#include<stdio.h>

#include<stdlib.h>

typedef enum{lparin,rparin,plus,minus,times,divide,mod,eos,operand}precedence;

char expr[100];

int x=1;

int top=-1;

int isp[]={0,19,12,12,13,13,13,0};

int icp[]={20,19,12,12,13,13,13,0};

int avl=0;

typedef struct

{

int key;

}term;

term \*element;

void create()

{

element=(term\*)malloc(sizeof(element));

}

void stackfull()

{

x=x\*2;

element=realloc(element,x\*sizeof(element));

}

int pop()

{

if (top==-1)

{

return -1;

}

else{

return element[top--].key;

}

}

void push(int item)

{

if (top>=(x-1))

{

stackfull();

element[++top].key=item;

}

else

{

element[++top].key=item;

}

}

precedence gettoken(char\*symbol,int\*n)

{

\*symbol=expr[(\*n)++];

switch(\*symbol)

{

case '(':return lparin;

case ')':return rparin;

case '+':return plus;

case '-':return minus;

case '/':return divide;

case '\*':return times;

case '%':return mod;

case '\0':return eos;

default:return operand;

}

}

void printtoken(precedence token)

{

switch(token)

{

case plus:printf("+");break;

case minus:printf("-");break;

case divide:printf("/");break;;

case times:printf("\*");break;

case mod:printf("%%");break;

}

}

void change()

{

char symbol;

precedence token;

int n=0;

top=-1;

token=eos;

push(token);

for (token=gettoken(&symbol,&n);token!=eos;token=gettoken(&symbol,&n))

{

if (token==operand)

{

printf("%c",symbol);

}

else if(token==rparin)

{

while((element[top].key)!=lparin)

printtoken(pop());

pop();

}

else{

while((isp[element[top].key])>=icp[token])

printtoken(pop());

push(token);

}

}

while((token=pop())!=eos)

{

printtoken(token);

}

}

void main()

{ printf("Enter the Infix Expression:\n");

gets(expr);

create();

change();

}