**计算机科学与工程学院实验报告3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **实验课程名称** | | **实验三 树和图应用类实验**  **——二叉树算术表达式求值** | | | **实验成绩** |  |
| **专业** | **计算机科学与技术** | | **班级** | **\*** | **指导教师签字** |  |
| **姓名** | **\*** | | **姓名** | **\*** | **姓名** |  |
| **学号** | **\*** | | **学号** | **\*** | **学号** |  |
| **附录实验程序及结果：**  **实验分工以及占比：**  **\*50%：**利用二叉树实现中缀表达式转前/后缀表达式  **\*50%：**利用二叉树实现后缀表达式求值  **实验程序：**    **结果：**    **源代码：**  #include<stdio.h>  #include<string.h>  #include<math.h>  #define **Inf** 1e9  struct **tree**  {      double date;      char ch;  **tree** \*l,\*r;  **tree**()      {          ch='\0';          date=0;          l=r=**NULL**;      }  };  char st[1005];  double **judge**(char \*s,int x,int y,double &n)  {      double num=0;      int i,ok=0;      for(i=x; i<y; i++)          if(s[i]>='0'&&s[i]<='9')          {              if(!ok)num=num\*10+s[i]-'0';              else num+=(s[i]-'0')\***pow**(10,ok-i);          }          else if(s[i]=='.'){ok=i;}          else return 0;      return n=num;  }  int **is\_operator**(char c)  {      if(c=='+'||c=='-'||c=='\*'||c=='/')return 1;      return 0;  }  **tree** \***build**(char \*s,int x,int y)  {  **tree** \*now=new **tree**;      double num=**Inf**;  **judge**(s,x,y,num);  *//printf("%c\n",s[x]);*      if(num!=**Inf**)      {          now->date=num;          return now;      }      int p=0,c1=-1,c2=-1;      for(int i=x; i<y; i++)if(s[i]=='-'&&**is\_operator**(s[i-1]))          {              int t=2;              st[0]='(';              st[1]='0';              int k=0,ok=1;              for(int j=i; j<y; j++)              {                  if(s[i]=='(')k++;                  else if(s[i]==')')k--;                  st[t++]=s[j];                  if((ok&&j==y-1)||(!k&&**is\_operator**(s[i+1])))                  {                      ok=0;                      st[t++]=')';                  }              }              s[t]='\0';  **memcpy**(s+i,st,sizeof(st));              i--;              y+=3;          }          else              switch(s[i])              {              case '(':p++;break;              case ')':p--;break;              case '+':              case '-':if(!p)c1=i;break;              case '\*':case '/':if(!p)c2=i;break;              }      if(c1<0)c1=c2;      if(c1<0)return **build**(s,x+1,y-1);      now->l=**build**(s,x,c1);      now->r=**build**(s,c1+1,y);      now->ch=s[c1];      return now;  }  double **dfs**(**tree** \*p)  {      if(!p)return 0;      if(p->l==p->r&&p->l==**NULL**)return p->date;      switch (p->ch)      {      case '+':          return p->date=**dfs**(p->l)+**dfs**(p->r);          break;      case '-':          return p->date=**dfs**(p->l)-**dfs**(p->r);          break;      case '\*':          return p->date=**dfs**(p->l)\***dfs**(p->r);          break;      case '/':          return p->date=**dfs**(p->l)/**dfs**(p->r);          break;      }      return 0;  }  void **dfs**(**tree** \*p,int choose)  {      if(!p)return ;      if(p->l==p->r&&p->l==**NULL**)      {  **printf**(" %g",p->date);      }      if(!choose)      {  **printf**("%c",p->ch);  **dfs**(p->l,choose);  **dfs**(p->r,choose);      }      else      {  **dfs**(p->l,choose);  **dfs**(p->r,choose);  **printf**("%c",p->ch);      }  }  char s[1005];  int **main**()  {  **tree** \*root=**NULL**;      while(**gets**(s)!=**NULL**)      {          root=**NULL**;          int len=**strlen**(s);          root=**build**(s,0,len);          double ans=**dfs**(root);  **puts**("前缀表达式：");  **dfs**(root,0);  **puts**("");  **puts**("后缀表达式：");  **dfs**(root,1);  **puts**("");  **printf**("%s=%g\n",s,ans);      }      return 0;  } | | | | | | |
| **实验课程总结**  通过本次实验课程内容的学习，我了解了有关树和图数据结构的相关问题。主要学习的是有关二叉树的使用以及dfs,bfs算法的有关使用，本次实验中利用二叉树实现了算术表达式求值。本次实验内容提升了我对树和图相关知识的了解，加强了有关树的应用。 | | | | | | |

**批改时间：**