|  |  |  |
| --- | --- | --- |
| **软件学院实验报告** | | |
| 姓名：胡逢彬 学号：2225060803 专业： 网络工程 年级：2022级 | | |
| 课程名称 | | 数据结构 |
| 实验名称 | | 实验3 单链表的基本操作实现 |
| 实验的准备阶段 | 实验内容 | **实验3、单链表的基本操作实现**  （1）实验目的  通过该实验，深入理解链表的逻辑结构、物理结构等概念，掌握链表基本操作的编程实现，熟练掌握C语言中指针的操作。和实验2对比，掌握线性结构两种不同存储方式的区别。  （2）实验内容  编程实现链表下教材第二章定义的线性表的基本操作，最好用菜单形式对应各个操作，使其编程一个完整的小软件。注意，每个功能模块一定要考虑非法的情况，并作出相应的提示，例如：求前驱，要分别能够测试第一个元素的前驱、其他正常的元素的前驱、输入一个在表中不存在的元素求其前驱，这三种情况应给出相应的提示语和结果值；插入和删除时要考虑插入或删除的位置是否合法等。  （3）实验要求：  菜单项包括：  1.初始化或重置链表  2.销毁链表  3.清空链表  4.链表长度  5.指定位置的元素值  6.链表已存在元素的位序  7.求输入元素的直接前驱  8.求输入元素的直接后继  9.在第i个位置插入一个元素  10.删除第i个元素  11.输出有的链表元素  12.初始化并用头插法（或尾插法）输入元素  13.实现单链表的逆序存放  ***要求：***所有的**提示语和输出语句**不允许出现在自定义的函数中(输出函数除外)，只能在main函数中出现提示语。  注：销毁链表时需要循环释放每个结点所占用的空间。  注：求前驱是指，输入一个元素值（而不是位置），求该元素在顺序表中的直接前驱元素值。求后继是指：输入一个元素值（而不是位置），求该元素在顺序表中的直接后继元素值。  （4）验收/测试用例  参考实验2 |
| 实验类型 | 验证性 |
| 实验的重点、难点 | 重点： 链表的定义和实现  难点： 单链表的逆序存放 |
| 实验环境 | VC++6.0 |
| 实验的实施阶段 | 实验步骤及完成任务情况 | 一、设计思想  注意事项：  写出整体程序框架是怎么安排（菜单结构的设计）的，另外分别写出程序中每个函数的设计思想和思路方法等。  设计单链表的整体程序框架可以按照以下菜单结构来设计：  1. 初始化或重置链表  2. 销毁链表  3. 清空链表  4. 链表长度  5. 获取指定位置的元素值  6. 获取链表中指定元素的位置  7. 获取指定元素的直接前驱  8. 获取指定元素的直接后继  9. 在指定位置插入一个元素  10. 删除指定位置的元素  11. 输出链表中的所有元素  12. 初始化并用头插法输入元素  13. 实现单链表的逆序存放  接下来分别介绍每个函数的设计思想和思路方法：  1. 初始化或重置链表  - 设计思想：初始化或重置链表时，需要创建一个新的空链表。  - 方法：创建一个空链表结构，将头指针置为空。  2. 销毁链表  - 设计思想：销毁链表时，需要释放链表占用的内存空间，并将头指针置为空。  - 方法：逐个释放链表节点所占用的内存空间，然后将头指针置为空。  3. 清空链表  - 设计思想：清空链表时，需要将链表中的所有节点释放，并将头指针置为空。  - 方法：逐个释放链表节点所占用的内存空间，然后将头指针置为空。  4. 链表长度  - 设计思想：获取链表长度时，需要遍历链表并计算节点个数。  - 方法：从头节点开始遍历链表，统计节点个数直至链表末尾。  5. 获取指定位置的元素值  - 设计思想：根据指定的位置，定位并返回对应节点的元素值。  - 方法：从头节点开始遍历链表，找到指定位置的节点，返回节点的元素值。  6. 获取链表中指定元素的位置  - 设计思想：根据指定元素的值，遍历链表并返回第一次出现位置的索引。  - 方法：从头节点开始遍历链表，对比节点元素值，找到匹配值的节点并返回位置索引。  7. 获取指定元素的直接前驱  - 设计思想：根据指定元素的值，遍历链表并返回该元素的直接前驱节点。  - 方法：从头节点开始遍历链表，找到指定元素的前一个节点并返回。  8. 获取指定元素的直接后继  - 设计思想：根据指定元素的值，遍历链表并返回该元素的直接后继节点。  - 方法：从头节点开始遍历链表，找到指定元素的后一个节点并返回。  9. 在指定位置插入一个元素  - 设计思想：根据指定位置，将新元素插入到指定位置之前或之后。  - 方法：遍历链表找到指定位置的节点，插入新节点并调整指针指向。  10. 删除指定位置的元素  - 设计思想：根据指定位置，删除对应位置的节点。  - 方法：遍历链表找到指定位置的节点，进行节点删除操作并重新连接链表。  11. 输出链表中的所有元素  - 设计思想：遍历链表并依次输出每个节点的元素值。  - 方法：从头节点开始遍历链表，依次输出每个节点的元素值。  12. 初始化并用头插法输入元素  - 设计思想：利用头插法初始化链表，将新元素插入到链表头部。  - 方法：遍历输入的元素，将每个元素依次插入到链表头部。  13. 实现单链表的逆序存放  - 设计思想：将单链表中的节点逆序存放，即逆序排列链表中的节点。  - 方法：遍历链表，逐个调整节点的指针指向，使得链表中的节点逆序排列。  二、主要源代码  #include<bits/stdc++.h>  using namespace std;  #define ll long long  const int N=2e6+10;  typedef int ElemType;  typedef struct LNode {  ElemType data;  struct LNode \*next;  } LNode, \*Linklist;  Linklist head;  int pos;int num;  bool flag=false; //flag = true list exist  void InitList(); //1  void Page();  void DeletList(); //2  void ClearList(); //3  int GetListLength(int x); //4  int GetElem(int pos); //5  int GetPos(Linklist L,int a); //6  int GetPre(Linklist L,int a); //7  int Getnext(Linklist L,int a); //8  void Insert(Linklist L,int pos,int a); //9  int DeletElem(Linklist L,int pos); //10  void PrintList(Linklist L); //11  Linklist TailInsert(Linklist &L); //12  void Reverse(Linklist &L); //13  int main() {  ios::sync\_with\_stdio(false);  cin.tie(0);  cout.tie(0);  int f=1;  while(f) {  system("cls");  Page();  //system("cls");  int op;  cin>>op;  system("cls");  switch(op) {  case 1:  InitList();  break;  case 2:  DeletList();  break;  case 3:  ClearList();  break;  case 4:  GetListLength(1);  break;  case 5:  cin>>pos;  GetElem(pos);  break;  case 6:  cout<<"请输入您要查询的元素："<<endl;  cin>>num;  GetPos(head,num);  break;  case 7:  cout<<"请输入您要查询的元素："<<endl;  cin>>num;  if(GetPos(head,num)==1)  {  cout<<"第一个元素没有前驱！"<<endl;  }  else if(GetPre(head,num)==-999)  {  cout<<"该元素不存在！"<<endl;  }  else  {  cout<<"该元素的前驱为："<<GetPre(head,num)<<endl;  }  break;  case 8:  cout<<"请输入您要查询的元素："<<endl;  cin>>num;  if(GetPos(head,num)==GetListLength(0))  {  cout<<"最后一个元素没有后继！"<<endl;  }  else if(GetPre(head,num)==-999)  {  cout<<"该元素不存在！"<<endl;  }  else  {  cout<<"该元素的后继为："<<Getnext(head,num)<<endl;  }  break;  case 9:  cout<<"请输入您要插入的位置："<<endl;  cin>>pos;  if(pos<1||pos>GetListLength(0))  {  cout<<"您输入的位置不合法！"<<endl;  }  else  {  cout<<"请输入您要插入的元素："<<endl;  cin>>num;  Insert(head,pos,num);  cout<<"插入成功！"<<endl;  cout<<"插入后链表中的元素有："<<endl;  PrintList(head);  }  break;  case 10:  cout<<"请输入您要删除的元素位置："<<endl;  cin>>pos;  if(pos<1||pos>GetListLength(0))  {  cout<<"您输入的位置不合法！"<<endl;  }  else  {  cout<<"您已成功删除元素:"<<DeletElem(head,pos)<<endl;  cout<<"删除后链表中的元素有："<<endl;  PrintList(head);  }  break;  case 11:  if(head==NULL)  {  cout<<"空链表"<<endl;  break;  }  cout<<"链表中的所有元素为："<<endl;  PrintList(head);  break;  case 12:  cout<<"请问你要输入多少个数据："<<endl;  TailInsert(head);  cout<<"初始化成功！"<<endl;  break;  case 13:  cout<<"当前链表为："<<endl;  PrintList(head);  cout<<endl;  cout<<"将链表逆序存放输出结果为："<<endl;  Reverse(head);  PrintList(head);  break;  case -1:  f=0;  break;  default:  f=0;  break;  }  system("pause");  }  }  void InitList() {  head=(LNode \*)malloc(sizeof(LNode));  if(head!=NULL) {  head->data=114514;  cout<<"创建成功"<<endl;  } else {  cout<<"创建失败"<<endl;  return ;  }  head->next=NULL;  }  void DeletList() {  Linklist L=head;  Linklist p=NULL;  while(L) {  p = L;  L = L->next;  free(p);  }  cout<<"已销毁"<<endl;  }  void ClearList() {  Linklist L=head->next;  Linklist p=NULL;  while(L) {  p = L;  L = L->next;  free(p);  }  head->next=NULL;  cout<<"已清空"<<endl;  }  //4  int GetListLength(int x) {  int len=0;  Linklist p=head;  // if(p)len++;  while(p->next) {  len++;  p=p->next;  }  if(x==1)  cout<<"链表长度为"<<len<<endl;  return len;  }  //5  int GetElem(int pos)  {  pos+=2;  Linklist p=head->next;  // while(p) {  // cout<<p->data<<" ";  // p=p->next;  // }  // cout<<endl;  //Linklist p=head;  p=head;  int m=1;  if(pos>GetListLength(0)+2)  {  cout<<"超出链表长度，查找失败"<<endl;  return -1;  }  else  {  cout<<"查找成功"<<endl;  }  // cout<<head->data<<endl;  while(p->next)  {  m++;  if(m==pos) {  cout<<"链表中第"<<pos-2<<"个元素的值是"<<p->data<<endl;  return p->data;  }  p=p->next;  }  cout<<"链表中第"<<pos-2<<"个元素的值是"<<p->data<<endl;  return -1;  }  //6  int GetPos(Linklist L,int a)  {  Linklist p;  p=L->next;  int cnt=0;    while(p)  {  // cout<<p->data<<"\*"<<endl;  cnt++;  if(p->data==a) {  cout<<"位置为第"<<cnt<<"个"<<endl;  return cnt;  }  p=p->next;    }    return ++cnt;  }  //7  int GetPre(Linklist L,int a) {  Linklist p=L->next,q=L;  while(p) {  if(p->data==a) {  return q->data;  }  p=p->next;  q=q->next;  }  return -999;  }  int Getnext(Linklist L,int a) {  Linklist p=L->next;  Linklist q=p->next;  while(p) {  if(p->data==a) {  return q->data;  }  p=p->next;  q=q->next;  }  return -999;  }  void Insert(Linklist L,int pos,int a) {  Linklist p=L;  int b=0;  while(p&&b<pos-1) {  b++;  p=p->next;  }  if(b==pos-1) {  Linklist q=(Linklist)malloc(sizeof(LNode));  q->data=a;  q->next=p->next;  p->next=q;  }  }  int DeletElem(Linklist L,int pos) {  Linklist p=L;  int m=0;  int res=0;  while(p&&m<pos-1) {  m++;  p=p->next;  }  if(m==pos-1) {  Linklist q;  q=p->next;  res=q->data;  p->next=q->next;  free(q);  return res;  }  return -1;  }  //11  void PrintList(Linklist L) {  Linklist p=L->next;  while(p) {  cout<<p->data<<" ";  p=p->next;  }  cout<<endl;  }  //12  Linklist TailInsert(Linklist &L)  {  int n;  cin>>n;  L=(LNode\*)malloc(sizeof(LNode));  L->next=NULL;  Linklist p;  p=L;  int a;  cout<<"请输入n个数字"<<endl;  for(int i=0; i<n; i++) {  Linklist q=(Linklist)malloc(sizeof(LNode));  cin>>a;  q->data=a;  p->next=q;  p=q;  }  p->next=NULL;  return L;  }  //13  void Reverse(Linklist &L) {  Linklist p=L->next;  L->next=NULL;  while(p!=NULL) {  Linklist q=p;  p=p->next;  q->next=L->next;//把q插入到L和L->next 中间  L->next=q;  }  }  void Page() {  cout<<"1.初始化或重置链表"<<endl;  cout<<"2.销毁链表"<<endl;  cout<<"3.清空链表"<<endl;  cout<<"4.链表长度"<<endl;  cout<<"5.指定位置的元素值"<<endl;  cout<<"6.链表已存在元素的位序"<<endl;  cout<<"7.求输入元素的直接前驱"<<endl;  cout<<"8.求输入元素的直接后继"<<endl;  cout<<"9.在第i个位置插入一个元素"<<endl;  cout<<"10.删除第i个元素"<<endl;  cout<<"11.输出有的链表元素"<<endl;  cout<<"12.初始化并用头插法（或尾插法）输入元素"<<endl;  cout<<"13.实现单链表的逆序存放"<<endl;  cout<<"退出(输入一个负数！)"<<endl;  cout<<"请输入您的选择："<<endl;  } |
| 实验结果的处理阶段 | 实验结果 | IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256  IMG_256 |
| 实验结果总结 | （注意：请同学们看完要求，把本段红色文字删除！！！）  写自己遇到的问题，以及解决的方法。至少写两个。  或者写遇到的一个问题，写一段自己的收获，自己学会了什么。 |