《数据结构》实验报告

**实验一 顺序表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 姓名 | 刘奕霏 | 班级 | 22数字媒体技术专升本1班 | 学号 | 2206851503 |
| 实验名称 | 顺序表 | | | | |
| 实验时间 | 2022年10月20日 | | 成绩 | |  |
| 1. **实验目的：**   1.掌握线性表的顺序存储结构；  2.验证顺序表及其基本操作的实现；  3.理解算法与程序的关系，能够将顺序表算法转换为对应的程序。  **二、实验工具：**  软件平台：Windows7或以上版本，Visual Studio 2019  **三、实验原理：**  1、算法描述：  （1）显示函数：在我们定义的指针的长度中使用for循环，用%4d表示输出一个[整型](https://so.csdn.net/so/search?q=%E6%95%B4%E5%9E%8B&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/super_iron_man/article/details/_blank)数值，输出宽度为4。  （2）查找函数：用指针输入需要查找元素的位置，然后来判断如果指针是小于0或者大于指针的长度就返回0.否则我们就输出查找位置的元素。  （3）插入函数：如果指针长度超过最大容量就输出溢出，否则就再判断如果指针元素小于1或者大于指针长度就返回0，否则就将指针元素和它后面的所有元素都后移一个位置，然后将新的元素写入到空出的下标为指针元素-1的位置上，线性长度将增加1。  （4）删除函数：首先定义i为删除元素的位序号，删除元素的值通过我们定义的字符代出，如果i超出范围就输出删除位置不正确，否则就将i赋值给指针元素，把第i个元素后的所有元素依次向前移动一个位置，线性表长度减1.  **四、实验步骤和内容：**  1.建立一个顺序表，随机产生10个100以内的整数，并按要求完成：  （1）编写显示函数，在屏幕上显示顺序表中的10个整数；  （2）编写查找函数，从键盘输入任一整数在顺序表中查找，若找到，返回该元素在顺序表中的位置，否则提示无此元素；  （3）编写插入函数，从键盘输入待插入元素及插入位置，将完成插入后的顺序表输出；  （4）编写删除函数，从键盘输入待删除元素位置，将该位置元素删除后的顺序表输出。  #include <iostream>  #include <stdio.h>  #include <malloc.h>  #define Maxsize 10  typedef int elemtype;  typedef struct  {  elemtype v[Maxsize];  int len;  }sqlist;  int insert(sqlist\* L, int i, elemtype x)  /\*L为sqlist顺序表类型指针变量,i为插入元素的位序号，x为插入元素的值\*/  {  int j;  if ((\*L).len == Maxsize) /\*判断线性表的存储空间是否已满\*/  {  printf("溢出!\n");  return 0;  }  else  if ((i < 1) || (i > (\*L).len + 1)) /\*检查i值是否超出所允许的范围\*/  {  printf("插入位置不正确 \n");  return 0;  }  else  {  for (j = (\*L).len - 1; j >= i - 1; j--)  /\*将第i个元素和它后面的所有元素均后移一个位置\*/  (\*L).v[j + 1] = (\*L).v[j];  (\*L).v[i - 1] = x; /\* 将新的元素写入到空出的下标为i-1的位置上\*/  (\*L).len = (\*L).len + 1; /\*线性表的长度增加1\*/  return 1;  }  }  int delet(sqlist\* L, int i, elemtype\* y)  /\*L为sqlist顺序表类型指针变量,i为删除元素的位序号，删除元素的值通过y代出\*/  {  int j;  if ((i < 1) || (i > (\*L).len)) /\* 判断i值是否超出所允许的范围\*/  {  printf("删除位置不正确 \n");  return (0);  }  else  {  \*y = (\*L).v[i - 1]; /\* 把第i个元素赋给\*y \*/  for (j = i; j < (\*L).len; j++)  /\* 把第i个元素后的所有元素依次向前移动一个位置\*/  (\*L).v[j - 1] = (\*L).v[j];  (\*L).len = (\*L).len - 1; /\* 线性表长度减1\*/  return(1);  }  }  void print(sqlist\* l)  {  int i;  for (i = 0; i < (\*l).len; i++)  {  printf("%4d", (\*l).v[i]);  }  printf("\n");  }  void creat(sqlist\* l)  {  int x, i;  srand((unsigned)time(NULL));  printf("列表初始化顺序表随机数据为:\n");  for (i = 0; i < Maxsize; i++)  {  (\*l).v[i] = rand() % 101;  (\*l).len++;  }  (\*l).len = i;  }  int getElem(sqlist l)  {  int p;  int e;  printf("请输入要查找元素的位置:\n");  scanf\_s("%d", &p);  if (p<0 || p>l.len - 1)  {  return 0;  }  else  {  e = l.v[p - 1]; // 第p个元素对应数组中的L.data[p-1]  printf("你要查找的元素为：%d\n", e);  }  }  int main()  {  sqlist a;  //InitList(&a);  int i, x, result;  creat(&a);  print(&a);  //creat(&a);  getElem(a);  printf("请输入要删除数据的位置： \n");  scanf\_s("%d", &i);  result = delet(&a, i, &x);  if (result != 0)  print(&a);  printf("删除的数据是： %d\n", x);  printf("请输入要插入的位置和数据\n");  scanf\_s("%d%d", &i, &x);  result = insert(&a, i, x);  if (result != 0)  print(&a);  }  **S@YP}BM@Y~71LVA~54]0QFR**  **五、实验总结：**  在本次实验中，构建了顺序表，实现了顺序存储结构，在实验过程中，出现很多错误，有由于粗心造成的单词拼写错误，还有中英文状态的符号不正确，还有函数没有写完整，通过本次实验，对各种函数的实现和功能有了初步了解，通过实验运行结果体会顺序表的顺序存储，能够帮助我们更好的理解知识点，学习。  **六、教师评语：** | | | | | |