/\*

240327 스택 전위, 후위, 표기식

\*/

//

#include <stdio.h>

#include <stdlib.h>

#define MAX\_STACK\_SIZE 100

// 프로그램 “일반적인 배열 스택 프로그램”에서 스택 코드 추가

typedef char element; // 교체!

// ...

// 프로그램 “일반적인 배열 스택 프로그램” 에서 스택 코드 추가 끝

// 후위 표기 수식 계산 함수

int eval(char exp[])

{

int op1, op2, value, i = 0;

int len = strlen(exp);

char ch;

StackType s;

init\_stack(&s);

for (i = 0; i<len; i++) {

ch = exp[i];

if (ch != '+' && ch != '-' && ch != '\*' && ch != '/') {

value = ch - '0'; // 입력이 피연산자이면

push(&s, value);

}

else { //연산자이면 피연산자를 스택에서 제거

op2 = pop(&s);

op1 = pop(&s);

switch (ch) { //연산을 수행하고 스택에 저장

case '+': push(&s, op1 + op2); break;

case '-': push(&s, op1 - op2); break;

case '\*': push(&s, op1 \* op2); break;

case '/': push(&s, op1 / op2); break;

} } }

return pop(&s);

}

int main(void)

{

int result;

printf("후위표기식은 82/3-32\*+\n");

result = eval("82/3-32\*+");

printf("결과값은 %d\n", result);

return 0;

}

#include <stdio.h>

#include <stdlib.h>

#define MAX\_STACK\_SIZE 100

// 프로그램 “일반적인 배열 스택 프로그램” 에서 스택 코드 추가

typedef char element; // 교체!

// ...

// 프로그램 “일반적인 배열 스택 프로그램” 에서 스택 코드 추가 끝

// 연산자의 우선순위를 반환한다.

int prec(char op)

{

switch (op) {

case '(': case ')': return 0;

case '+': case '-': return 1;

case '\*': case '/': return 2;

}

return -1;

}

// 중위 표기 수식 -> 후위 표기 수식

void infix\_to\_postfix(char exp[])

{

int i = 0;

char ch, top\_op;

int len = strlen(exp);

StackType s;

init\_stack(&s); // 스택 초기화

for (i = 0; i<len; i++) {

ch = exp[i];

switch (ch) {

case '+': case '-': case '\*': case '/': // 연산자

// 스택에 있는 연산자의 우선순위가 더 크거나 같으면 출력

while (!is\_empty(&s) && (prec(ch) <= prec(peek(&s))))

printf("%c", pop(&s));

push(&s, ch);

break;

case '(': // 왼쪽 괄호

push(&s, ch);

break;

case ')': // 오른쪽 괄호

top\_op = pop(&s);

while (top\_op != '(‘) { // 왼쪽 괄호를 만날때까지 출력

printf("%c", top\_op);

top\_op = pop(&s);

}

break;

default: // 피연산자

printf("%c", ch);

break;

}

}

while (!is\_empty(&s)) // 스택에 저장된 연산자들 출력

printf("%c", pop(&s));

}

//

int main(void)

{

char \*s = "(2+3)\*4+9";

printf("중위표시수식 %s \n", s);

printf("후위표시수식 ");

infix\_to\_postfix(s);

printf("\n");

return 0;

}

//

// 프로그램 “일반적인 배열 스택 프로그램” 에서 스택 코드 추가

// ...

typedef struct { // 교체!

short r;

short c;

} element;

// 프로그램 “일반적인 배열 스택 프로그램” 에서 스택 코드 추가 끝

element here = { 1,0 }, entry = { 1,0 };

char maze[MAZE\_SIZE][MAZE\_SIZE] = {

{ '1', '1', '1', '1', '1', '1' },

{ 'e', '0', '1', '0', '0', '1' },

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

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

{ '1', '0', '1', '0', '0', 'x' },

{ '1', '1', '1', '1', '1', '1' },

};

// 위치를 스택에 삽입

void push\_loc(StackType \*s, int r, int c)

{

if (r < 0 || c < 0) return;

if (maze[r][c] != '1' && maze[r][c] != '.') {

element tmp;

tmp.r = r;

tmp.c = c;

push(s, tmp);

}

}

// 미로를 화면에 출력한다.

void maze\_print(char maze[MAZE\_SIZE][MAZE\_SIZE])

{

printf("\n");

for (int r = 0; r < MAZE\_SIZE; r++) {

for (int c = 0; c < MAZE\_SIZE; c++) {

printf("%c", maze[r][c]);

}

printf("\n");

}

}

// 미로 탐색 프로그램2

int main(void)

{

int r, c;

StackType s;

init\_stack(&s);

here = entry;

while (maze[here.r][here.c] != 'x') {

r = here.r;

c = here.c;

maze[r][c] = '.';

maze\_print(maze);

push\_loc(&s, r - 1, c);

push\_loc(&s, r + 1, c);

push\_loc(&s, r, c - 1);

push\_loc(&s, r, c + 1);

if (is\_empty(&s)) {

printf("실패\n");

return;

}

else

here = pop(&s);

}

printf("성공\n");

return 0;

}