

## 2021-1학기 전공튜터링 운영결과보고서

학 과	응용소프트웨어공학과			
과 목 명	데이터구조			
참여학생	튜터	응용소프트웨어 공학과 황진주	튜티3	응용소프트웨어 공학과 최범진
	튜티1	응용소프트웨어 공학과 임미선	튜티4	응용소프트웨어 공학과 이상준
	튜티2	응용소프트웨어 공학과 이지현	튜터5	x



## 2021-1학기 전공튜터링 운영결과보고서

- 제출장소: DOOR 전공튜터링 멘토링 게시판
- 제출서류: 운영결과보고서, 학습 일지 7부, 요약 정리 7부
- 제출기한: 2021.6.30.(수) 16:00 까지

튜터	학과: 응용소프트웨어공학과 학번: 20193148 성명: 황진주		
튜티	1	학과: 응용소프트웨어공학과 학번: 20203218 성명: 임미선	
	2	학과: 응용소프트웨어공학과 학번: 20203143 성명: 이지현	
	3	학과: 응용소프트웨어공학과 학번: 20183177 성명: 최범진	
	4	학과: 응용소프트웨어공학과 학번: 20183210 성명: 이상준	
	5	학과: x 학번: 성명:	
학습기간	2021년 4월 5일 ~ 2021년 6월 30일 (2.5시간 × 8회 = 20시간)		
학습장소	Discord 온라인 수업		
학습내용	이수구분: (체크 <input checked="" type="checkbox"/> 전공심화 <input type="checkbox"/> , 전공핵심 <input checked="" type="checkbox"/>	교과목번호: 400650	교과목명: 데이터구조

위와 같이 2021-1학기 전공 튜터링 프로그램 운영 결과보고서를 제출합니다.

2021년 06월 05일

튜터링 대표 튜터: 황진주 (인)

황진주

동의대학교 교수학습개발센터장 귀하

## 1. 팀 소개

학과 생활의 전반적인 지원과 개개인의 교과 능력 증대를 위한 교과동아리 DoD의 동아리 회원들의 모임이다. 다른 특성을 지닌 학생들이 다 함께 모르는 점을 공유하고, 알려주며 상호간의 발전을 위해 협력을 도모함을 목적으로 하고 있다.

## 2. 출석현황

	1회차	2회차	3회차	4회차	5회차	6회차	7회차	8회차
운영일시	05월 04일 20시-22시	05월 11일 20시-22시	05월 18일 20시-22시	05월 25일 20시-22시	06월 01일 20시-22시	06월 01일 20시-22시	06월 03일 20시-22시	06월 08일 20시-22시
총인원	5	5	5	5	4	5	5	5
참석인원	5	5	5	5	4	5	5	5
결석명단	0	0	0	0	1	0	0	0

## 3. 튜터링 진행 방법

**진행장소** : 코로나19로 인한 비대면 조치로, Discord 프로그램을 사용해 온라인 수업을 진행

**교재** : 해당 과목 교수님께서 직접 작성하신 [웹페이지](#)를 기반으로 학습을 진행

**학습 프로그램** : Visual Studio 2019

### 학습법

- 실습 : 예시 상황을 주면 그에 대한 해결법을 추론 해보고, 추론결과로 코드를 작성하는 방식의 수업을 진행.
- 기출 풀이 : 기출 문제를 풀이 후, 풀어본 문제에 대한 새로운 문제를 만들고 해결책을 제시.
- 모듈화 : 작성 알고리즘을 도식화 하는 방법에 대해 학습해 해결하고자 하는 문제의 분할 능력을 키움.
- 퀴즈 : 뒤쳐지는 인원이 없도록, 학습이 진행된 내용에 대해서는 퀴즈를 진행하여 이해도를 파악함.
- 과제 풀이 : 과제가 원하는 답안, 풀이 방향에 대해 제시.
- 수업 요약지 : 튜터가 이전에 수업을 들으며 작성한 전체적인 수업 요약지를 통해 수업 흐름을 제시.
- 과목 학습지 제작 : 복습을 위한 튜터링 진행 시 핵심 키워드를 빈칸으로 만든 학습지를 제작하여 배포.
- 수행 과제 풀이 : 튜터가 수행한 해당 과목의 과제를 튜티와 함께 해결하며, 교수님의 스타일을 전수한다.

### 일정진행

- 가용시간표 : 가용 시간표를 통해 튜터링이 가능한 요일과 시간 산출
- 일정한 학습 기간 : 화요일 저녁에 꾸준한 진행으로 반복적인 학습을 할 수 있도록 함
- 횟수 이상의 학습 : 기존 8회로 기획된 수업이지만, 시험기간 및 선행 및 복습을 위한 추가적인 튜터링 진행

#### 4. 우리 팀(전공 튜터링) 운영의 잘된 점(성과)와 피드백(다음에 개선할 사항)

##### 운영성과(튜터)

- 교육의 질 관리 연구 및 개발을 통한 높은 학생 만족도 취득.
- 비교과 프로그램 효과성 분석 및 활성화 방안 연구.
- 개개인의 역량 탐색을 위한 효과적인 방안 도출을 통한 '함께하는 학습' 진행, 적극적 참여로 인한 의욕 증대.
- 튜터의 경험에 비롯한 실수를 공유 후 튜티들의 극복법을 보며, 스스로의 학습 계획 및 실수 극복에 대해 알.
- 상대방에게 질문을 할 시 어떻게 질문을 하여야 정확하고, 효율적인 답안을 얻을 수 있는지 알게됨. 스스로도
- 학습 전달을 위한 교과내용 복습을 통한 학습 성취도 증대.
- 개방되고 밝은 분위기로 선/후배 간의 친목 도모.

##### 운영성과(튜티)

- 다른 과목에 비해 비교적 어렵다고 회자되는 데이터구조 과목의 과제에 대한 포괄적인 개요를 전달받음으로, 제시된 과제에 의도에 맞는 정확한 답안 제출이 가능하였으며, 이를 통한 해당 자료구조에 대한 응용법을 이해함.
- 제한된 시간만이 아닌 추가적인 학습의 기회 제공으로, 시간 부담 없이 모든 내용을 이해함.
- 프로그래밍에 필수적인 내용인 자료구조에 대해 놓치는 내용 없이 이해함으로, 개발직군에 대한 전문성을 지님
- 다른 튜티와의 협력을 통한 문제해결을 통해 프로젝트형 학습의 장점과 과정을 알게됨.
- 자신이 이해한 바를 타인에게 설명하기 위해 생각을 말로 정리하는 단계에서 스스로의 모르는 점을 짚어냄. 또한 조리있게 자신의 알고리즘을 발표하기 위한 스피치 능력이 증대됨.
- 자료구조의 구현을 위한, 자료구조를 통해 구현할 수 있는 알고리즘을 도출하는 과정에서 문제해결역량을 키움.
- 교과 과목 내용을 수업만으로 끝내는 것이 아닌 통합적인 정리를 하도록하여 포트폴리오 작성법에 대해 배움.
- 개방되고 밝은 분위기로 선/후배 간의 친목 도모.

##### 피드백

- 수업내용 중 학생의 이해도에 따른 즉흥적인 내용 구성이 있었는데, 이로 인해 오히려 체계가 잡히지 않은 흐름을 보여 혼동을 주게 됨. 즉흥적이고 자유로운 주제 구성을 위해서는 튜터의 개인 역량증대와 학습내용에 대한 깊은 이해도를 보여야 함을 인지함.
- 튜티의 성향이 수업의 진행 방식과 맞아 원활히 진행되었지만, 문답 방식을 부담스러워해 강의식 수업 진행 방식을 선호한다면 오히려 역효과를 볼 수 있음.

## 5. 후배들에게 전하는 전공 튜터링에 대한 제언

### 튜터에게

#### 황진주(튜터)

- 프로그래밍 과목에 대해 설명할 때에는 코드만을 제시하고 이해를 요구하기 보다는 글 혹은 도식화된 내용을 제시하여 명령어, 함수를 대입해 보는 방식을 적용하는 것이 이해에 효율적이다.
- 튜터링을 하는 이유는 교수님과의 질의응답 혹은 지도법에 대한 부담 완화를 위해 동기/선배와 함께 학습을 하는 이유도 있다는 것을 인지하고, 최대한 부담감 없는 편안한 분위기를 제공하는 것이 좋다.
- 한 차례의 내용이 끝나고 '이해가 되었는가?'에 대한 질문으로 끝나지 말고, 간단한 질의응답을 통해 정확히 이해함을 다시 짚어주는 것이 좋다. 이에 대한 질의 없이 튜터링을 진행하다가 한 팀원이 튜터링 수업을 따라가지 못 하겠다며 튜터링을 돌연 중단을 선언한 튜티가 있는 팀도 있었다.
- 어떤 것을 가르쳐야 할지 모를 때에는 자신이 수업을 들을 때의 과제를 인용하거나 자신이 공부를 하며 헛갈렸던 내용에 대해 질문을 하는 것이 도움이 된다.

#### 최범진(튜티)

- 튜티의 입장에서 튜티를 위해 고생해 주시는 튜터분들에게 많은 걸 바라지는 않습니다만, 오히려 그런 점 때문에 튜터가 힘들지는 않을까 생각해서 궁금한 점을 제대로 물어보지 않게 될 경우를 생각해야 합니다.
- 그래서 튜터로써 튜터링에 참가하게 되는 분들께서는 과거 자신이 힘들었던 부분이나 궁금했던 기억을 되짚어가며 수업하면 튜티들에게 큰 도움을 줄 수 있다는 생각이 듭니다.

#### 임미선(튜티)

- 매주 튜터링이 진행될 때 마다 그 주 진행될 내용에 대해 공지를 해주셨다. 또한 일정을 설계하기 위해 튜터링이 있기 전 수업진도를 다시 체크하고 짚어줬으면 하는 내용을 물어본 후 튜터링 내용을 정했다. 다음 튜터링을 진행 할 때도 체계적인 진행이 있으면 좋겠다.
- 학과 수업에 관한 근황을 수업을 듣는 튜티들보다 빠르게 접하고 알려주셨던 일이 인상 깊었다. 과제정보를 알아와 과제 코드에 대해 모두 주석을 달아주고, 혹여나 과제를 제출하지 않은 튜티가 있는지 물어보주며 학원선생님과 같이 물어주었다. 튜티들과 소통이 자주 이루어진다면 튜티들은 여러 가지로 이점을 볼 수 있으니 다음 튜터분께서도 적극적인 소통이 이루어졌으면 좋겠다.

#### 이지현(튜티)

- 튜터링의 일수에 상관 없이 매주 화요일에 고정적인 시간에 반복적으로 진행되어 복습에 도움이 되었다. 혹여나 수업에서 개념을 놓치게 되더라도 다음 주에 튜터링을 통해 극복이 가능하다는 것이 학업 부담 완화에 큰 도움이 되었다.
- 튜터와 튜티간 합의하에 시간을 잘 정하여 한다면 능률적인 진행이 될 수 있을 것 같다.

#### 이상준(튜티)

- 튜터링을 진행할 시 학습지를 만들어오고, 시험 기출 문제 자료를 들고와 함께 풀기도 하고, 혹여나 과제를 해결하지 못하는 튜티에게 먼저 연락을 해 도와주는 등 많은 도움을 받을 수 있어 좋았다. 다음에 튜터링을 진행하게 될 튜터 또한 지금과 같이는 아니더라도 적극적이었으면, 좋겠다.

## 튜티에게

### 황진주(튜터)

- 튜터링 진행 중 의문이 생긴다면 적극적인 질문을 하는 것이 튜터링 진행에 매우 도움이 된다.
- 질문하고 이해해주는 모습이 너무나도 감사하고 기특하고 대단하고 대견하고 멋지고 최고다.
- 시험이 종료되고 덕분에 좋은 성과를 낼 수 있어 좋았다는 말이 너무나도 고마웠다.

### 최범진(튜티)

- 튜티를 하면서 느끼는 점은 튜터께서 튜티를 위해 상당히 많은 시간을 할애한다는 것이고, 그러한 모습을 보면서 미안하고 고마운 마음에라도 더 열심히 수업에 임하게 된다는것을 느꼈습니다.
- 미래에 튜티로써 튜터링에 참가하게 되는 분들께서는 튜티를 위해 열심히 준비해서 가르쳐주는 튜터들에게 어느정도 존중과 감사를 표현하는 것이 중요하다고 생각합니다.

### 임미선(튜티)

- 선배의 도움 없이는 많이 막막했을 전공 공부를 비교적 쉽게 할 수 있어서 매우 좋았다.

### 이지현(튜티)

- 비대면 수업이 진행되면서 이해도가 떨어지고 그에 따라 어떻게 해야할 지 어려울 때가 많았는데, 튜터링을 통해서 전공수업에 대한 이해도가 높아져서 좋았다.

### 이상준(튜티)

- 모르는게 있으면 튜터분에게 바로바로 물어볼 수 있다는게 제일 좋았고, 수업 덕분에 개념에 대한 이해를 쉽게 할수 있어서 좋았습니다.
- 약속시간을 지켜야하겠고, 튜터분의 수업에 적극적으로 참여했으면 좋겠습니다.



## 2021-1학기 전공튜터링 ( 2 )회차 주간학습일지

작성일: 04월 27일 (화요일)

학 과	응용소프트웨어공학부	튜 터 명	항진주
학습일시	04월 27일 13시 00분~15시 30분	학 습 장 소	Discord
참 석 자	최범진, 이상준, 이지현, 임미선	결석자(사유)	x
학습주제	시간 복잡도 계산과 다양한 배열형태 학습		
학습목표	1. 코드를 읽고 그에 대한 시간복잡도를 구할 수 있다. 2. 다차원의 배열을 1차원으로 변환하여 생각할 수 있다.		
학습방법 (중복체크가능)	<input type="checkbox"/> 발표 및 토의	<input checked="" type="checkbox"/> 강의(설명)식	<input checked="" type="checkbox"/> 퀴즈 및 문제풀이
	<input type="checkbox"/> 실험·실습·실기	<input type="checkbox"/> 기타 ( )	

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 튜터가 제작한 학습지를 통한 학습 진행.

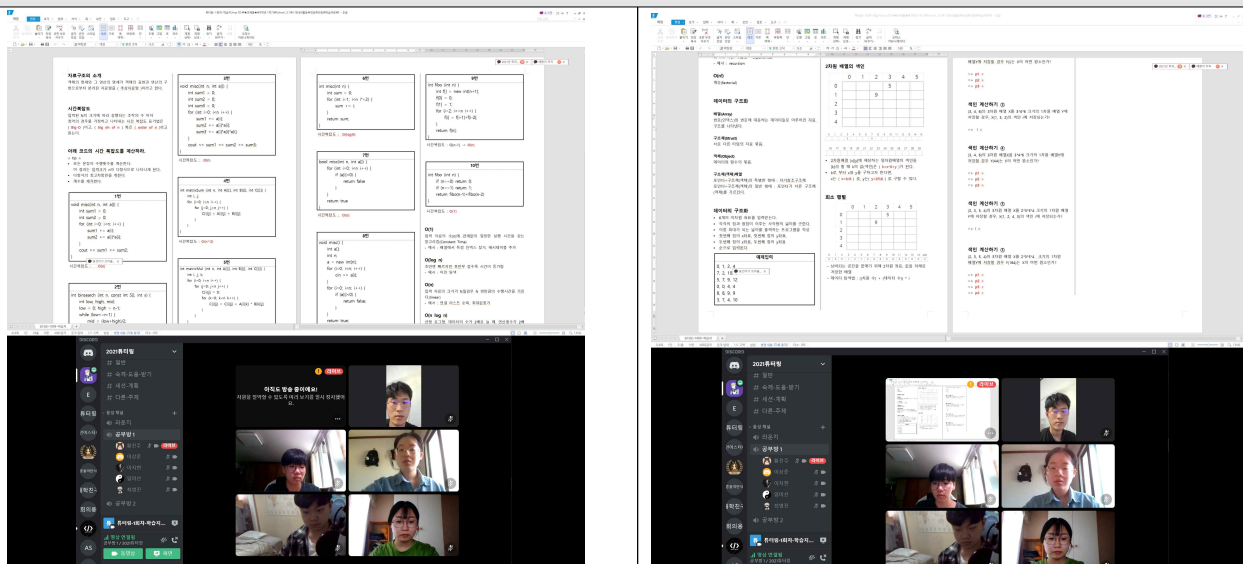
### 세부 학습 내용

1. 시간 복잡도를 BIG-O 기법을 기반으로 계산
2. n차원 배열을 1차원으로 변경할 시의 색인(index) 계산
3. 희소배열의 색인(index) 계산 및 코드 작성

### 다음 튜터링 진행 일자 및 시간

05월 04일 (화) / 20 : 00 ~ 22 : 00

### 학습활동 사진



※ 듀얼모니터 사용으로, 화면이 2개로 나뉘어 찍혔습니다.



## 자료구조의 소개

객체의 명세와 그 연산의 명세가 객체의 표현과 연산의 구현으로부터 분리된 자료형을 ( **추상자료형** )이라고 한다.

## 시간복잡도

입력된 N의 크기에 따라 실행되는 조작의 수 이자,

최악의 경우를 가정하고 나타내는 시간 복잡도 표기법은

( **Big-O** )이고, ( **big oh of n** ) 혹은 ( **order of n** )라고 읽는다.

## 아래 코드의 시간 복잡도를 계산하라.

- 모든 문장의 수행횟수를 계산한다. 이 결과는 입력크기 n의 다항식으로 나타나게 된다.
- 다항식의 최고차항만을 취한다.
- 계수를 제거한다.

### 1번

```
void misc(int n, int a[]) {  
    int sum1 = 0;  
    int sum2 = 0;  
    for (int i=0; i<n; i++) {  
        sum1 += a[i];  
        sum2 += a[i]*a[i];  
    }  
    cout << sum1 << sum2;  
}
```

시간복잡도 :  **$O(n)$**

### 2번

```
int binsearch (int n, const int S[], int x) {  
    int low, high, mid;  
    low = 0; high = n-1;  
    while (low<=high) {  
        mid = (low+high)/2;  
        if (x==S[mid]) return mid;  
        else if (x<S[mid]) high=mid-1;  
        else low = mid+1;  
    }  
    return -1;  
}
```

시간복잡도 :  **$O(\log N)$**

### 3번

```
int fibo (int n) {  
    int f[] = new int[n+1];  
    f[0] = 0;  
    f[1] = 1;  
    for (i=2; i<=n i++) {  
        f[i] = f[i-1]+f[i-2];  
    }  
    return f[n];  
}
```

시간복잡도 :  $O(n-2) \rightarrow$   **$O(n)$**

### 4번

```
int matrixMul (int n, int A[], int B[], int C[]) {  
    int i, j, k;  
    for (i=0; i<n i++) {  
        for (j=0; j<n j++) {  
            C[i][j] = 0;  
            for (k=0; k<n k++) {  
                C[i][j] = C[i][j] + A[i][k] * B[k][j];  
            }  
        }  
    }  
}
```

시간복잡도 :  **$O(n^3)$**



시간복잡도에 따른 특징

시간복잡도	특징
O(1)	입력 자료의 수(n)에 관계없이 일정한 실행 시간을 갖는 알고리즘(Constant Time) - 예시 : 배열에서 특정 인덱스 찾기, 해시테이블 추가
O(log n)	초반엔 빠르지만 후반부 갈수록 시간이 증가함 - 예시 : 이진 탐색
O(n)	입력 자료의 크기가 N일경우 N 번만큼의 수행시간을 가진다.(linear) - 예시 : 연결 리스트 순회, 최대값찾기
O(n log n)	선형 로그형, 데이터의 수가 2배로 늘 때, 연산횟수가 2배 조금 넘게 증가한다. - 예시 : 퀵 정렬, 병합정렬 등
O(n^2)	주로 2중 for loop를 사용하여 조합가능한 모든 쌍을 대상으로 하는 경우가 전형적(quadratic) - 예시 : 버블 정렬,삽입 정렬
O(2^n)	지수형 빅오, '지수적증가'는 무서운 연산횟수 증가를 야기한다. - 예시 :피보나치수열
O(c^n)	최악의 시간 복잡도 - exponential - 예시 : recursion
O(n!)	계승(factorial)

<데이터의 구조화>

배열(Array) :번호(인덱스)와 번호에 대응하는 데이터들로 이루어진 자료구조를 나타낸다.

구조체(Struct) : 서로 다른 타입의 자료 묶음.

객체(Object) : 데이터와 함수의 묶음.

구조체(객체)배열

포인터+구조체(객체)의 특별한 형태 : 자기참조구조체

포인터+구조체(객체)의 일반 형태 : 포인터가 다른 구조체(객체)를 가르킨다.

포인터

	함수 선언에서 사용	문장 연산자로 사용
*	주소를 저장하는 변수(포인터 변수)의 선언	해당 포인터 변수에 저장된 값을 원래 이름으로 하는 변수(메모리)
&	해당 변수에 대한 별칭(참조 변수)의 선언	해당 변수의 주소

2차원 배열의 색인

	0	1	2	3	4	5		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0				5							5				9								
1			9																				
2																							
3																							
4																							
								16	17	18	19	20	21	22	23	24	25	26	27	28	29		

- 2차원배열 [x][y]에 해당하는 일차원배열의 색인을 [k]라 할 때 k의 값(색인)은 (  $k=x*6+y$  )가 된다.
- k로 부터 x와 y를 구하고자 한다면, x는 (  $x=k/6$  ) 로, y는(  $y=k\%6$  ) 로 구할 수 있다.

## 희소 행렬

	0	1	2	3	4	5
0				5		
1			9			
2						
3						
4						

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	size
0	3	5	1	2	7	0	0	0	0	0	0	0	0	0	2

- 낭비되는 공간을 없애기 위해 2차원 좌표, 값을 차례로 저장한 배열
- 데이터 탐색법 :  $((\text{차원 수}) + (\text{데이터 수})) * i$

### 색인 계산하기 ①

(d1, d2, d3)의 3차원 배열X을 d1\*d2\*d3 크기의 1차원 배열Y에 저장할 경우 X(p1, p2, p3)는 Y의 몇번째 위치(i라 하자)에 저장되는가?

$$\Rightarrow i = p1*(d2*d3) + p2*d3 + p3$$

### 색인 계산하기 ②

(d1, d2, d3)의 3차원 배열X을 d1\*d2\*d3 크기의 1차원 배열Y에 저장할 경우 Y(i)는 X의 어떤 원소인가?

$$\Rightarrow p1 = i/(d2*d3),$$

$$\Rightarrow p2 = (i\%(d2*d3))/d3,$$

$$\Rightarrow p3 = i\%d3 == (i\%(d2*d3))\%d3$$

### 색인 계산하기 ③

(2, 3, 5, 4)의 3차원 배열 X를 2\*3\*5\*4 크기의 1차원 배열 Y에 저장할 경우, X(1, 2, 4, 3)의 색인 i에 저장되는가?

$$\Rightarrow i = 1*(3*5*4) + 2*(5*4) + 4*(4) + 3 = 119$$

### 색인 계산하기 ④

(2, 3, 5, 4)의 3차원 배열 X를 2\*3\*5\*4 크기의 1차원 배열Y에 저장할 경우 Y(119)는 X의 어떤 원소인가?

$$\Rightarrow p1 = 119 / (3*5*4), = 1$$

$$\Rightarrow p2 = (119 \% (3*5*4)) / (5*4), = 2$$

$$\Rightarrow p3 = (119 \% (3*5*4)) \% (5*4) / 4 = 4$$

$$\Rightarrow p4 = 119 \% 4 \text{ or } (119 \% (3*5*4)) \% (5*4) \% 4 = 3$$



## 2021-1학기 전공튜터링 ( 3 )회차 주간학습일지

작성일: 05월 09일 (일요일)

학 과	응용소프트웨어공학부	튜 터 명	황진주
학습일시	05월 04일 20시 00분 ~ 22시 30분	학 습 장 소	Discord
참 석 자	최범진, 이상준, 이지현, 임미선	결석자(사유)	x
학습주제	추상자료형의 이해와 구현		
학습목표	1. Class 객체에 대해 이해하고, 구조체와 비교할 수 있다. 2. List 자료구조를 알고, 데이터를 삽입, 삭제 할 수 있다.		
학습방법 (중복체크가능)	<input type="checkbox"/> 발표 및 토의 <input checked="" type="checkbox"/> 실험·실습·실기	<input checked="" type="checkbox"/> 강의(설명)식 <input type="checkbox"/> 기타 (    )	<input checked="" type="checkbox"/> 퀴즈 및 문제풀이

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 교수님이 제작한 학습 사이트를 기반으로 학습 진행.

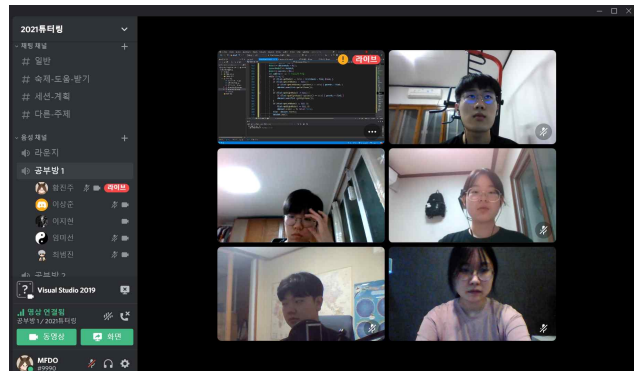
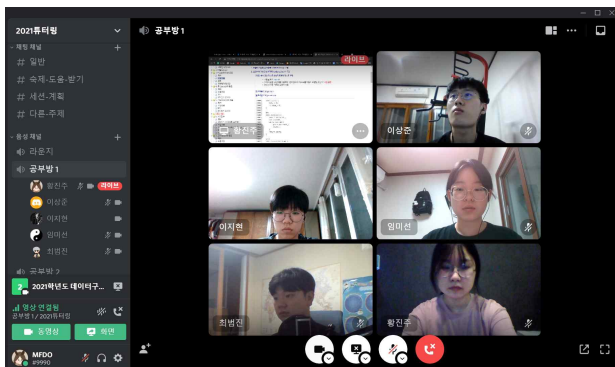
### 세부 학습 내용

1. Class 객체의 특성을 Struct와 비교하며 차이점과 그에 따른 활용도 이해
2. List 자료구조의 선형적인 특성을 이해
3. Class를 활용해 배열기반 List 자료구조의 데이터 삽입(Insert), 삭제(Delete)를 직접 구현

### 다음 튜터링 진행 일자 및 시간

05월 11일 / 20시 00분 ~ 22시 30분

### 학습활동 사진





## 2021-1학기 전공튜터링 ( 3 )회차 주요내용 요약 정리

주제(범위) : List, List를 배열로 구현하는 법, Class란 (+구조체와 클래스의 차이) 작성자 : 최범진

### 데이터구조에서 List란?

순서와 원소의 쌍, **선형목록(Linear List)**으로 불리기도 한다.

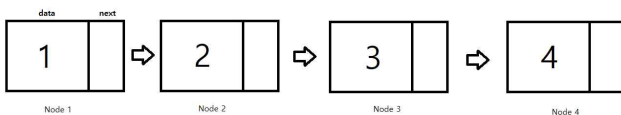
※ **순서목록 (ordered list)**: 사전, 학번 순의 학생목록 등 원소들이 특정 key로 정렬되어있는 목록

### List의 추상자료형

선형목록의 추상자료형		순서목록의 추상자료형	
<b>객체의 명세</b>	<ul style="list-style-type: none"> <li>- 원소들의 목록</li> <li>- 0번부터 n-1까지 위치 차지</li> </ul>	<b>객체의 명세</b>	<ul style="list-style-type: none"> <li>- 원소들의 정렬된 목록</li> <li>- 0번부터 n-1까지의 위치를 차지</li> <li>* i번째 원소는 (i+1)번째 원소보다 앞선다.</li> </ul>
<b>연산의 명세</b>	<pre>int GetSize(); element GetAt(int position); void SetAt(int position, element item); void InsertAt(int position, element item); int Index(key value); element DeleteAt(int position); void Delete(key value); void InsertToHead(element item); void InsertToTail(element item); element DeleteFromHead(); element DeleteFromTail();</pre>	<b>연산의 명세</b>	<pre>int GetSize(); void Insert(element item); element Find(key value); void Delete(key value); element GetAt(index position); index Index(key value);</pre>

### ※ 노드와 리스트의 구조

- **노드(Node)**: 리스트에서 데이터를 저장하는 단위
- **포인터(Pointer)**: 리스트에서 다음 노드의 주소값



=> 노드는 정보를 저장하는 변수 data와 다른 노드를 가리키는 포인터 변수 next로 구현된다.

### <List를 배열로 구현하는 법>

※ List를 배열로 구현하였을 경우 장점은 무엇인가?

구현 난이도가 낮고, 메모리에 순차적으로 정보가 할당되는 특성상 인덱스를 사용하여, 찾고자 하는 데이터를 빠르게 찾을 수 있다.

※ List를 배열로 구현하였을 경우 생기는 문제점은 무엇인가?

데이터의 입력과 삭제가 반복될 때 배열의 공간은 남았지만,  
인덱스 지정의 문제로 공간이 낭비되는 경우가 생긴다.

#### List 객체 선언

	public:	private:
<b>List.h</b>	<pre>list(LMAX = size=100);    // 데이터 크기 지정 int GetSize();           // List 사이즈 반환 void Insert(int item);    // 데이터 삽입 void Delete(int value);   // 데이터 삭제 int GetAt(int position);  // n번째 값 반환 int Index(int value);     // n번째에 값 삽입</pre>	<pre>int LMAX; // List 최대 크기 int arr[10]; // 데이터 저장공간 int size; // 데이터 사이즈 저장</pre>

#### List 객체 데이터 삽입

	void list::Insert(int item)
<b>List.cpp</b>	<pre>if (size==LMAX) return;           // 최대 사이즈 초과시 종료 for (int i=0; 0&lt;size; i++) {     if(arr[i+1] == NULL)           // 저장공간이 비었다면         arr[i+1] = item;          // 값을 저장한다. } size++;                           // size 업데이트</pre>

#### List 객체 데이터 삭제

	void list::Delete(int value)
<b>List.cpp</b>	<pre>if(size == 0){ return; }           // 리스트에 데이터가 없다면 종료 for(int i = 0; i &lt; size; i++) {     if(arr[i] == vlaue) {           // 입력데이터와 일치하는 값을 찾으면         for(int j = i; j &lt; size; j++) // 현재값부터 끝까지 값을 옮겨준다.             arr[j] = arr[j+1];      // (배열로 구현 시의 단점)     } }</pre>

#### 데이터구조에서 Class란?

특정 객체를 생성하기 위해 **변수와 메소드(함수)**를 정의하는 일종의 틀.

#### 구조체와 클래스의 차이점

구조체는 모든 데이터가 공개형(Public) 이지만, 클래스는 기본적으로 폐쇄형(Private)이고,  
공개형 지정을 위해선 따로 선언을 해주어야한다.

특징	구조체(struct)	클래스(class)
형식	값(stack)	참조(heap)
생성	선언만으로 생성	new 연산자 사용
생성자	매개변수 없이 선언 불가능	매개변수 없이 선언 가능
상속	불가능	가능
default	private	public



## 2021-1학기 전공튜터링 ( 4 )회차 주간학습일지

작성일: 05월 16일 (일요일)

학 과	응용소프트웨어공학부	튜 터 명	황진주
학습일시	05월 11일 / 20시 00분 ~ 22시 30분	학 습 장 소	Discord
참 석 자	최범진, 이상준, 이지현, 임미선	결석자(사유)	x
학습주제	List 자료구조의 작성		
학습목표	1. 포인터에 대해 알고 그에 대한 활용을 할 수 있다. 2. Class pointer를 이용하여 List를 구현한다.		
학습방법 (중복체크가능)	<input checked="" type="checkbox"/> 발표 및 토의 <input checked="" type="checkbox"/> 실험·실습·실기	<input checked="" type="checkbox"/> 강의(설명)식 <input type="checkbox"/> 기타 (    )	<input type="checkbox"/> 퀴즈 및 문제풀이

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 교수님이 제작한 학습 사이트를 기반으로 학습 진행.

학습에 필요한 포괄적인 개념을 강의 형태로 학습.

튜터와 구현을 위한 객체를 문답 형태로 함께 참여해 작성하고, 직접 그에 대한 상세 구현을 함.

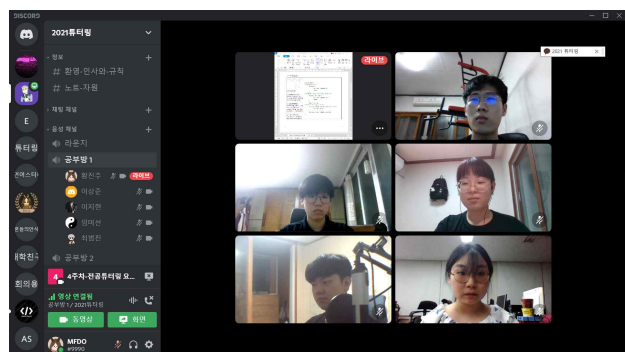
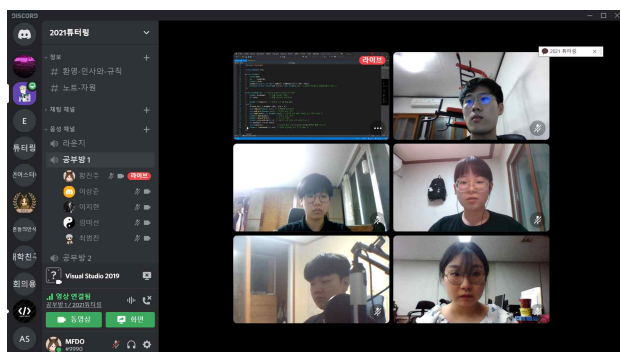
### 세부 학습 내용

1. 객체 포인터가 가지는 특성과 포인터의 개념을 이해한다.
2. List 특성을 분석하고, 구현을 위한 Class 객체를 구성한다.
3. 구성한 객체 정의를 이용해 포인터기반의 List의 데이터 삽입(Insert), 삭제(Delete)를 구현한다.

### 다음 튜터링 진행 일자 및 시간

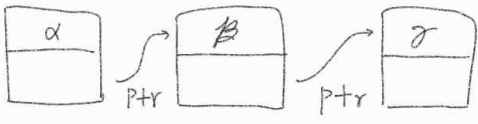
05월 18일 / 20시 00분 ~ 22시 30분

### 학습활동 사진

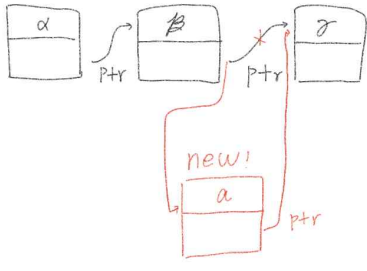




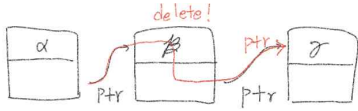
## - 포인터를 이용한 리스트 구현

	<p><b>List</b></p> <p>: 원소뿐만 아니라 다음 원소가 어디 있는지에 대한 <b>위치정보를 포함</b>하는 자료구조</p> <p><b>구현</b></p> <ul style="list-style-type: none"><li>- class 안에 <b>다음 노드를 가리키는 class*</b>를 둔다.</li><li>- 데이터 삽입을 위한 Insert 함수 구현</li><li>- 데이터 삭제를 위한 Delete 함수 구현</li></ul>
---	--

## - 리스트 삽입(Insert)

 <p>삽입할 데이터가</p> <p><b>1. 첫 번째 데이터라면</b> 데이터가 <b>비어있는지 확인</b> <b>첫 노드</b>를 저장하는 포인터에 입력된 데이터를 삽입한다.</p> <p><b>2. 중간 데이터라면</b> 입력데이터를 <b>가리키는 노드를 찾고</b>, 데이터를 삽입한다. 삽입된 데이터는 <b>기존 데이터가 가리키는 노드</b>를 가리킨다.</p> <p><b>3. 마지막 데이터라면</b> <b>제일 마지막 노드</b>를 찾는다. 찾은 노드에 데이터를 삽입.</p>	<pre>Insert(int item) {     //처음 생성되는 리스트     if (list==0) {         list = new node(item, 0);         return;     }     //생성되어있는 리스트 사이에 새로운 리스트 추가     if (item &lt; list-&gt;data) {         list = new node(item, list);         return;     }     node * ptr = list;     //마지막 리스트의 다음 리스트 추가     while (ptr-&gt;next!=0 &amp;&amp; ptr-&gt;next-&gt;data &lt; item) {         ptr = ptr-&gt;next;     }     ptr-&gt;next_ = new node(item, ptr-&gt;next); }</pre>
--	---

## - 리스트 삭제 (Delete)



삭제할 데이터가

### 1. 첫 번째 데이터라면

현재 데이터의 **다음 데이터를**  
**첫 노드로 갱신**해준 뒤 삭제

### 2. 중간 데이터라면

입력데이터를 **가리키는 노드를**  
**찾고**, 데이터를 삭제한다.

삭제 노드를 가리키던 노드는  
**삭제 노드가 가리키는 노드를**  
가리킨다.

### 3. 마지막 데이터라면

2번과 같은 방식으로 노드를  
삭제한다.

```
Delete(int item) {
    node * ptr = list;
    int item;

    //삭제할 리스트가 없을 때
    if (list==0) return -1;

    //삭제해야 할 리스트가 리스트 사이에 있을 때
    if (item == list->data) {
        list = list->next_;
        item = ptr->data_;
        delete ptr;
        return item; }

    //삭제해야 할 리스트가 마지막일 때
    while (ptr->next!=0 && ptr->next->data != item) {
        ptr = ptr->next;
    }
    if (ptr->next==0) return -1;
    node * dptr = ptr->next;
    ptr->next = ptr->next->next;
    item = dptr->data;
    delete dptr;
    return item;
}
```





## 2021-1학기 전공튜터링 ( 5 )회차 주간학습일지

작성일: 05월 23일 (일요일)

학 과	응용소프트웨어공학부	튜 터 명	황진주
학습일시	05월 18일 / 20시 00분 ~ 22시 30분	학 습 장 소	Discord
참 석 자	최범진, 이지현, 임미선	결석자(사유)	이상준(개인 일정)
학습주제	Queue 자료구조의 이해		
학습목표	1. Queue 자료구조의 특성을 이해한다. 2. 여러 가지의 Queue 형태를 안다.		
학습방법 (중복체크가능)	<input checked="" type="checkbox"/> 발표 및 토의 <input checked="" type="checkbox"/> 실험·실습·실기	<input checked="" type="checkbox"/> 강의(설명)식 <input type="checkbox"/> 기타 (    )	<input type="checkbox"/> 퀴즈 및 문제풀이

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 교수님이 제작한 학습 사이트를 기반으로 학습 진행.

학습에 필요한 포괄적인 개념을 강의 형태로 학습.

튜터와 구현을 위한 객체를 문답 형태로 함께 참여해 작성하고, 직접 그에 대한 상세 구현을 함.

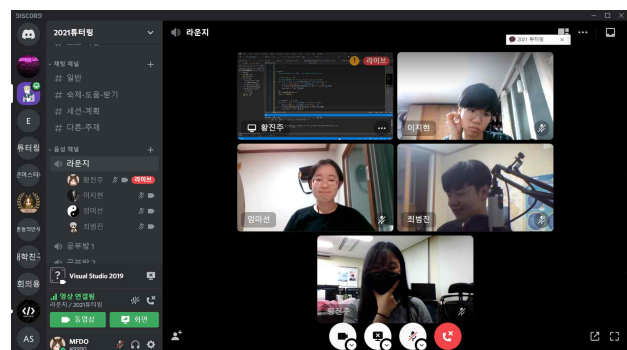
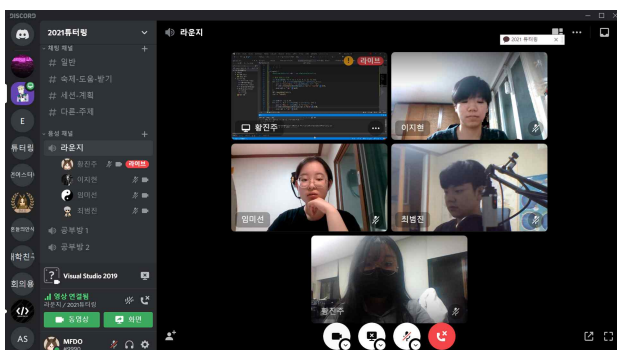
### 세부 학습 내용

1. 선형 큐의 작동 방식을 알고, 리스트와 배열을 통해 구현하는 방식을 안다.
2. 환형 큐의 작동 방식 이해하고, 리스트와 배열 중 효율적인 구현방식을 택할 수 있다.
3. 환형 큐 동작을 기반으로, 선형큐를 이용할 때의 문제점을 파악한다.

### 다음 튜터링 진행 일자 및 시간

05월 25일 / 20시 00분 ~ 22시 30분

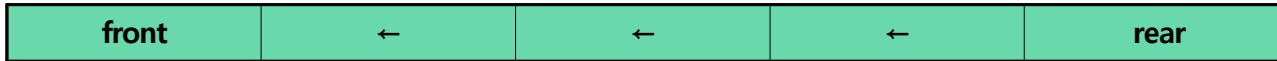
### 학습활동 사진





## <큐의 작동방식>

### <Quene>

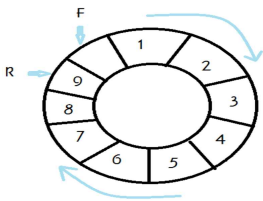


삭제 ←

← 삽입

- 정의 : 삽입, 삭제의 위치와 방법이 한적인 **유한 순서 리스트**(Finite ordered list)이다.
- 특성 : 오른쪽 끝인 rear(or front)부분에서 삽입되어서 왼쪽 끝인 front부분에서 삭제  
> 삽입된 순서대로 삭제되는 **선입선출-FIFO(First In First Out)**의 구조  
ex) 놀이공원의 줄서기, 은행업무 번호표 등 일반적인 줄서기

## <환형 큐의 작동방식>



### 환형 큐(Cricular Quene)

- 크기가 정해져 있는 배열로 원형 큐라고도 불린다.
- **단점** : 큐가 가득차면 더 이상 원소를 집어넣을수 없어, 큐의 크기를 기억하고 있어야하는
- **장점** : 일반 큐와 비교했을 때 공간적인 부분에서 공간의 효율성을 극대화 할 수 있다는
- **삽입** : Rear인 R에서 시계방향으로 움직이면서 삽입을 하고 Front인 F는 가만히 있다.
- **삭제** : F에서 삭제가 일어나므로 R은 정지해있고 F만 돌면서 삭제가 이루어진다.
- **포화상태** : 한 칸을 비우지 않으면, 포화상태인지 구분을 할 수 없어 한 칸이 비워졌다.
  - > 포화 상태 : rear!=front
  - > 공백 상태 : rear==front

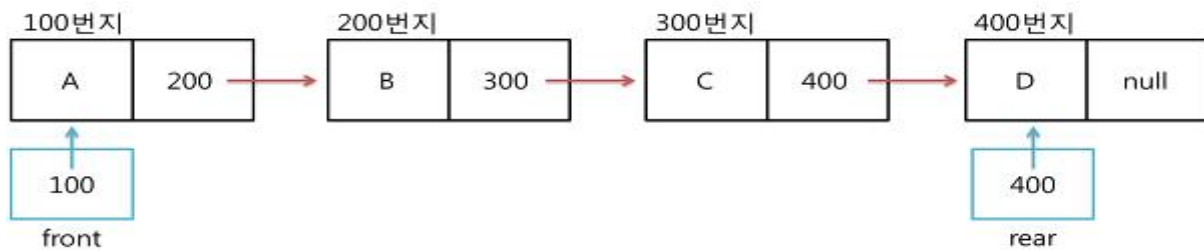
## <선형큐 사용시 문제점>

Quene		4	7	2	5
Index	0	1(front)	2	3	4(rear)

위 그림이 **선형 큐(Linear Quene)**인데 위 그림처럼 2, 4, 7, 25를 순서대로 넣고 맨 앞에 2를 뺐다고 가정하면, 그림처럼 rear는 인덱스4에 위치하고 front는 인덱스 1에 위치하게 된다. 그렇게되면 0번 인덱스에 공간이 남았지만, 더이상 데이터를 채워 넣을 수 없게 되는 문제가 발생하기 때문에 이런 공간 문제를 해결하기 위해 **원형큐(Circular Quene)**가 등장한다.

## <List 이용한 큐 구현>

**연결리스트(Linked List)**로 구현한 **큐(Quene)**로써 **연결된 큐(Linked Quene)**라고한다.



위 그림을 설명하자면 원래 rear에서 입력을 받아서 front로 출력을 했는데, **연결된 큐(Linked Quene)**는 처음에 front로 100을 받아서 A,B,C,D까지 값을 이동한 후 마지막에 rear를 400번지에 넣어주고 출력을 시키는 형태이다.

배열로 구현된 큐(Quene)에 비해서 코드가 더 복잡해지고 메모리 공간을 더 사용하기는 하지만 **크기 제한에 묶여있지 않다는 장점**이 있다. 인덱스로 쓰이는 정수 front와 rear(back)은 head와 tail로써 포인터로 사용된다.



## 2021-1학기 전공튜터링 ( 6 )회차 주간학습일지

작성일: 05월 30일 (일요일)

학 과	응용소프트웨어공학부	튜 터 명	황진주
학습일시	05월 25일 / 20시 00분 ~ 22시 30분	학 습 장 소	Discord
참 석 자	최범진, 이지현, 임미선, 이상준	결석자(사유)	
학습주제	Stack 자료구조의 이해		
학습목표	1. Stack 자료구조의 특성을 이해한다. 2. Stack을 활용한 알고리즘 문제 해결이 가능하다.		
학습방법 (중복체크가능)	<input type="checkbox"/> 발표 및 토의 <input checked="" type="checkbox"/> 실험·실습·실기	<input checked="" type="checkbox"/> 강의(설명)식 <input type="checkbox"/> 기타 (    )	<input checked="" type="checkbox"/> 퀴즈 및 문제풀이

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 교수님이 제작한 학습 사이트를 기반으로 학습 진행.  
학습 주제와 일치하는 알고리즘 문제를 풀어본다.

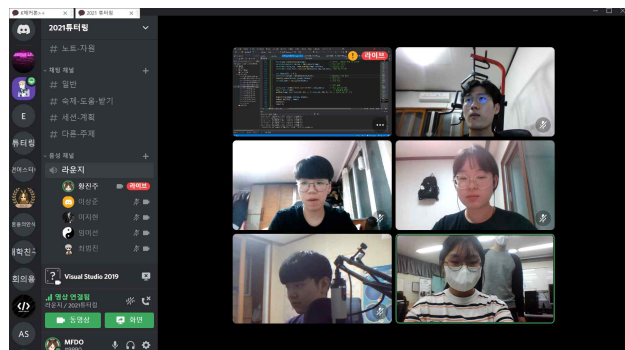
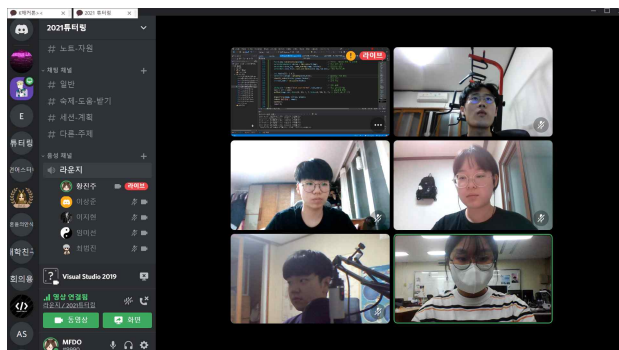
### 세부 학습 내용

1. Stack의 작동 방식을 알고, List를 통해 구현하는 방식을 안다.
2. List 자료구조와의 유사성을 알고, 이 특성을 이용해 스택의 삽입과 삭제를 구현한다.
3. Stack을 활용하여 올바른 괄호가 작성되었는지 확인하는 WFF알고리즘 작성을 한다.

### 다음 튜터링 진행 일자 및 시간

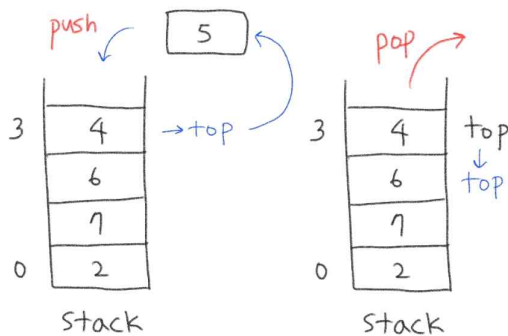
06월 01일 / 20시 00분 ~ 22시 30분

### 학습활동 사진



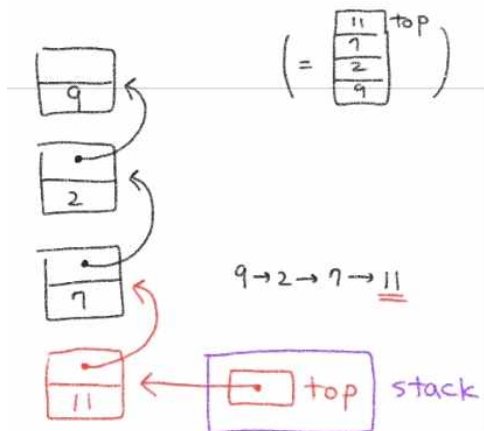


### 〈스택의 작동 방식〉



- 후입선출(LIFO : Last In First Out)  
>> 마지막에 들어간 것이 먼저 나온다.
- 삽입과 삭제가 top에서만 이루어지는 선형적인 자료구조
- ex) 접시를 쌓는 방식, 길찾기 알고리즘(DFS)

### 〈List를 이용하여 스택 구현〉



- 리스트가 다음 노드를 가리키는 특성을 이용한다.
- 어떤 위치든 삭제가 가능한 리스트와는 다르게,  
가장 마지막으로 삽입된 데이터만을 삭제 가능하다.
- 주의점 : top은 **마지막으로 삽입된 원소**를 가리킨다.  
>> **역순**으로 연결됨!

[node.h]

```
#define ERROR -1; //양의 정수만을 다루는 것을 가정
class node {
public:
    node(int d, node * n = 0) {
        next=n;
        data=d;
    }
    node * next;
    int data;
};
```

[stack.h]

```
class Stack {
public:
    Stack ();
    bool IsEmpty();
    bool IsFull();
    bool Push(int el);
    int Pop();
    int Top();
private:
    node * top;
};
```

[stack.cpp]

<pre>#include "stack.h"  bool Stack::IsEmpty() {     return (top==0); }  bool Stack::IsFull() {     return false; }  bool Stack::Push(int el) {     if (IsFull()) return false;     top=new node(el, top);     return true; }</pre>	<pre>Stack::Stack() {     top=0; }  int Stack::Pop() {     if (IsEmpty()) return 0;     node * ptr = top;     top = top-&gt;next;     int el = ptr-&gt;data;     delete ptr;     return el; }  int Stack::Top() {     if (IsEmpty()) return 0;     return top-&gt;data; }</pre>
---	---

#### <스택을 이용한 괄호검사 알고리즘을 작성>

- WFF(Well Formed Formular) : 잘 갖춰진 식 (ex : "{ }", "[ { { } } { } ]")
- 잘 갖춰지지 않은 식 : ex) "{ [ ]", "( { } ]"
- '(', '[', '{' : push()
- ')', ']', '}' : 비교 후에 앞쪽 괄호와 맞춰지면 pop()
- 문자열의 끝까지 진행 후에 스택이 empty 상태면 WFF 만족!

[WFF 코드]

```
#include "stack.h"
#include <iostream>
using namespace std;

bool Match(char a, char b) {
    if (a=='(' && b==')') return true; if (a=='[' && b==']') return true; if (a=='{' && b=='}') return true;
    return false;
}

void main () {
    Stack stack; char c;
    while ((c = getchar())!= '\n') {
        if (c=='(' || c=='[' || c=='{') stack.Push((int) c);
        else if (c==')' || c==']' || c=='}') {
            if (stack.IsEmpty()) break;
            else if (!Match((char) stack.Top(), c)) break;
            stack.Pop();
        }
    }
    if (stack.IsEmpty() && c=='\n') cout << "WFF!!" << endl;
    else cout << "Not a WFF!!" << endl;
}
```



## 2021-1학기 전공튜터링 ( 7 )회차 주간학습일지

작성일: 06월 06일 (일요일)

학 과	응용소프트웨어공학부	튜 터 명	황진주
학습일시	06월 01일 / 20시 00분 ~ 22시 30분	학 습 장 소	Discord
참 석 자	최범진, 이지현, 임미선, 이상준	결석자(사유)	
학습주제	Tree 자료구조의 이해		
학습목표	1. Tree 자료구조의 특성을 일상생활에 접목 가능하다. 2. Tree의 각 구조 명칭을 알고, 구조에 따른 구조를 안다.		
학습방법 (중복체크가능)	<input type="checkbox"/> 발표 및 토의 <input checked="" type="checkbox"/> 실험·실습·실기	<input checked="" type="checkbox"/> 강의(설명)식 <input type="checkbox"/> 기타 ( )	<input checked="" type="checkbox"/> 퀴즈 및 문제풀이

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 교수님이 제작한 학습 사이트를 기반으로 학습 진행.  
학습 주제와 일치하는 알고리즘 문제를 풀어본다.

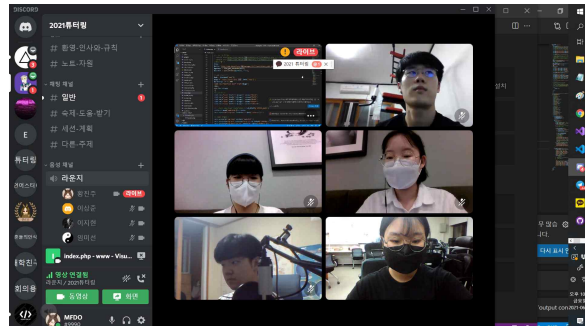
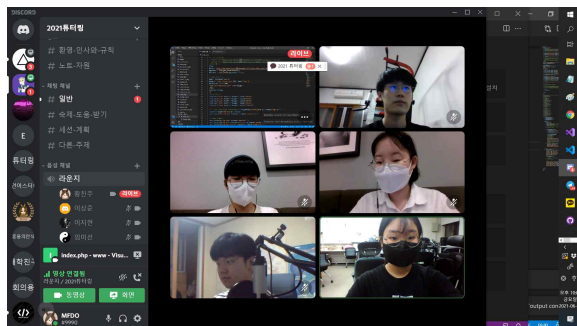
### 세부 학습 내용

- 이전 자료구조에 대한 복습과 응용을 통해 트리 자료구조를 토출한다? (+ 일상속의 트리의 예)
- 트리의 각 부분의 명칭을 알고, 각 부분의 존립에 따라 생기는 구조를 학습한다.
- 좌우 자식 노드가 부모 값에 따라 결정되는 구조인 이진탐색트리에 대해 안다.

### 다음 튜터링 진행 일자 및 시간

06월 03일 / 20시 00분 ~ 22시 30분

### 학습활동 사진

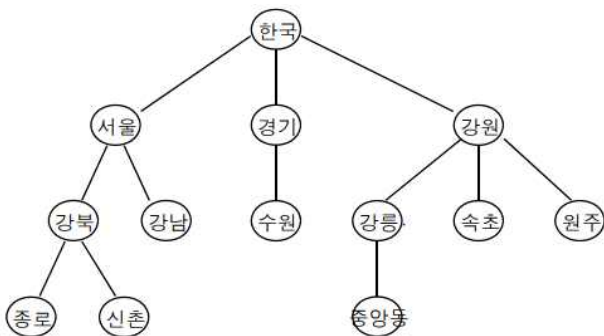




## <트리>

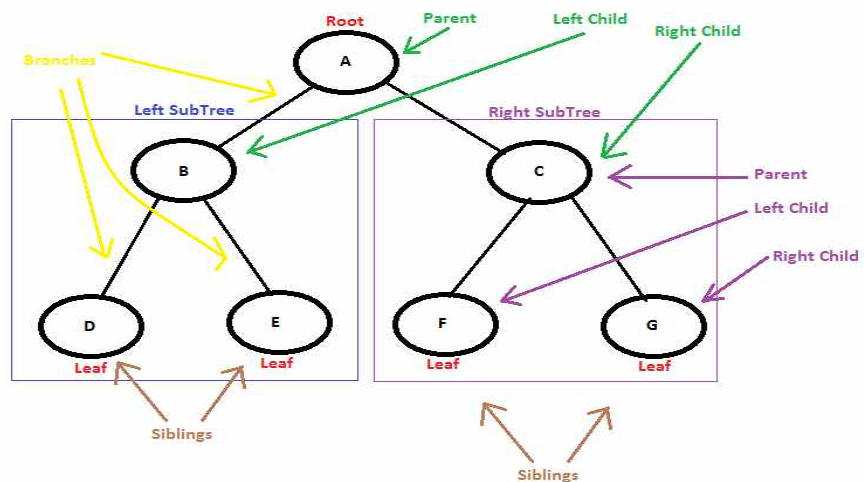
- 정의 : 사이클이 없는 연결 **그래프(graph)**이며 1개 이상의 노드를 갖는 노드의 집합이다.
- 일반적으로 tree 노드중 하나를 root로써 정한 **Rooted Tree**를 tree라고 한다. 주소, 디렉터리 등 구조 등 매우 큰 집합을 나무형태로써 체계적으로 분할할 때 유용하다.

ex) 게임집, 컴퓨터 파일구조, 회사의 조직 구조, 나라, 지방, 시, 군등 계층적인 데이터의 저장



## <트리의 각 부분별 명칭과 형태>

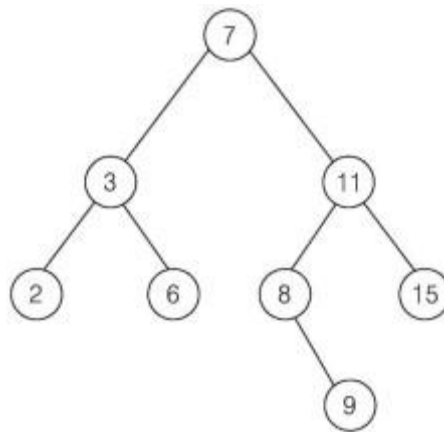
- Node : 노드(정점)
- Edge : 간선
- Parent node : 부모노드
- Child node : 자식노드
- Sibling node : 형제노드
- Root node : 루트노드
- Leaf node : 리프노드
- Ancestor node : 조상노드
- Descendant node : 후손노드
- Terminal node : 단말노드
- Non-terminal node : 비단말노드
- Subtree : 서브트리
- Height : 높이
- Depth : 깊이





### <이진 탐색 트리란?>

이진 탐색 트리(Binary Search Tree)는 이진탐색(binary search)과 연결리스트(linked list)를 합친 것을 의미하며, 자료의 **입력과 삭제가 효율적**이라는 장점을 보유하고 있다. 기본적으로 이진 트리이면서 같은 값을 가진 노드가 없어야 하며, 아래로 내려가는 서브 트리의 경우 **왼쪽은 현재 노드보다 작은 값이, 오른쪽은 현재 노드보다 큰 값이 삽입된다.**



**데이터의 탐색**의 경우, 위의 이진 탐색 트리에서 만약 데이터 '9'를 탐색하고 싶다면 '7'보다 큰 값인 오른쪽으로 내려가, '11'보다 작은 왼쪽으로 내려가고, '8'보다 큰 오른쪽으로 내려가 찾는 방식으로 탐색한다.

**데이터의 삽입**도 같은 방법으로 진행된다. 현재 노드보다 큰 값인지, 작은 값인지를 확인하고, 하위 노드로 내려가는 방식으로 탐색 과정을 가지며, 탐색 과정의 끝에 해당 데이터가 삽입되게 된다.

**데이터 노드를 삭제하는 경우에는** 하위 노드가 없다면 상관 없지만, **하위 노드가 존재하는 노드가 삭제되는 경우 하위 데이터 노드를 상위 노드에 연결해 주어야 한다.**

**삭제해야 할 노드가 두 개 이상인 경우,** 삭제해야 할 노드를 좌, 우의 하위 트리중 가까운 데이터 노드로 대체하고 이후 대체된 노드를 삭제한 후 대체를 위해 제공된 하위 데이터 노드를 상위 노드에 연결해 주면 된다.



## 2021-1학기 전공튜터링 ( 8 )회차 주간학습일지

작성일: 06월 06일 (일요일)

학 과	응용소프트웨어공학부	튜 터 명	황진주
학습일시	06월 03일 / 20시 00분 ~ 22시 30분	학 습 장 소	Discord
참 석 자	최범진, 임미선, 이상준	결석자(사유)	이지현
학습주제	Tree 자료구조의 이해		
학습목표	1. Tree 자료구조의 특성을 이해한다. 2. Tree의 다양한 자료구조를 안다.		
학습방법 (중복체크가능)	<input type="checkbox"/> 발표 및 토의 <input checked="" type="checkbox"/> 실험·실습·실기	<input checked="" type="checkbox"/> 강의(설명)식 <input type="checkbox"/> 기타 ( )	<input checked="" type="checkbox"/> 퀴즈 및 문제풀이

### 튜터링학습 진행 방법

튜터링 Discord방에서 수업 진행 / 교수님이 제작한 학습 사이트를 기반으로 학습 진행.  
학습 주제와 일치하는 알고리즘 문제를 풀어본다.

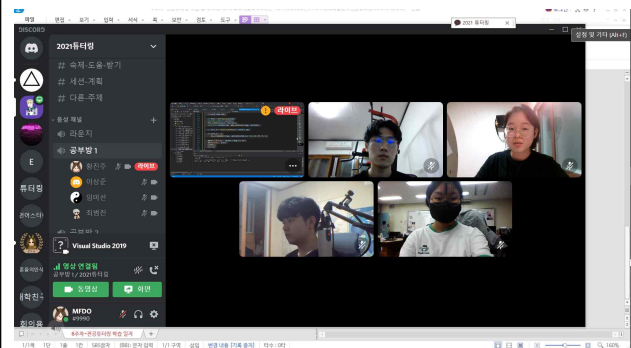
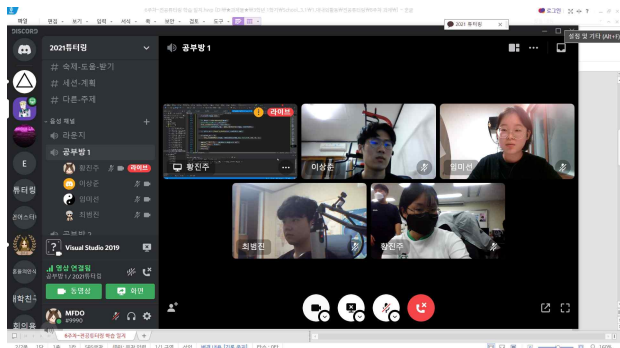
### 세부 학습 내용

- 반복적 구조를 가지는 재귀함수에 대해 이해하고 활용한다.
- 트리 탐색을 위한 DFS를 Stack과 재귀함수를 이용해 구현해본다.
- 트리 탐색을 위한 BFS를 Queue를 이용해 구현해본다.

### 다음 튜터링 진행 일자 및 시간

X

### 학습활동 사진

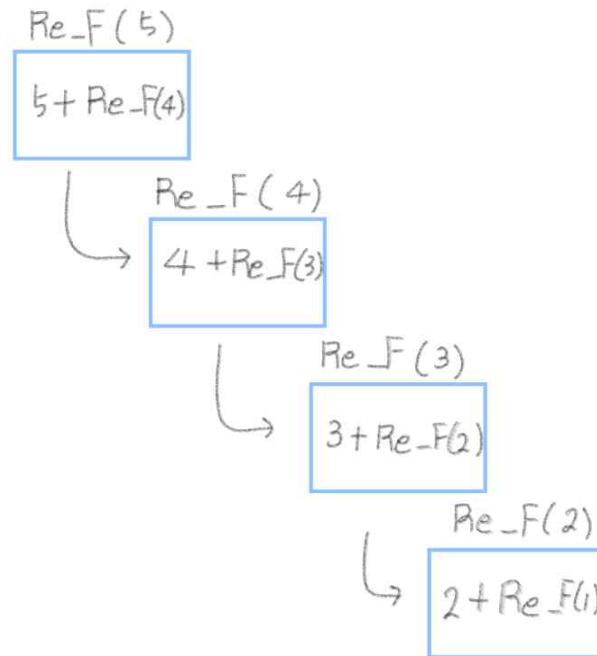




## 1. 재귀함수란?

- 재귀함수(순환함수) : 자기 자신을 호출하는 함수, 함수가 정의될 때 자기자신을 사용하며 정의하는 함수

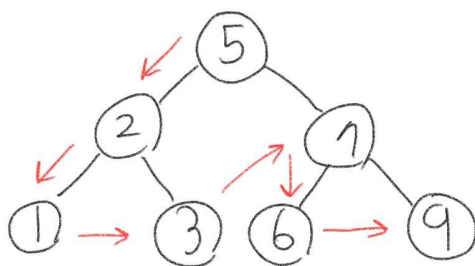
### ▼ [재귀함수의 예시]



## 2. 트리 탐색을 위한 DFS

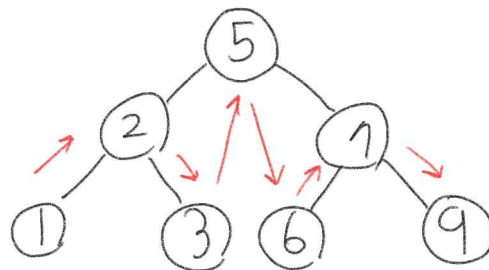
- DFS(Depth First Search, 깊이우선탐색) -> 전위, 중위, 후위 순회로 나뉜다.
- 스택 혹은 재귀 함수를 이용해서 구현한다.

### ▼ [전위 순회]



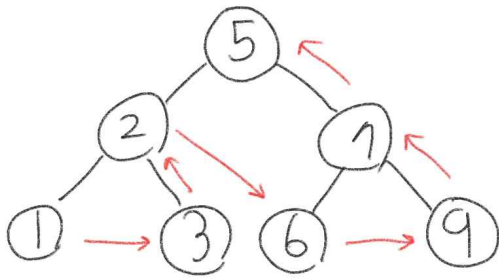
DFS : 5 → 2 → 1 → 3 → 6 → 9

### ▼ [중위 순회]



DFS : 1 → 2 → 3 → 5 → 6 → 7 → 9

### ▼ [후위 순회]



DPS : 1 → 3 → 2 → 6 → 9 → 7 → 5

### ▼ [DFS 구현]

```
void binaryTree::DFS (node * ptr) {
    if (ptr==0) return;
    DFS(ptr->lc_);
    DFS(ptr->rc_);
};

void binaryTree::PreOrderTraversal(node * ptr) {
    if (ptr==0) return;
    visit(ptr);
    PreOrderTraversal(ptr->lc_);
    PreOrderTraversal(ptr->rc_);
};

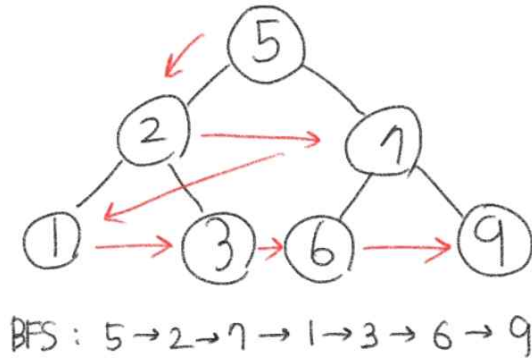
void binaryTree::InOrderTraversal(node * ptr) {
    if (ptr==0) return;
    InOrderTraversal(ptr->lc_);
    visit(ptr);
    InOrderTraversal(ptr->rc_);
};

void binaryTree::PostOrderTraversal(node * ptr) {
    if (ptr==0) return;
    PostOrderTraversal(ptr->lc_);
    PostOrderTraversal(ptr->rc_);
    visit(ptr);
};
```

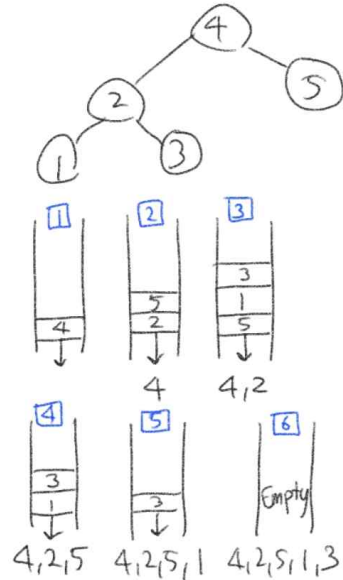
### 3. 트리 탐색을 위한 BFS

- BFS(Breadth First Search, 너비우선탐색) : queue를 이용하여 구현한다.

#### ▼ [BFS를 이용한 노드 방문 순서]



#### ▼ [queue를 이용한 BFS 노드 방문 순서]



#### ▼ [queue를 이용한 BFS 구현]

```
void binaryTree::BFS () {  
    queue q;  
    q.InsertQ(root_);  
    while (!q.IsEmpty()) {  
        node * ptr = q.DeleteQ();  
        visit(ptr);  
        if (ptr->lc_!=0) q.InsertQ(ptr->lc_);  
        if (ptr->rc_!=0) q.InsertQ(ptr->rc_);  
    }  
};
```