LINUX堆、栈、红黑树试验报告

姓名：潘白冰

学号：161403210

班级：硬件二班

一：实验内容

建立堆排序及其算法

构造栈的算术表达式

绘制红黑树

均运用C/C++语言完成

二：实验代码

堆：

#include <cstdio>

#include <cstring>

#include <iostream>

#include <queue>

using namespace std;

///宏定义

#define ElemType int

#define OK 1

#define ERROR 0

#define INFEASIBLE -1

#define OVERFLOW -2

//堆结构体，以小根堆为基准

typedef struct HeapNode

{

int verb;

struct HeapNode \*left;

struct HeapNode \*right;

}HeapNode,\*Heap;

//堆的插入

void insertHeap(int v,Heap &L)

{

if(!L)

{

L=new HeapNode;

L->verb=v;

L->left=NULL;

L->right=NULL;

return;

}

if(!(L->left)) insertHeap(v,L->left);

else if(!(L->right))

{

insertHeap(v,L->right);

if(L->left->verb>v)

{

int t=L->left->verb;

L->left->verb=v;

L->right->verb=t;

}

}

else if(v<L->verb) insertHeap(v,L->left);

else insertHeap(v,L->right);

if(L->left->verb<L->verb)

{

int t=L->left->verb;

L->left->verb=L->verb;

L->verb=t;

}

}

//堆的调整

void initHeap(Heap &L)

{

if(!(L->left)||!(L->right)) return;

if(L->left->verb>L->right->verb)

{

Heap tp=L->left;

L->left=L->right;

L->right=tp;

}

if(L->left) initHeap(L->left);

else return;

if(L->verb>L->left->verb)

{

int t=L->verb;

L->verb=L->left->verb;

L->left->verb=t;

}

}

//堆的删除

void deleteHeap(Heap &L)

{

if(!(L->left))

{

if(!(L->right))

{

L=NULL;

return;

}

L=L->right;

return;

}

L->verb=L->left->verb;

deleteHeap(L->left);

if(L->right&&L->verb>L->right->verb)

{

insertHeap(L->verb,L->right);

L=L->right;

}

}

//BFS输出二叉堆树形图

void bfs(Heap L)

{

queue<Heap> q,tq;

bool sign=false;

Heap mp;

q.push(L);

while(!q.empty())

{

sign=false;

while(!q.empty())

{

tq.push(q.front());

if(q.front()->verb!=-1) sign=true;

q.pop();

}

if(!sign) break;

while(!tq.empty())

{

Heap tp=tq.front();

if(tp->verb!=-1) printf("%d ",tp->verb);

else printf("空 ");

if(tp->left) q.push(tp->left);

else

{

mp=new HeapNode;

mp->verb=-1;

mp->left=mp->right=NULL;

q.push(mp);

}

if(tp->right) q.push(tp->right);

else

{

mp=new HeapNode;

mp->verb=-1;

mp->left=mp->right=NULL;

q.push(mp);

}

tq.pop();

}

printf("\n");

}

}

int main()

{

int num;

Heap root=NULL;

cout<<"请输入整数个数：";

cin>>num;

int a[100];

cout<<"请依次按顺序输入整数数据，每个数据一行，注意输入数据不能有误。"<<endl;

for(int i=0;i<num;i++)

{

scanf("%d",&a[i]);

insertHeap(a[i],root);

}

initHeap(root);

cout<<"树形图如下："<<endl;

bfs(root);

cout<<"按照小根堆从小到大排序为：";

while(root)

{

printf("%d ",root->verb);

initHeap(root);

deleteHeap(root);

}

printf("\n");

getchar();getchar();

return 0;

}

栈代码：

由于栈需要很多方法，所以另写STACK方法

Stack：

#include <iostream>

using namespace std;

#define MAXSIZE 100

#define OK 1

#define ERROR 0

#define INFEASIBLE -1

#define OVERFLOW -2

#define Status int

template<typename ElemType>

class SqStack

{

public:

ElemType \*base;

ElemType \*top;

int stacksize;

//构造函数

SqStack(){

base=new ElemType[MAXSIZE];

if(!base) exit(OVERFLOW);

top=base;

stacksize=MAXSIZE;

}

//顺序栈的入栈操作

void Push(ElemType e){

if(top-base==stacksize) {cout<<"ERROR"<<endl;return;}

\*top++=e;

}

//顺序栈的出栈操作

void Pop(){

if(top==base) {cout<<"ERROR"<<endl;return;}

--top;

}

//取栈顶元素

ElemType GetTop(){

if(top==base) exit(1);

return \*(top-1);

}

//判断栈是否为空

bool Empty(){

if(base==top) return true;

return false;

}

//判断栈是否已满

bool Full(){

if(top-base==stacksize) return true;

return false;

}

};

栈算术表达式代码：

#include "stack.h"

#include <cstring>

#include <iostream>

using namespace std;

SqStack<char> optr,opnd;

SqStack<double> oprd;

char sign[8]="+-\*/()#"; //符号数组（对应下标）

//表征符号优先级的数组

int p[8][8]={{1,1,-1,-1,-1,1,1},{1,1,-1,-1,-1,1,1},{1,1,1,1,-1,1,1},{1,1,1,1,-1,1,1},{-1,-1,-1,-1,-1,0,-2},{1,1,1,1,-2,1,1},{-1,-1,-1,-1,-1,-2,1}};

//构造数据的算法

double Combine()

{

int front=0;

opnd.Push('\0');

char \*sss;

sss=opnd.base;

if(sss[0]=='-') front=1;

int point=(int)strlen(sss);

int i;

for(i=0;i<(int)strlen(sss);i++)

if(sss[i]=='.')

{point=i;break;}

double s=0;

double t=1;

for(i=point-1;i>=front;i--)

{

s+=(sss[i]-48)\*t;

t\*=10;

}

t=1;

for(i=point+1;i<(int)strlen(sss);i++)

{

t/=10;

s+=(sss[i]-48)\*t;

}

while(!opnd.Empty())

opnd.Pop();

return (!front)?s:s\*(-1);

}

//括号匹配检验算法。

bool Match(char x[])

{

int i;

bool p=true;

SqStack<char> s;

if(x[(int)strlen(x)-1]!='#')

return false;

for(i=0;i<strlen(x);i++)

{

if(x[i]=='(')

s.Push(x[i]);

if(s.Empty()&&x[i]==')')

return false;

if(x[i]==')'&&s.GetTop()!='(')

{p=false;break;}

if(x[i]==')')

s.Pop();

}

return p;

}

//优先级检验算法

int Precede(char a,char b)

{

int i;

int Sp,Sq;

for(i=0;i<8;i++)

{

if(a==sign[i])

Sp=i;

if(b==sign[i])

Sq=i;

}

return p[Sp][Sq];

}

//判断c是否为运算符

bool In(char c)

{

int i;

bool p=false;

for(i=0;i<4;i++)

if(c==sign[i])

{p=true;break;}

return p;

}

//运算定义算法函数

double Operate(double num1,char oper,double num2)

{

switch(oper)

{

case '+':return num1+num2;

case '-':return num1-num2;

case '\*':return num1\*num2;

case '/':

{

if(num2==0)

throw num2;

return num1/num2;

}

default:

{

cout<<"出现错误"<<endl;

exit(0);

}

}

}

int main()

{

//freopen("in.txt","r",stdin);

int i;

char exprition[100];

bool exit=false;;

while(1) //注意：用fget读入的时候会读取一行，换句话说会有换行符。

{

cout<<"请您输入一个合法表达式，以#结束表达式的输入，若其中出现负数，请将负数写在()内。注意，结束程序请输入end"<<endl;;

cout<<"请您输入一个表达式:";

scanf("%s",exprition);

if(exprition[0]=='e'&&exprition[1]=='n'&&exprition[2]=='d')

{cout<<"感谢您的使用!"<<endl;return 0;}

while(!Match(exprition))

{

cout<<"您输入的表达式有误，请您重新输入。"<<endl;

scanf("%s",exprition);

if(exprition[0]=='e'&&exprition[1]=='n'&&exprition[2]=='d')

{cout<<"感谢您的使用!"<<endl;return 0;}

}

exprition[strlen(exprition)]='\0';

optr.Push('#');

for(i=0;i<strlen(exprition);i++)

{

if(exprition[i]>='0'&&exprition[i]<='9'||exprition[i]=='.')

opnd.Push(exprition[i]);

else

{

if(In(exprition[i])&&!opnd.Empty())

oprd.Push(Combine());

if(Precede(optr.GetTop(),exprition[i])==-1)

{

if(exprition[i]=='-'&&exprition[i-1]=='(')

{opnd.Push(exprition[i]);continue;}

optr.Push(exprition[i]);

}

else if(Precede(optr.GetTop(),exprition[i])==1)

{

if(!In(exprition[i])&&!opnd.Empty())

oprd.Push(Combine());

double num1,num2;

char oper;

num1=oprd.GetTop();

oprd.Pop();

if(oprd.Empty())

{cout<<"表达式的值为："<<num1<<endl;continue;}

num2=oprd.GetTop();

oprd.Pop();

oper=optr.GetTop();

optr.Pop();

try{

oprd.Push(Operate(num2,oper,num1));

}

catch(double)

{

cout<<"您的数学老师还活着吗，除数不能为0。"<<endl;

break;

}

if(exprition[i]!='#'&&exprition[i]!=')')

optr.Push(exprition[i]);

while(exprition[i]==')'&&optr.GetTop()=='(')

optr.Pop();

if(exprition[i]=='#'&&optr.GetTop()!='#')

i--;

}

else if(Precede(optr.GetTop(),exprition[i])==0)

{

if(optr.GetTop()=='('&&exprition[i]==')')

optr.Pop();

}

}

if(optr.GetTop()=='#'&&exprition[i]=='#')

{cout<<"表达式的值为："<<oprd.GetTop()<<endl;break;}

}

cout<<"请按任意键继续..."<<endl;

getchar();getchar();

}

return 0;

}

实验心得：根据此次实验，又对C和数据结构有了很好的认识