input new name:hhhh  Please input new class number:5  Please input new dom number:1  Please input new phone:15755555555  1: name  2: classnum  3: dom  4: phone  5: show all  6: find by class  7: show by index  8: insert by index  9: delete by index  Please input function number:6  Please input the class you want to show:1  index: 1 name: name classnum: 1 dom: 1 phone: 1575332828  index: 4 name: name classnum: 1 dom: 3 phone: 1575332828  1: name  2: classnum  3: dom  4: phone  5: show all  6: find by class  7: show by index  8: insert by index  9: delete by index  Please input function number:9  Please input new index:4  No such function!  1: name  2: classnum  3: dom  4: phone  5: show all  6: find by class  7: show by index  8: insert by index  9: delete by index  Please input function number:5  total:5  index: 0 name: name classnum: 0 dom: 0 phone: 1575332828  index: 1 name: name classnum: 1 dom: 1 phone: 1575332828  index: 2 name: name classnum: 0 dom: 2 phone: 1575332828  index: 3 name: hhhh classnum: 5 dom: 1 phone: 15755555555  index: 4 name: name classnum: 0 dom: 4 phone: 1575332828   1. **分析与探讨（结果分析，若存在问题，探讨解决问题的途径）**   经过测试，结果正常，本程序在执行逻辑上暂不存在问题，在实验过程中遇到的问题如下：  **1.**提交oj时因为不专心，导致对代码的简单修改都花费了两个小时，提交之后出现bug，debug花费了半个小时，bug原因是通讯录默认初始化了五条数据，这五条数据干扰了宿舍号异或和计算。  **解决**：应当尽量将工作在实验室完成，在宿舍干扰因素略多。  **2.**Debug可以通过将从文件获取输入的方式来简化操作，因为长时间不做题目，导致忘记了读取文件到标准输入流的写法，查询如何从文件读取数据花费五分钟左右，极其浪费时间。  **解决：**应当在常用代码片段中添加从文件读取数据这一项，方便日后查阅。   1. **附录：实现源代码（本实验的全部源程序代码，程序风格清晰易理解，有充分的注释）**   程序3-1 vector类  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* @brief vector类的简单实现 \*  \* @author TriAlley \*  \* @email lg139@139.com \*  \* @license GNU General Public License (GPL) \*  \* \*  \* Change History : \*  \* <Date> | <Version> | <Author> | <Description> \*  \* 2019/09/26 | 1.0 | TriAlley | Create file \*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #pragma once  #include<cstring> //memset、memcpy  #include<cstdlib> //malloc  /\*常见错误的枚举\*/  enum err { index\_out\_of\_range }err;  /\*\*  \* @brief vector类  \* private:  \* T\* head; //内容数组头指针  \* int maxindex; //指示index的最大值  \* int frontindex; //指向第一个空闲位置  \* void \_\_exlength()；//私有函数，提供将数组长度成两倍的功能  \*  \* public:  \* vector(); //构造函数，参数为初始长度，默认为50  \* ~vector(); //析构函数，释放构造函数申请的内存资源  \* T& operator [] (int i); //重载方括号[]  \* void push(T in); //push  \* void del(int i); //根据下标删除元素  \* void insert(int i, T in); //将元素插入到下标位置，其余元素后移  \* int find(T target); //查找元素，返回下标，查找失败返回-1  \* int getlength() { return frontindex; } //返回数组元素数量  \* int getmaxlength() { return maxindex + 1; } //返回数组最大长度  \*/  template<typename T>  class vector {  private:  T\* head; //the array head pointer  int maxindex; //the length-1 of array  int frontindex;  void \_\_exlength() {  T\* newhead = new T[(maxindex + 1) \* 2];  memcpy(newhead, head, sizeof(T) \* (maxindex + 1));  maxindex = maxindex \* 2 + 1;  delete[] head;  head = newhead;  }  public:  vector(int n = 50) {  maxindex = n - 1;  frontindex = 0;  head = new T[n];  memset(head, 0, sizeof(T) \* n);  }  ~vector() {  delete[] head;  }  T& operator [] (int i) {  if (i >= frontindex) {  throw index\_out\_of\_range;  }  return head[i];  }  void push(T in) {  if (frontindex == maxindex) {  \_\_exlength();  }  head[frontindex++] = in;  }  void del(int i) {  if (i >= frontindex) {  //err= index\_out\_of\_range  throw index\_out\_of\_range;  }  for (int j = i; j < frontindex; j++) {  head[j] = head[j + 1];  }  frontindex--;  }  void insert(int i, T in) {  if (i >= frontindex) {  throw index\_out\_of\_range;  }  //判断后移是否越界  if (frontindex + 1 > maxindex) {  \_\_exlength();  }  //向后复制时需要总尾开始，否则第一个元素将会覆盖全部  for (int j = frontindex; j >= i; j--) {  head[j + 1] = head[j];  }  head[i] = in;  frontindex++;  }  int find(T target) {  for (int i = 0; i <= frontindex; i++) {  if (head[i] == target) {  return i;  }  }  return -1;  }  int getlength() { return frontindex; }  int getmaxlength() { return maxindex + 1; }  };  程序3-2 使用通讯录类测试vector  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  \* @brief vector的测试程序 \*  \* @author TriAlley \*  \* @email lg139@139.com \*  \* @license GNU General Public License (GPL) \*  \* \*  \* Change History : \*  \* <Date> | <Version> | <Author> | <Description> \*  \* 2019/09/26 | 1.0 | TriAlley | Create file \*  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  #define \_CRT\_SECURE\_NO\_WARNINGS  #include<iostream>  #include<stdio.h>  using namespace std;  #include "vector.h"  //姓名、电话号码、班级、宿舍。由键盘输入或文件录入  typedef struct {  char name[20];  char phone[20];  int classnum;  int dom;  }student;  bool strcp(const char\* a, const char\* b) {  int i = 0;  while (a[i] != '\0' && b[i] != '\0') {  if (a[i] != b[i]) {  return false;  }  i++;  }  if (a[i] != b[i]) {  return false;  } else {  return true;  }  }  class contact {  private:  vector<student> studets;  public:  contact(int n = 5) {  /\*构建初始线性表\*/  //strcpy属于c标准库  for (int i = 0; i < n; i++) {  student temp;  strcpy(temp.name, "name");  strcpy(temp.phone, "1575332828");  temp.classnum = i % 2;  temp.dom = i;  studets.push(temp);  }  }  student& operator [] (int i) {  return studets[i];  }  int getlength() {  return studets.getlength();  }  int findindexbyname(char\* name) {  for (int i = 0; i < studets.getlength(); i++) {  if (strcp(studets[i].name, name)) {  return i;  }  }  return -1;  }  void deletebyindex(int i) {  studets.del(i);  }  void insertbyindex(int i, char\* name, int classnum, int dom, const char\* phone) {  student temp;  strcpy(temp.name, name);  temp.dom = dom;  temp.classnum = classnum;  strcpy(temp.phone, phone);  studets.insert(i, temp);  }  void push(char\* name, int classnum, int dom, const char\* phone) {  student temp;  strcpy(temp.name, name);  temp.dom = dom;  temp.classnum = classnum;  strcpy(temp.phone, phone);  studets.push(temp);  }  void changenamebyindex(int i, char\* iname) {  strcpy(studets[i].name, iname);  }  void changeclassbyid(int i, int iclassnum) {  studets[i].classnum=iclassnum;  }  void changedombyid(int i, int dom) {  studets[i].dom = dom;  }  void changephonebyid(int i, char\* phone) {  strcpy(studets[i].phone, phone);  }  void showall() {  cout << "\ntotal:" << studets.getlength()<<endl;  for (int i = 0; i < studets.getlength(); i++) {  showbyindex(i);  }  cout << endl;  }  void showbyindex(int i) {  cout << "index: " << i  << " name: " << studets[i].name  << " classnum: " << studets[i].classnum  << " dom: " << studets[i].dom  << " phone: " << studets[i].phone  << endl;  }  void showbyclass(int classnum) {  for (int i = 0; i < studets.getlength(); i++) {  if (studets[i].classnum == classnum) {  showbyindex(i);  cout << endl;  }  }  }  };  int main() {  try {  contact c;  c.showall();  for (;;) {  int index;  char temp[20];  int i = 0;  cout  << "\n1: name" << endl  << "2: classnum" << endl  << "3: dom" << endl  << "4: phone" << endl  << "5: show all" << endl  << "6: find by class" << endl  << "7: show by index" << endl  << "8: insert by index" << endl  << "9: delete by index" << endl  << "Please input function number:";  int func\_num = 0;  cin >> func\_num;  switch (func\_num){  case 1:  cout << "Please input the item you want to change:";  cin >> index;  i = 0;  cout << "Please input new name:";  getchar();  gets\_s(temp);  c.changenamebyindex(index, temp);  break;  case 2:  cout << "Please input the item you want to change:";  cin >> index;  cout << "Please input new class number:";  cin >> i;  c.changeclassbyid(index, i);  break;  case 3:  cout << "Please input the item you want to change:";  cin >> index;  cout << "Please input new dom number:";  cin >> i;  c.changedombyid(index, i);  break;  case 4:  cout << "Please input the item you want to change:";  cin >> index;  i = 0;  cout << "Please input new phone:";  getchar();  gets\_s(temp);  c.changephonebyid(index, temp);  break;  case 5:  c.showall();  break;  case 6:  cout << "Please input the class you want to show:";  cin >> index;  c.showbyclass(index);  break;  case 7:  cout << "Please input the index you want to show:";  cin >> index;  c.showbyindex(index);  break;  case 8:  char name[20];  char phone[20];  int index;  int dom;  int classnum;  cout << "Please input new index:";  cin >> index;  cout << "Please input new name:";  getchar();  gets\_s(name);  cout << "Please input new class number:";  cin >> classnum;  cout << "Please input new dom number:";  cin >> dom;  cout << "Please input new phone:";  getchar();  gets\_s(phone);  c.insertbyindex(index, name, classnum, dom, phone);  break;  case 9:  //int index;  cout << "Please input new index:";  cin >> index;  c.deletebyindex(index);  default:  cout << "No such function!";  break;  }  }  } catch (enum err err) {  switch (err) {  case index\_out\_of\_range:  cout << endl << "index\_out\_of\_range" << endl;  break;  default:  cout << endl << "unknown err" << endl;  }  }  } | | | |