**《数据结构综合设计》实验报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 学院 | 人工智能与大数据学院 | 专业 | 虚拟现实技术 | 班级 | 21级1班 | 学生姓名 | 翟家杏 |
| 实验  周次 | 5-6 | 实验  日期 | 2023.4.3 | 学时 | 4 | 教师姓名 | 李昊康 |
| 项目名称 | | 树与二叉树的实现及应用 | | | | | |
| 实验  类别 | 🗹验证型实验 🞎设计型实验 🞎综合型实验 🞎其它 | | | | | 成绩：94 | |
| 1. 实验目的及具体要求   实验目的：   1. 掌握二叉树的特点及其存储方式。 2. 掌握二叉树的创建和显示方法。 3. 复习二叉树遍历的概念，掌握二叉树遍历的基本方法。 4. 掌握求二叉树的叶子节点数、树的总结点数和树的深度等基本算法。   具体要求：  用先序法建立一颗二叉树，并能按照广义表表示法显示二叉树结构。  编写先序遍历、中序遍历、后序遍历程序。  编写求二叉树结点数、树的总结点树和深度的程序。  设计选择式菜单，以选择菜单的方式进行操作。   1. 实验仪器、设备和材料   装有并能运行VS2019的电脑。   1. 实验内容、步骤及实验数据记录   #include <stdio.h>  #include<malloc.h>  #define MAX 100  int count = 0;  typedef struct tnode  {  char data;  struct tnode\* lchild, \* rchild;  }BT;  BT\* CreateBTree()  {  BT\* t;  char ch;  scanf("%c", &ch);  getchar();  if (ch == '0')  t = NULL;  else  {  t = (BT\*)malloc(sizeof(BT));  t->data = ch;  printf("请输入%c结点左孩子结点:", t->data);  t->lchild = CreateBTree();  printf("请输入%c结点右孩子结点:", t->data);  t->lchild = CreateBTree();  }  return t;  }  void ShowBTree(BT\* T)  {  if (T != NULL)  {  printf("%c", T->data);  if (T->lchild != NULL)  {  printf("(");  ShowBTree(T->lchild);  if (T->rchild != NULL)  {  printf(",");  ShowBTree(T->rchild);  }  printf("(");  }  else  if (T->rchild != NULL)  {  printf(")");  ShowBTree(T->lchild);  if (T->rchild != NULL)  {  printf(",");  ShowBTree(T->rchild);  }  printf(")");  }  }  }  void PreOrder(BT\* T)  {  if (T == NULL) return;  else  {  printf("%c", T->data);  PreOrder(T->lchild);  PreOrder(T->rchild);  }  }  void InOrder(BT \*T)  {  if (T == NULL) return;  else  {  InOrder(T->lchild);  printf("%c", T->data);  InOrder(T->rchild);  }  }  void PostOrder(BT \* T)  {  if (T== NULL) return;  else  {  PostOrder(T->lchild);  PostOrder(T->rchild);  printf(" % c", T->data);  }  }  void LevelOrder(BT\* T)  {  int f, r;  BT\* p, \* q[MAX];  p = T;  if (p != NULL)  {  f = 1; q[f] = p; r = 2;  }  while (f != r)  {  p = q[f];  printf("%c", p->data);  if (p->lchild != NULL)  {  q[r] = p->lchild; r = (r + 1) % MAX;  }  if (p->rchild != NULL)  {  q[r] = p->rchild;  r = (r + 1) % MAX;  }  f = (f + 1) % MAX;  }  }  void Leafnum(BT\* T)  {  if (T)  {  if (T->lchild == NULL && T->rchild == NULL)  count++;  Leafnum(T->lchild);  Leafnum(T->rchild);  }  }  void Nodenum(BT\* T)  {  if (T)  {  count++;  Nodenum(T->lchild);  Nodenum(T->rchild);  }  }  int TreeDepth(BT\* T)  {  int ldep = 0, rdep = 0;  if (T == NULL)  return 0;  else  {  ldep = TreeDepth(T->lchild);  rdep = TreeDepth(T->rchild);  if (ldep > rdep)  return ldep + 1;  else  return rdep + 1;  }  }  void MenuTree()  {  printf("\n 二叉树子系统");  printf("\n=======================");  printf("\n| 1-建立一个新二叉树 |");  printf("\n| 2-广义表 表示法显示 |");  printf("\n| 3-先序遍历 |");  printf("\n| 4-中序遍历 |");  printf("\n| 5-后序遍历 |");  printf("\n| 6-层次遍历 |");  printf("\n| 7-求叶子树结点数目 |");  printf("\n| 8-求二叉树总结点数目 |");  printf("\n| 9-求树深度 |");  printf("\n| 0-返回 |");  printf("\n|===================");  printf("\n请输入菜单号(0-9) :");  }  main()  {  BT\* T = NULL;  char ch1, ch2, a;  ch1 = 'y';  while (ch1 == 'y' || ch1 == 'Y')  {  MenuTree();  scanf("%c", &ch2);  getchar();  switch (ch2)  {  case'1':  printf("请按先序序列输入二叉树的结点:\n");  printf("说明：输入结点后按回车键('0'表示后继结点为空):\n");  printf("请输入根结点:");  T = CreateBTree();  printf("二叉树成功建立！");  break;  case'2':  printf("二叉树广义表表示法如下：");  ShowBTree(T);  break;  case'3':  printf("二叉树的先序遍历序列为：");  PreOrder(T);  break;  case'4':  printf("二叉树的中序遍历序列为：");  InOrder(T);  break;  case'5':  printf("二叉树的后序遍历序列为：");  PostOrder(T);  break;  case'6':  printf("二叉树的层次遍历序列为：");  LevelOrder(T);  break;  case'7':  count = 0; Leafnum(T);  printf("该二叉树共有%d个叶子。", count);  break;  case'8':  count = 0; Nodenum(T);  printf("该二叉树共有%d个结点。", count);  break;  case'9':  printf("该二叉树的深度是%d.", TreeDepth(T));  break;  case'0':  ch1 = 'n';  break;  default:  printf("输入有误，请输入0-9进行选择！");  }  if (ch2 != '0')  {  printf("\n按回车键继续，按任意键返回主菜单！\n");  a = getchar();  if (a != '\xA')  {  getchar();  ch1 = 'n';  }  }  }  }   1. 实验结果及分析   1.建立一个新二叉树     1. 广义表表示法显示      1. 先序遍历      1. 中序遍历      1. 后序遍历      1. 层次遍历      1. 求叶子结点数目      1. 求二叉树总结点数目      1. 求树深度      1. 0——返回 2. 实验分析   通过实验，学习并掌握了二叉树的特点及其存储方式。同时懂得了二叉树的创建和显示方法。在此基础上又复习了二叉树遍历的概念，掌握了二叉树遍历的基本方法。了解了求二叉树的叶子节点数、树的总结点数和树的深度等基本算法。 | | | | | | | |

说明：1. 实验周次：填写实际上课周，如第5-8周上课填“5-8”或第10周上课填“10”。

1. 实验报告各部分内容需详实填写，按实验指导书上的评分标准给出分数。
2. 实验类型参考实验类型说明文件。