#include<conio.h>

#include<stdio.h>

#include<stdlib.h>

int getOperatorPosition(char );

#define node struct tree1

int matrix[5][5]={

{1,0,0,1,1},

{1,1,0,1,1},

{0,0,0,2,3},

{1,1,3,1,1},

{0,0,0,3,2}};

int tos=-1;

void matrix\_value(void);

//node create\_node(char,\*node);void show\_tree( node \*);

int isOperator(char );

struct tree1

{

char data;

node \*lptr;

node \*rptr;

}\*first;

struct opr

{

char op\_name;

node \*t;

}oprate[50];

char cur\_op[5]={'+','\*','(',')','['};

char stack\_op[5]={'+','\*','(',')',']'};

void main()

{

char exp[10];

int ssm=0,row=0,col=0;

node \*temp;

// clrscr();

printf("Enter Exp : ");

scanf("%s",exp);

matrix\_value();

while(exp[ssm] != '\0')

{

if(ssm==0)

{

tos++;

oprate[tos].op\_name = exp[tos];

}

else

{

if(isOperator(exp[ssm]) == -1)

{

oprate[tos].t = (node\*) malloc (sizeof(node));

oprate[tos].t->data = exp[ssm];

oprate[tos].t->lptr = '\0';

oprate[tos].t->rptr = '\0';

}

else

{

row = getOperatorPosition(oprate[tos].op\_name);

col = getOperatorPosition(exp[ssm]);

if(matrix[row][col] == 0)

{

tos++;

oprate[tos].op\_name = exp[ssm];

}

elseif(matrix[row][col] == 1)

{

temp = (node\*) malloc (sizeof(node));

temp->data = oprate[tos].op\_name;

temp->lptr = (oprate[tos-1].t);

temp->rptr = (oprate[tos].t);

tos--;

oprate[tos].t = temp;

ssm--;

}

elseif(matrix[row][col] == 2)

{

//temp = (node\*) malloc (sizeof(node));

temp = oprate[tos].t;

tos--;

oprate[tos].t = temp;

}

elseif(matrix[row][col] == 3)

{

printf("\nExpression is Invalid...\n");

printf("%c %c can not occur simultaneously\n",oprate[tos].op\_name,exp[ssm]);

break;

}

}

}

ssm++;

}

printf("show tree \n\n\n");

show\_tree(oprate[tos].t);

printf("Over");

getch();

getch();

}

int isOperator(char c)

{

int i=0;

for(i=0;i<5;i++)

{

if (c==cur\_op[i] || c==stack\_op[i])

break;

}

if(i==5)

return (-1);

elsereturn i;

}

int getOperatorPosition(char c)

{

int i;

for(i=0;i<5;i++)

{

if (c==cur\_op[i] || c==stack\_op[i])

break;

}

return i;

}

void show\_tree(node \*start)

{

if(start->lptr != NULL)

show\_tree(start->lptr);

if(start->rptr != NULL)

show\_tree(start->rptr);

printf("%c \n",start->data);

}

void matrix\_value(void)

{

int i,j;

printf("OPERATOR PRECEDENCE MATRIX\n");

printf("===========================\n ");

for(i=0; i<5; i++)

{

printf("%c ",stack\_op[i]);

}

printf("\n");

for(i=0;i<5;i++)

{

printf("%c ",cur\_op[i]);

for(j=0;j<5;j++)

{

if(matrix[i][j] == 0)

printf("< ");

elseif(matrix[i][j] == 1)

printf("> ");

elseif(matrix[i][j] == 2)

printf("= ");

elseif(matrix[i][j] == 3)

printf(" ");

}

printf("\n");

}

}