**实 验 报 告**

**（与程序设计有关）**

**课程名称： 数据结构**

**实验题目： 线性表**

**班级学号：** 2203050320

**姓 名:** 闻家尉

**成 绩：**

**沈 阳 理 工 大 学**

**2023年 11 月 1 日**

|  |
| --- |
| **实验目的及要求：**  1.掌握链表的灵活运用；  2.学习链表初始化和建立一个新的单循环链表；  3.知道怎样去实现单循环链表删除结点操作； |
| **软硬件环境：**计算机一台，Turbo C 软件或VC++ 软件 |
| **算法或原理分析（实验内容）：**  1. 问题描述  编号是1,2,…,n(n>0)的n个人按照顺时针方向围坐一圈，每人持有一正整数密码。开始时任选一个正整数作为报数上限值m，从某个人开始按顺时针方向自1开始顺序报数，报到m时停止报数，报m的人出列，将他的密码作为新的m值，从他在顺时针方向的下一个人开始重新从1报数，如此下去，直到所有人全部出列为止。令n最大值取30。设计一个程序来求出出列顺序，并输出结果。 2.基本要求建立模型，确定存储结构；对任意n个人，密码为m，实现约瑟夫环问题；出圈的顺序可以依次输出，也可以用一个数组存储。3.算法提示由于该问题是由古罗马著名史学家Josephus提出的问题演变而来，所以通常称之为Josephus问题。Josephus问题的解决需要采用循环链表，先构造一个有n个结点的单循环链表，再给出一个报数上限值m(假设m>1)，在链表的首结点开始从1计数，计到m时，对应的结点从链表中删除，然后在被删除结点的下一个结点又从1开始计数，直到最后一个结点从链表中删除算法结束。 本设计采用的是不带头结点的循环链表，其中循环链表中结点的结构如下：  typedef struct  { int num;  int cipher;  struct node \*next;  }linklist;  该问题可由两部分组成，分别由如下两个算法完成：  （1）建立一个由头指针head指示的有n个结点的约瑟夫单循环链表creat。  （2）寻找、输出和删除head所指的单循环链表的第m个结点select。该算法由如下具体步骤组成：  ①在head中的第一个结点起循环记数找第m个结点；  ②输出该结点的num值，把该结点的cipher(密码)值赋给m；  ③删除该结点；  ④转去执行①,直到所有结点被删除为止。 |

|  |  |  |  |
| --- | --- | --- | --- |
| **程序代码或实现过程：**  #include<bits/stdc++.h>  using namespace std;  struct Node  {  int location;  int number;  Node \*prior;  Node \*next;  };  struct FuNode  {  int loc\_result;  int num\_result;  Node \*curr;  };  class DoubleLinklist  {  public:  DoubleLinklist(int count,int number):Count(count),Number(number){}  void CreateLink();  void DeleteLink();  vector<int> node;    private:  int Count;  int Number;  Node \*head;  FuNode Again(Node\*);  };  void DoubleLinklist::CreateLink()  {  Node \*head = new Node;  Node \*next;  Node \*curr;  cout << "请输入第1个节点的密码：";  cin >> head->number;  head->location = 1;  curr = head;  for(int i=1;i < this->Count;i++)  {  next = new Node;  curr->next = next;  next->location = i+1;  cout << "请输入第" << i+1 << "个节点的密码：";  cin >> next->number;  next->prior = curr;  curr = next;  }  next->next = head;  head->prior = next;  this->head = head;  }  FuNode DoubleLinklist::Again(Node \*head)  {  FuNode no;  int num = 1;  if(head->next == head)  {  no.curr = NULL;  no.loc\_result = head->location;  no.num\_result = head->number;  cout << "删除的位置为：" << head->location << " 密码为：" << head->number << endl;  }  else  {  while(num != this->Number)  {  head = head->next;  num++;  }  no.curr = head->next;  no.loc\_result = head->location;  no.num\_result = head->number;  head->prior->next = head->next;  head->next->prior = head->prior;  cout << "删除的位置为：" << head->location << " 密码为：" << head->number << endl;  Number = head->number;  delete head;  }  return no;  }  void DoubleLinklist::DeleteLink()  {  FuNode no;  while(this->head != NULL)  {  no = Again(this->head);  this->node.push\_back(no.loc\_result);  this->node.push\_back(no.num\_result);  this->head = no.curr;  }  }  int main()  {  int i,pos;  cout << "请输入参与游戏的总人数和最开始指定的人的位置：";  cin >> i >> pos;  DoubleLinklist d(i,pos);  d.CreateLink();  cout << "出局顺序为：" << endl;  d.DeleteLink();  return 0;  } | | | |
| **结果分析：** | | | | |
| **教师签字** |  | **日 期** |  | |

*双面打印*