实验二 栈

详见educoder

第一关：

本关任务：实现栈基本操作，并使用栈基本操作实现十进制数到二进制数的转换。

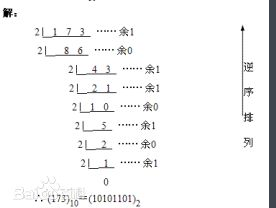
**相关知识**

为了完成本关任务，你需要掌握：

1. 栈的基本操作，包括：初始化栈、入栈、出栈、判定栈空、判定栈满、取栈顶元素等；
2. 十进制数（只考虑整数）转换为其他进制数的转换规则。

**十进制数转换为二进制数的规则**

采用除2取余法。如：



**如何存储和输出二进制数**

从数制转换示例可以看出，十进制数转换成相应二进制数**可使用栈存放，再依次弹出栈顶元素输出**，即可得到相应二进制结果。

**编程要求**

在右侧编辑器补充代码，完成十进制转二进制的转换。

**测试说明**

平台会对你编写的代码进行测试：

测试输入：173 预期输出：10101101

测试输入：5 预期输出：101

#ifndef \_STACK\_H\_

#define \_STACK\_H\_

#include <iostream>

using namespace std;

typedef int DataType;

//采用链式栈

struct node{

   DataType element;  //数据元素

   struct node \*next; //指向下一个数据元素的指针

};

typedef struct node \*PtrToNode;

typedef struct node \* Stack;

/\*

   函数名：isEmpty

   函数功能：判定栈是否为空

   函数参数：栈头结点指针

   返回值：若栈为空，则返回1，否则返回0

\*/

int isEmpty(Stack s)

{

    //此处请填写代码，用于实现判定栈是否为空的功能

    /\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/

    /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

/\*

   函数名：createStack

   函数功能：创建一个空栈，实际上只需要初始化栈头结点

   函数参数：无

   返回值：栈头结点指针

\*/

Stack createStack(void)

{

    //此处请填写代码，用于创建一个空的链栈

    /\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/

    /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

/\*

   函数名：push

   函数功能：向栈中压人一个数据元素值为x

   函数参数：待压栈的数据元素，栈头结点指针

   返回值：无

\*/

void push(DataType  x,Stack s)

{

    //表头作为栈顶，此处请填写代码实现压栈功能

    /\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/

    /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

/\*

   函数名：pop

   函数功能：弹出栈顶元素并返回元素值

   函数参数：栈头结点指针

   返回值：栈顶元素的值

\*/

DataType pop(Stack s)

{

   //此处请填写代码实现出栈并返回栈顶元素的功能，若栈为空，则返回-1

   /\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/

   /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

/\*

   函数名：convert10to2

   函数功能：将十进制数x转换为二进制并输出

   函数参数：十进制数x

   返回值：无

\*/

void convert10to2(int x)

{

    //此处请填写代码实现将参数x转换为二进制并输出的功能，请使用上述栈的基本操作实现

    /\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/

    /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

#endif

#include "stack.h"

int main(void)

{

    int x;

    cin>>x;

    convert10to2(x);

}

注意：

1. 本题中需要自己实现一个栈，从题中看，需要实现的是链栈。
2. 算法较为简单

第二关

#### 任务描述

本关任务：编写一个能将中缀表达式转换为后缀表达式的程序。

#### 相关知识

为了完成本关任务，你需要掌握：

1. 中缀表达式转换为后缀表达式的基本原理
2. 如何利用栈实现中缀表达式到后缀表达式的转换。

##### 中缀表达式与后缀表达式

中缀表达式：所有运算符出现在它的两个运算分量之间，如：

31\*（5-22）+70

后缀表达式：所有运算符出现在它的两个运算分量后面，如：

31 5 22 - \* 70 +

先将#压入栈中，然后上面式子的输出顺序以及栈中变化为

31：输出31，栈中元素为（假设左边为栈底）#

\*：\*与栈顶元素比较，比栈顶元素优先级大，则\*入栈，栈中元素为（假设左边为栈底）#\*

（：（入栈，栈中元素为（假设左边为栈底）#\*（

5：输出5，栈中元素为（假设左边为栈底）#\*（

-：-与栈顶元素比较，比栈顶元素优先级大，则-入栈，栈中元素为（假设左边为栈底）#\*（-

22：输出22，栈中元素为（假设左边为栈底）#\*（-

）：-（依次出栈，仅打印-，栈中元素为（假设左边为栈底）#\*

+：+与栈顶元素比较，比栈顶元素优先级小，则一直出栈到+为老大为止，即出栈并打印\*，之后+入栈，栈中元素为（假设左边为栈底）#+

70：输出70，栈中元素为（假设左边为栈底）#+

表达式结束，将栈中除#外，全部元素输出，此时输出+

中缀表达式到后缀表达式的转换方法：

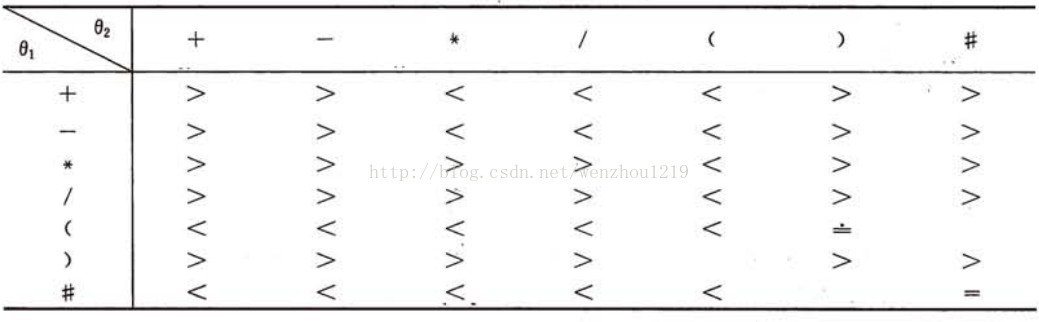
初始化一个栈，用于存放运算符，在栈中压人#，#优先级最低。

从左到右遍历中缀表达式的每个数字和符号，若是数字，则输出，成为后缀表达式的一部分，若是符号：

（1）若是左括号，直接压入栈；

（2）若是右括号，则从符号栈弹栈输出，直至**弹出左括号**（左右括号不输出）

（3）如果遇到任何其他的操作符，从栈中弹出元素直到遇到更低优先级的元素(或者栈中只有#)为止**（注意：这个更低优先级的元素此时不会被出栈，而左括号的优先级低，具体见下方表格）**。弹出这些元素后，再将遇到的操作符压入到栈中。



（4）如果我们读到了中缀表达式输入的末尾，则弹出栈中所有元素依次输出。

**有一点需要注意，只有在遇到“)”的情况下我们才弹出“(”，其他情况我们都不会弹出” ( “。**

#### 编程要求

在右侧编辑器补充代码，将中缀表达式转换为后缀表达式。

#### 测试说明

平台会对你编写的代码进行测试：

测试输入：31\*（5-22）+70 预期输出：31 5 22 - \* 70 +

测试输入：52+3\*21-32/2 预期输出：52 3 21 \* + 32 2 / -

-----------------------------main.cpp----------------------------------------------

#include "stack\_symbol.h"

int main(void)

{

    char express[80];

    cin>>express ;

    inToPost(express);

}

 -------------------stack\_symbol.h------------------------

#ifndef \_STACK\_H\_

#define \_STACK\_H\_

#include <stdlib.h>

#include <iostream>

using namespace std;

typedef char DataType;

//采用链式栈

struct node{

    DataType element;  //数据元素

    struct node \*next; //指向下一个数据元素的指针

};

typedef struct node \*PtrToNode;

typedef PtrToNode Stack;

/\*

   函数名：isEmpty

   函数功能：判定栈是否为空

   函数参数：栈头结点指针

   返回值：若栈为空，则返回1，否则返回0

\*/

int isEmpty(Stack s)

{

    return  s->next ==NULL;

}

/\*

   函数名：createStack

   函数功能：创建一个空栈，实际上只需要初始化栈头结点

   函数参数：无

   返回值：栈头结点指针

\*/

Stack createStack(void)

{

    Stack s ;

    s =new struct node;

    s->next = NULL;

    s->element =0;

    return s;

}

/\*

   函数名：push

   函数功能：向栈中压人一个数据元素值为x

   函数参数：待压栈的数据元素，栈头结点指针

   返回值：无

\*/

void push(DataType  x,Stack s)

{

    //表头作为栈顶

    PtrToNode temp ;

    temp=new struct node;

    temp->element = x;

    temp->next = s->next;

    s->next = temp;

}

/\*

   函数名：pop

   函数功能：弹出栈顶元素并返回元素值

   函数参数：栈头结点指针

   返回值：栈顶元素的值

\*/

DataType pop(Stack s)

{

    PtrToNode temp;

    int t;

    if(isEmpty(s)==0)

    {

        temp = s->next;

        t = temp->element;

        s->next = temp->next;

        free(temp);

         return t;

    }

}

DataType top(Stack s)

{

    if(isEmpty(s)==0)

    {

        return s->next->element ;

    }

}

/\*

   函数名：inToPost

   函数功能：将中缀表达式转换为后缀表达式输出

   函数参数：中缀表达式，放在字符数组中

   返回值：无

\*/

void inToPost(char \*expression)

{

   //在此处填写代码，完成中缀表达式转换为后缀表达式并输出

   /\*\*\*\*\*\*\*\*\*\*  Begin  \*\*\*\*\*\*\*\*\*\*/

   /\*\*\*\*\*\*\*\*\*\*  End  \*\*\*\*\*\*\*\*\*\*/

}

#endif

注意：

1. main函数中，最后一行自行加上return 0;
2. 理解文档中栈的实现，并会用，最好是能自己实现一个栈，文档中的栈是用链表实现的。栈里面存放的是字符。
3. 需要看懂中缀表达式转后缀表达式的算法，然后编写代码实现。分多种情况。具体算法和情况见上面说明。思考运算符优先级比较方法，如何通过表格判断。

