# 자료구조 1 (참고)

최백준 choi@startlink.io

# 후위표기법

# 후위 표기법

Postfix Notation



- 중위 표기법(Infix Notation)은 일반적으로 수식을 표기할 때 사용하는 방식이다.
- 연산자를 피연산자의 사이에 두는 방식이다.
- 우리가 일반적으로 사용하는 방식이 후위 표기법이다.
- 예) 1+ 2, 3 × 4, 2 - 1

# 후위 표기법

Postfix Notation



- 후위 표기법(Postfix Notation)은 연산자를 피연산자의 다음에 두는 방식이다.
- 예) 1 2 +, 3 4 ×, 2 1 –
- 전위 표기법(Prefix Notation)은 연산자를 앞에 두는 방식이며, 폴란드 표기법(Polish Notation) 이라고 한다.
- 후위 표기법은 역폴란드 표기법(Reverse Polish Notation)이라고도 한다.

# 후위 표기법

Postfix Notation



- 중위 표기법은 사람이 보기 편리하지만, 컴퓨터가 이 수식을 처리하기 까다롭다.
- 후위 표기법을 사용하면, 컴퓨터가 수식을 특별한 변환없이 처리할 수 있다.

# 후위 표기법

Postfix Notation



- 중위 표기법 → 후위 표기법
- 1 + 2 → 1 2 +
- 3 + 2 × 3 → 3 2 3 × +
- (3 + 2) × 3 → 3 2 + 3 ×
- (3 + 2) × 3 - 4 → 3 2 + 3 × 4 -
- 3 + 2 × 3 - 4 → 3 2 3 × + 4 -
- (3 + 2) × (3 - 4) → 3 2 + 3 4 - ×
- (3 + 2) × ((6 + 3) - 4) → 3 2 + 6 3 + 4 - ×

# 후위 표기법

Postfix Notation



- 후위 표기법으로 표현된 식을 계산하는 방법은 다음과 같다.

1. 피연산자는 스택에 넣는다
2. 연산자를 만나면 피연산자 2개를 스택에서 꺼내 계산하고, 계산된 결과를 다시 스택에 넣는다

# 후위 표기법

Postfix Notation

- 3 2 + 6 3 + 4 - ×

| 연산자/피연산자 | 스택 | 비고 |
|---|---|---|
| 3 | 3 | |
| 2 | 3, 2 | |
| + | 5 | 3 + 2 |
| 6 | 5, 6 | |
| 3 | 5, 6, 3 | |
| + | 5, 9 | 6 + 3 |
| 4 | 5, 9, 4 | |
| - | 5, 5 | 9 – 4 |
| × | 25 | 5 × 5 |

# 후위 표기식2



https://www.acmicpc.net/problem/1935

- 후위 표기법으로 표현된 수식을 계산하는 문제

# 후위 표기식2



https://www.acmicpc.net/problem/1935

- 소스: http://codeplus.codes/862d47dc373249ab9e9596dd0fd89a8d

# 차량기지 알고리즘

Shunting-yard Algorithm



- 중위 표기법으로 표현된 식을 후위 표기법으로 바꾸는 알고리즘이다.
- 연산자를 저장하는 스택을 기반으로 이루어져 있다.

# 차량기지 알고리즘

Shunting-yard Algorithm

- 3 + 2 × 3 – 4
- 연산자의 우선순위가
- 스택의 가장 위에 있는 연산자의 우선순위보다
- 작거나 같은 동안
- 스택에 있는 연산자를 결과에 추가한다

| 연산자/피연산자 | 연산자 스택 | 결과 |
|---|---|---|
| 3 | | 3 |
| + | + | 3 |
| 2 | + | 3 2 |
| × | + × | 3 2 |
| 3 | + × | 3 2 3 |
| - | + | 3 2 3 × |
| | | 3 2 3 × + |
| | - | 3 2 3 × + |
| 4 | - | 3 2 3 × + 4 |
| | | 3 2 3 × + 4 - |

- 모든 연산자/피연산자 처리가 끝나면, 연산자 스택에 있는 연산자를 하나씩 결과에 추가한다.

# 차량기지 알고리즘

Shunting-yard Algorithm



- 괄호가 있는 식의 경우에는
- 여는 괄호는 연산자 스택에 넣고
- 닫는 괄호가 나오면 여는 괄호가 나올 때까지 연산자 스택에서 계속해서 연산자를 꺼낸다.

# 차량기지 알고리즘

Shunting-yard Algorithm

- 3 + 2 × 5 ÷ (3 × 5 – 4) + 1

| 연산자/피연산자 | 연산자 스택 | 결과 |
|---|---|---|
| 3 | | 3 |
| + | + | 3 |
| 2 | + | 3 2 |
| × | + × | 3 2 |
| 5 | + × | 3 2 5 |
| ÷ | + | 3 2 5 × |
| | + ÷ | 3 2 5 × |
| ( | + ÷ ( | 3 2 5 × |
| 3 | + ÷ ( | 3 2 5 × 3 |
| × | + ÷ ( × | 3 2 5 × 3 |

| 연산자/피연산자 | 연산자 스택 | 결과 |
|---|---|---|
| 5 | + ÷ ( × | 3 2 5 × 3 5 |
| - | + ÷ ( | 3 2 5 × 3 5 × |
| 4 | + ÷ ( - | 3 2 5 × 3 5 × 4 |
| ) | + ÷ ( | 3 2 5 × 3 5 × 4 - |
| | + ÷ | 3 2 5 × 3 5 × 4 - |
| + | + | 3 2 5 × 3 5 × 4 - ÷ |
| | | 3 2 5 × 3 5 × 4 - ÷ + |
| | + | 3 2 5 × 3 5 × 4 - ÷ + |
| 1 | + | 3 2 5 × 3 5 × 4 - ÷ + 1 |
| | | 3 2 5 × 3 5 × 4 - ÷ + 1 + |

# 후위 표기식



https://www.acmicpc.net/problem/1918

- 중위 표기식을 후위 표기식으로 변경하는 문제

# 후위 표기식



https://www.acmicpc.net/problem/1918

- 소스: http://codeplus.codes/15d369e31682444f96645d4f37739885

# 문자열

# 아스키코드



ASCII

- 문자 인코딩 방법
- 외울 필요는 없다.
- 대표적인 아스키 코드
- '0' => 48
- 'A' => 65
- 'a' => 97
- 0은 아스키 코드로는 NULL을 나타낸다.
- 숫자가 저장되어있는데, 출력만 글자로 해주는 것으로 이해하면 편하다.

# 아스키코드



ASCII

```
printf("%c",65);
printf("%c",48);
```

- 위의 코드의 실행 결과는 A0 이다.

# 알파벳 개수



https://www.acmicpc.net/problem/10808

- 알파벳 소문자로 이루어진 단어에서 각 알파벳이 몇 개인지 구하는 문제
- 소스: http://codeplus.codes/a8e424a753924f60acabf812b6c86751

# 알파벳 찾기

https://www.acmicpc.net/problem/10809

- 알파벳 소문자로 이루어진 단어에서 각 알파벳이 몇 번째에 처음 등장하는지 찾는 문제
- 소스: http://codeplus.codes/6002d5f1b043437580cc792b27b9d2af

# 문자열 분석



https://www.acmicpc.net/problem/10820

- 문자열 N개에 포함되어 있는 소문자, 대문자, 숫자, 공백의 개수를 세는 문제
- 소스: http://codeplus.codes/a874d61b6cd74e599bb5a774104542e5

# 단어 길이 재기



https://www.acmicpc.net/problem/2743

- 단어를 입력받고 길이를 재는 문제
- strlen이나 string의 length나 size를 이용하면 되지만 이런 것을 사용할 수 없는 경우에는
- 다음과 같이 길이를 잴 수 있다.

```
scanf("%s",s);
int len = 0;
for (int i=0; s[i]; i++) {
    len += 1;
}
printf("%d\n",len);
```

# 단어 길이 재기



https://www.acmicpc.net/problem/2743

- strlen 함수의 시간 복잡도는 O(N) 이기 때문에, 다음과 같이 작성하면 O(N^2) 코드이다.

```
for (int i=0; i<strlen(s); i++) {
    // Do something
}
```

- 아래와 같이 작성하는 것이 올바르다.

```
int len = strlen(s);
for (int i=0; i<len; i++) {
    // Do something
}
```

# 단어 길이 재기



https://www.acmicpc.net/problem/2743

- 소스: http://codeplus.codes/9a5af0fab1e749ce9ffbd1a0f5ad580b

# ROT13

https://www.acmicpc.net/problem/11655

- ROT13으로 암호화하는 프로그램을 만드는 문제
- 소스: . http://codeplus.codes/5fee489925594c4ab3c326e0f7f040d6

# 문자열 -> 정수



stoi, stol, stoll

- C++ string을 문자로 바꾸려면 stoi, stol, stoll 등등의 함수를 사용하면 된다.
- stoi: string -> int
- stol: string -> long
- stoll: string -> long long
- stof: string -> float
- stod: string -> double
- stold: string -> long double
- stoul: string -> unsigned long
- stoull: string -> unsigned long long

# 정수 -> 문자열



to_string

- to_string 함수를 사용하면 된다.

# 네 수

https://www.acmicpc.net/problem/10824

- 네 자연수 A, B, C, D가 주어진다. 이 때, A와 B를 붙인 수와 C와 D를 붙인 수의 합을 구하는 문제
- 소스: http://codeplus.codes/3f32a097f4f74e8b91be9a1e48639c5c

# 접미사 배열

https://www.acmicpc.net/problem/11656

- 접미사 배열은 문자열 S의 모든 접미사를 사전순으로 정렬해 놓은 배열이다.
- baekjoon의 접미사는 baekjoon, aekjoon, ekjoon, kjoon, joon, oon, on, n 으로 총 8가지가 있고, 이를 사전순으로 정렬하면, aekjoon, baekjoon, ekjoon, joon, kjoon, n, on, oon이 된다.
- 문자열 S가 주어졌을 때, 모든 접미사를 사전순으로 정렬한 다음 출력하는 프로그램을 작성하시오.

# 접미사 배열



https://www.acmicpc.net/problem/11656

- 문자열의 부분 문자열은 substr를 이용해서 구할 수 있다.
- 소스: http://codeplus.codes/1d067f56ae8341aa8ab9c4ad1f8260b0

# 끝

# 코드 플러스

https://code.plus

- 슬라이드에 포함된 소스 코드를 보려면 "정보 수정 > 백준 온라인 저지 연동"을 통해 연동한 다음, "백준 온라인 저지"에 로그인해야 합니다.
- 강의 내용에 대한 질문은 코드 플러스의 "질문 게시판"에서 할 수 있습니다.
- 문제와 소스 코드는 슬라이드에 첨부된 링크를 통해서 볼 수 있으며, "백준 온라인 저지"에서 서비스됩니다.
- 슬라이드와 동영상 강의는 코드 플러스 사이트를 통해서만 볼 수 있으며, 동영상 강의의 녹화와 다운로드, 배포와 유통은 저작권법에 의해서 금지되어 있습니다.
- 다른 경로로 이 슬라이드나 동영상 강의를 본 경우에는 codeplus@startlink.io 로 이메일 보내주세요.
- 강의 내용, 동영상 강의, 슬라이드, 첨부되어 있는 소스 코드의 저작권은 스타트링크와 최백준에게 있습니다.