实验二 栈的应用

应用技术 学院 计算机科学与技术 专业 1 班

学号 202033050040 姓名 李鹏程

一．实验目的

1．了解栈的定义及特点；

2．掌握顺序栈的实现；

3．掌握栈的应用。

二．实验内容

1．建立能够存放十进制整数的顺序栈；编写栈的基本函数；

2．利用栈的特性完成数制转换（十进制数转成二进制数、八进制数和十六进制数）；

3．建立能够存放字符的顺序栈，编写栈的五个基本函数；

4．利用栈的特性对输入的字符串完成是否为回文的判断；

三．实验步骤

1. （1） 程序代码

#include <stdio.h>

#include <stdlib.h>

#define Status int

#define SElemtype int

#define OVERFLOW -2

#define OK 1

#define ERROR 0

#define TRUE 1

#define FALSE 0

#define STACK\_INIT\_SIZE 100

#define STACKINCREMENT 10

typedef struct

{

SElemtype \* base;

SElemtype \* top;

int stacksize;

}SqStack;

Status InitStack(SqStack &S);

Status Push(SqStack &S,SElemtype e);

Status Pop(SqStack &S);

Status StackEmpty(SqStack &S);

void conversion();

int main(int argc, char const \*argv[])

{

conversion();

return 0;

}

Status InitStack(SqStack &S){

S.base = (SElemtype \*)malloc(STACK\_INIT\_SIZE \* sizeof(SElemtype));

if(!S.base)

exit(OVERFLOW);

S.top = S.base;

S.stacksize = STACK\_INIT\_SIZE;

return OK;

}

Status Push(SqStack &S,SElemtype e){

if(S.top - S.base >= S.stacksize){

S.base = (SElemtype \*)realloc(S.base,(S.stacksize + STACKINCREMENT) \* sizeof(SElemtype));

S.top = S.base + S.stacksize;

S.stacksize += STACKINCREMENT;

}

\*S.top++ = e;

return OK;

}

Status Pop(SqStack &S,SElemtype &e){

if(S.top == S.base){

printf("栈空");

return ERROR;

}

e = \*--S.top;

return OK;

}

Status StackEmpty(SqStack &S){

if (S.top - S.base == 0)

return TRUE;

else

return FALSE;

}

void conversion(){

SqStack S;

SElemtype num,flag;

InitStack(S);

printf("输入十进制整数");

scanf("%d",&num);

printf("选择转换进制:\n");

printf("1.二进制\n");

printf("2.八进制\n");

printf("3.十六进制\n");

L1: scanf("%d",&flag);

if(flag == 1 || flag == 2 || flag == 3){

switch(flag){

case 1:

while(num){

Push(S,num % 2);

num /= 2;

}

break;

case 2:

while(num){

Push(S,num % 8);

num /= 8;

}

break;

case 3:

while(num){

Push(S,num % 16);

num /= 16;

}

break;

}

}else{

printf("请重新选择进制:");

goto L1;

}

while(!StackEmpty(S)){

Pop(S,flag);

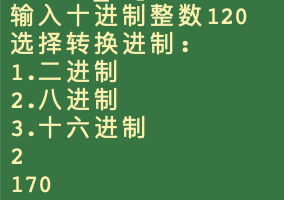
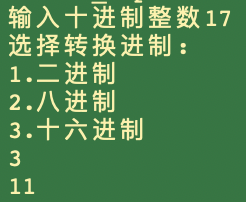
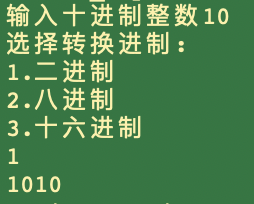
printf("%d",flag);

}

}

（2）运行程序

（3）运行结果（截图）



1. （1） 程序代码

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define SElemtype char

#define Status int

#define OVERFLOW -2

#define OK 1

#define ERROR 0

#define STACK\_INIT\_SIZE 100

#define STACKINCREMENT 10

#define N 100

typedef struct

{

SElemtype \* top;

SElemtype \* base;

int stacksize;

}SqStack;

Status InitStack(SqStack &S);

Status Push(SqStack &S,SElemtype e);

Status Pop(SqStack &S,SElemtype &e);

Status EmptyStack(SqStack &S);

void palindromics();

int main(int argc, char const \*argv[])

{

palindromics();

return 0;

}

//栈的初始化

Status InitStack(SqStack &S){

S.base = (SElemtype \*)malloc(STACK\_INIT\_SIZE \* sizeof(SElemtype));

if(!S.base)

exit(OVERFLOW);//存储分配失败

S.top = S.base;

S.stacksize = STACK\_INIT\_SIZE;

return OK;

}

//进栈

Status Push(SqStack &S,SElemtype e){

if(S.top - S.base >= S.stacksize){

S.base = (SElemtype \*)realloc(S.base,(S.stacksize + STACKINCREMENT) \* sizeof(SElemtype));

S.top = S.base + S.stacksize;

S.stacksize += STACKINCREMENT;

}

\* S.top++ = e;

return OK;

}

//出栈

Status Pop(SqStack &S,SElemtype &e){

if(S.top == S.base){

printf("栈空");

return ERROR;

}

e = \*--S.top;

return OK;

}

//栈空判断

Status EmptyStack(SqStack &S){

if(S.top - S.base == 0)

return OK;

else

return ERROR;

}

//栈顶获取

Status GetTop(SqStack &S,SElemtype e){

if(S.top == S.base)

return ERROR;

e = \*(S.top - 1);

return OK;

}

//回文判断

void palindromics(){

SqStack S;

InitStack(S);

char str[N],rts[N];

int flag;

printf("请输入字符串:");

scanf("%s",str);

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

{

Push(S,str[i]);

}

while(!EmptyStack(S)){

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

{

Pop(S,rts[i]);

}

}

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

{

if(str[i] == rts[i])

continue;

else

printf("不是回文");

flag = -1;

break;

}

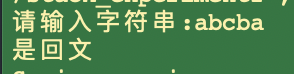
if(flag == 1)

printf("是回文");

}

（2）运行程序

（3）运行结果（截图）

截屏2022-10-25 10.36.44

四．实验小结

1.基本掌握栈的基本操作

2.理解栈的结构原理，了解其应用场景，并通过这一结构解决部分问题

3.本次实验的两个程序都是栈的应用，利用栈对进制转换余数进行存储以及对回文的存储判断，加深了对栈的理解和应用能力