Assignment_02

12201928 이상혁

구조 Stack.h Stack.cpp main.cpp

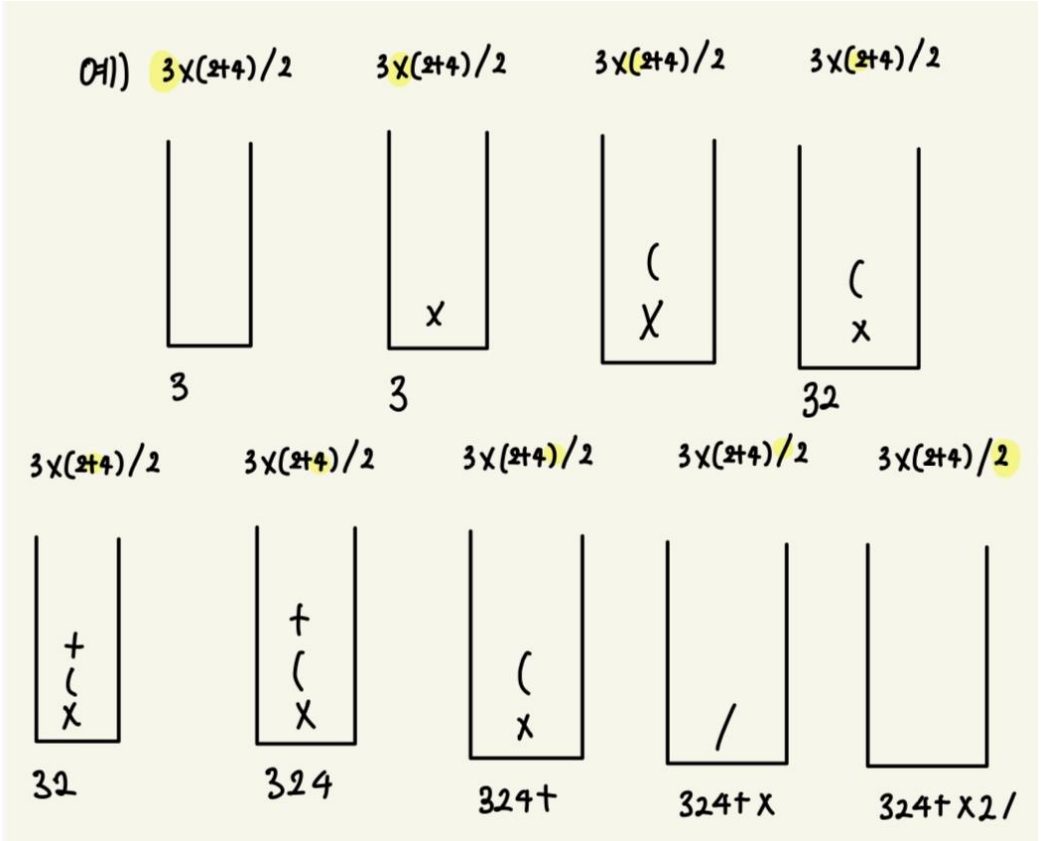
알고리즘의 설계

(1) 연산자의 우선순위 판단

(2) 중위 표기법식을 한 글자씩 읽기

피연산자인 경우 출력
연산자인 경우 스택에서 현 연산자보다 같거나 우선순위 인 것들을 출력 후 POP -> 연산자는 스택에 PUSH

' (' 인 경우 스택에 PUSH
') ' 인 경우 ' (' 가 나올 때까지 스택에서 POP 후 출력



(3) 스택에 남아 있는 연산자는 모두 POP, 출력

Assignment_02

12201928 이상혁

Stack.h

user12201928@ee2f31c5f7f7: ~/Assignment_02

```
#pragma once
#define MAX_ITEMS 100


class Stack {
private:
        int top;
        int items[MAX_ITEMS];
public:
        Stack();
        ~Stack();
        void makeEmpty();
        bool isEmpty() const;
        bool isFull()const;
        void push(int newItem);
        void pop();
        int getTop();
};
```

Stack.cpp

user12201928@ee2f31c5f7f7: ~/Assignment_02

```
#include "Stack.h"

Stack::Stack()
{
        top = -1;
}

Stack::~Stack()
{
}

void Stack::makeEmpty()
{
        top = -1;
}

bool Stack::isEmpty() const
{
        return (top == -1);
}
bool Stack::isFull() const
{
        return (top == MAX_ITEMS - 1);
}

void Stack::push(int newItem)
{
        top++;
        items[top] = newItem;
}

void Stack::pop()
{
        top--;
}

int Stack::getTop()
{
        return items[top];
}
```

Max_ITEMS = 스택에 저장할 수 있는 최대 항목수를 100으로 define

클래스 형태의 Stack 선언

private
– top 스택의 맨 위 항목의 인덱스를 나타내는 변수 (-1로 초기화 한다)
– items 스택의 데이터를 저장하는 배열
public
– Stack() 클래스의 생성자로 스택을 초기화한다. top을 -1로 설정해 빈스택을 만들게 된다.
– ~Stack() 클래스의 소멸자로 메모리 누수를 방지한다.

void makeEmpty() 생성자와 동일하게 top을 -1로 설정하여 빈 스택으로 만든다.

bool isEmpty() const 스택이 비어있는지의 여부를 반환하는 함수이며 스택의 맨위 인덱스인 top이 -1이면 스택이 비어있는 것으로 간주한다.

bool isFull() const 스택이 가득 찼는지를 확인하는 함수이며 스택의 top이 Max-items -1 과 동일한 경우 가득 찬것으로 간주한다.

void push 스택에 새 항목을 추가하는 함수이다. top을 하나 증가시키고 매개변수로 받은 새로운 항목을 맨위에 인덱스에 추가시킨다.

void pop() 스택에서 top을 제거하는 함수이다. top을 하나씩 줄이며 기능을 수행한다.

int getTop() 스택의 맨위 인덱스에 들어있는 값을 반환하는 함수이다.

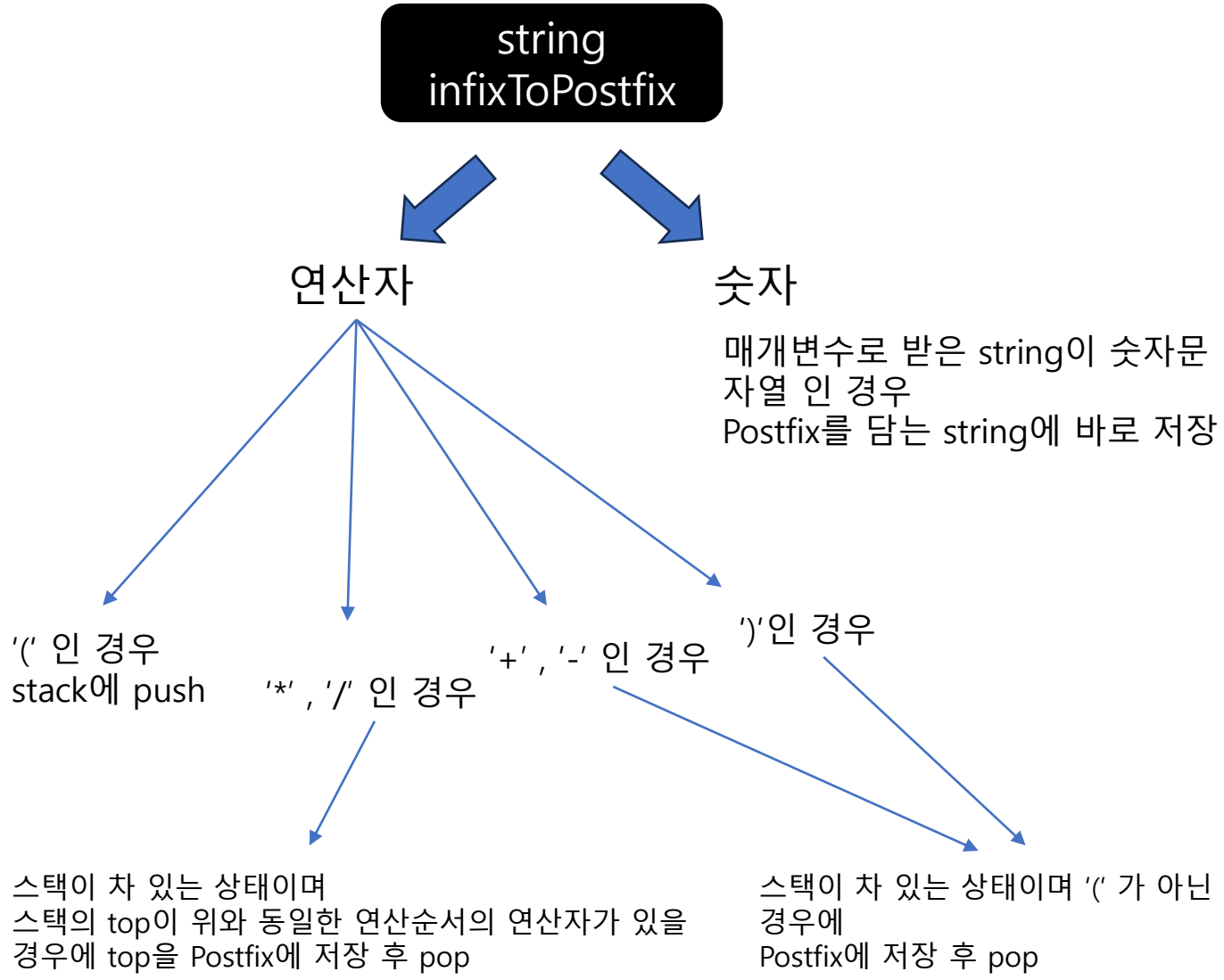
Assignment_02

12201928 이상혁

main.cpp

user12201928@ee2f31c5f7f7: ~/Assignment_02

```
#include <iostream>
#include <string>
#include "Stack.h"
using namespace std;
string infixToPostfix(string& infix) {
    Stack s;
    string Postfix;
    s.makeEmpty();
    for (char a : infix) {
        if (a >= '0' && a <= '9') {
            Postfix += a;
        }
        else {
            if (a == '(') s.push(a);
            else if (a == '*' || a == '/') {
                while (!s.isEmpty() && (s.getTop() == '*' || s.getTop() ==
{
                    Postfix += s.getTop();
                    s.pop();
                }
                s.push(a);
            }
            else if (a == '+' || a == '-') {
                while (!s.isEmpty() && s.getTop() != '(') {
                    Postfix += s.getTop();
                    s.pop();
                }
                s.push(a);
            }
            else {
                while (!s.isEmpty() && s.getTop() != '(') {
                    Postfix += s.getTop();
                    s.pop();
                }
                s.pop();
            }
        }
    }
    while (!s.isEmpty()) {
        Postfix += s.getTop();
        s.pop();
    }
```



위와 같은 과정을 거쳤음에도 스택이 차 있는 경우에는 top에 위치한 값을
Postfix에 저장 후 pop

Assignment_02

12201928 이상혁

main.cpp

```
int evaluatePostfix(string& postfix) {
    Stack s;
    char a;
    int n1, n2, result;
    for (char a : postfix){
        if (a >= '0' && a <= '9') {
            s.push(a - '0');
        }
        else if (a == '+' || a == '-' || a == '*' || a == '/'
            n2 = s.getTop();
            s.pop();
            n1 = s.getTop();
            s.pop();
            switch (a) {
            case '+':
                result = n1 + n2;
                break;
            case '-':
                result = n1 - n2;
                break;
            case '*':
                result = n1 * n2;
                break;
            case '/':
                if (n2 != 0) {
                    result = n1 / n2;
                }
                else {
                    cout << "Error divide by zero" << endl;
                    return -1;
                }
                break;
            }
            s.push(result);
        }
    }
    return s.getTop();
}
```

int evaluatePostfix

연산자

문자열 숫자를 계산이 가능하게 숫자로 변환 시킨 후 스택에 push

숫자

연산자가 나오게 되면 그 다음 두개의 숫자를 stack에서 뽑아낸다.

그 이후 연산자에 맞는 연산을 실시
-> case 문으로 구성

0으로 나누었을 땐 오류
계산을 한 값을 스택에 push 후 top을 리턴

Assignment_02

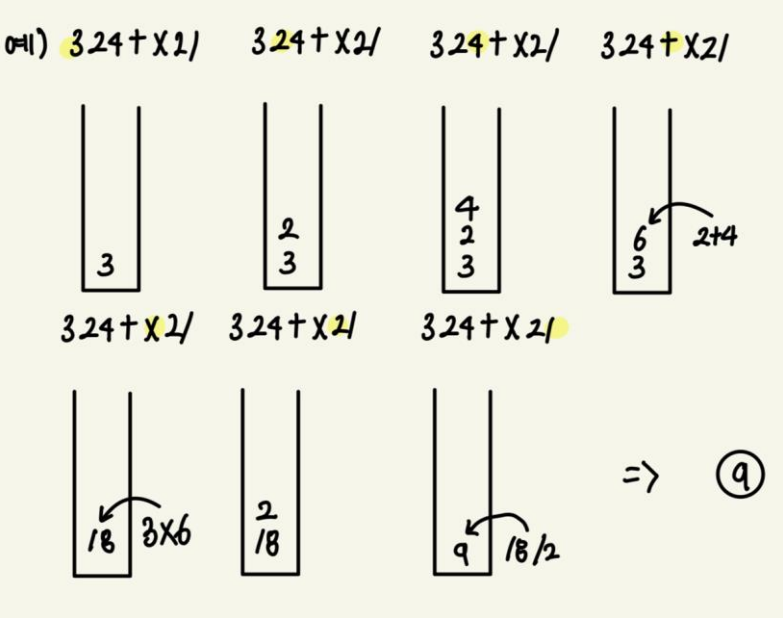
12201928 이상혁

main.cpp

Example 1

```
int main() {
    string infix = "3*(2+4)/2";
    string postfix = infixToPostfix(infix);
    cout << "Postfix: " << postfix << endl; // Postfix: 324+*2/
    cout << "Evaluation: " << evaluatePostfix(postfix) << endl; // Evaluation: 9
    return 0;
}
```

```
user12201928@ee2f31c5f7f7:~/Assignment_02$ g++ -o main.out Stack.cpp main.cpp
user12201928@ee2f31c5f7f7:~/Assignment_02$ ./main.out
Postfix: 324+*2/
Evaluation: 9
```



ex1

3*(2+4)/2 -> 324+*2/
계산결과: 9

Example 2

```
int main() {
    string infix = "1*(3+5)";
    cout << "Question: " << infix << endl;
    string postfix = infixToPostfix(infix);
    cout << "Postfix: " << postfix << endl;
    cout << "Evaluation: " << evaluatePostfix(postfix) << "\n\n";
    infix = "5+6";
    cout << "Question: " << infix << endl;
    postfix = infixToPostfix(infix);
    cout << "Postfix: " << postfix << endl;
    cout << "Evaluation: " << evaluatePostfix(postfix) << "\n\n";
    infix = "4+2*3";
    cout << "Question: " << infix << endl;
    postfix = infixToPostfix(infix);
    cout << "Postfix: " << postfix << endl;
    cout << "Evaluation: " << evaluatePostfix(postfix) << "\n\n";
    infix = "4+1*5-6/7";
    cout << "Question: " << infix << endl;
    postfix = infixToPostfix(infix);
    cout << "Postfix: " << postfix << endl;
    cout << "Evaluation: " << evaluatePostfix(postfix) << "\n\n";
```

```
user12201928@ee2f31c5f7f7:~/Assignment_02$ g++ -o main.out Stack.cpp main.cpp
user12201928@ee2f31c5f7f7:~/Assignment_02$ ./main.out
Question: 1*(3+5)
Postfix: 135+*
Evaluation: 8

Question: 5+6
Postfix: 56+
Evaluation: 11

Question: 4+2*3
Postfix: 423*+
Evaluation: 10

Question: 4+1*5-6/7
Postfix: 415*+67/-
Evaluation: 9
```

ex2

1*(3+5) -> 135+*
계산결과: 8

4+2*3 -> 423*+
계산결과: 10

5+6 -> 56+
계산결과: 11

4+1*5-6/7 -> 415*+67/-
계산결과: 9