

# CSE304 Algorithm

## Lab 1: 순차검색, 배열의 수 더하기, 교환정렬

Fall 2025

CheolJun Park (박철준)

School of Computing, KHU

# Session 1 (55분) – 이론 내용 기반 실습

## ◆ 작성 후 과제 제출

- e-campus 과제 및 평가 -> 오늘 날짜에 작성한 파일 3개 업로드
  - ✓ (1.1.seqsearch.py, 1.2.arrsum.py, 1.3.exchangesort.py)
  - ✓ 반드시 .py로 제출물 업로드 (압축 x)
  - ✓ 소스코드 안에 print문 등 썼을 경우 꼭 주석처리 하기
- 소스코드 이름 변경하지 말고 그대로 올리기 (예시: 1.1.seqsearch.py)
  - ✓ 잘못된 예시 : ~~~.zip, 1.1-1.seqsearch.py

## ◆ AI 도구 사용 자제

- ChatGPT, Copilot 등 AI 도구나 google 검색 활용은 최대한 자제하기
  - ✓ 중간/기말고사 공부한다고 생각하고 혼자 풀어보기

## ◆ 도움 요청 관련

- 막히는 부분이 있으면, (주변에 방해가 되지 않는 선에서) 주변 학우, 조교님께 질의하시면 됩니다

# 실습 과제 경로

<https://github.com/khu-syssec-lab/25fall-cse304-algorithms>

The screenshot shows the GitHub repository page for `25fall-cse304-algorithms`. The repository is public and has 1 branch and 0 tags. The main branch is selected. The repository was last committed 1 minute ago by `chpark-khu` with 4 commits. The file list shows `Week-1` and `README.md`. The `README.md` file is open, showing the title `25fall-cse304-algorithms` and the description: "경희대학교 알고리즘 (CSE 304) 수업의 2025년도 가을학기 실습 자료 입니다."

The screenshot shows the GitHub repository page for `25fall-cse304-algorithms`, specifically the `Week-1` directory. The file list shows the following files:

- `1.1.seqsearch-test.py`
- `1.1.seqsearch.py`
- `1.2.arrsum-test.py`
- `1.2.arrsum.py`
- `1.3.exchangesort-test.py`
- `1.3.exchangesort.py`
- `README.md`

The `README.md` file is open, showing the title `Python 알고리즘 실습 안내` and the following text:

본 실습에서는 실습 예제별 **skeleton code**를 완성한 후, 해당하는 `-test.py`를 실행하여 작성한 코드가 올바르게 동작하는지 확인합니다.

**실습 과정**

1. 각 문제에 대한 **skeleton code**가 제공됩니다. 필요한 부분을 채워서 코드를 완성하세요.
2. `-test.py` 파일을 실행하여 작성한 코드가 테스트를 통과하는지 확인하세요.
3. 올바르게 작성하였으면, 모든 테스트 케이스가 통과합니다.
4. 테스트가 실패하면, 출력값과 기대값을 비교하여 코드를 수정하세요.

**실행 방법**

예를 들어, `1.1.seqsearch.py`를 작성한 이후, 테스트하려면 다음과 같이 실행합니다.

# 실습 알고리즘

---

1. 순차검색
2. 배열의 수 더하기
3. 교환 정렬

# 알고리즘 1.1: 순차검색 알고리즘 (sequential search)

- ◆ 문제:
  - 크기가  $n$ 인 배열  $S$ 에  $x$ 가 있는가?
- ◆ 입력 (파라미터):
  - 양수  $n$ , 배열  $S[1..n]$ , 키  $x$
- ◆ 출력:
  - $x$ 가  $S$ 의 어디에 있는지의 위치. 만약 없으면 0

```
void seqsearch(int n, const keytype S[], keytype x, index& location)
{
    location = 1;
    while (location <= n && S[location] != x)
        location++;
    if (location > n)
        location = 0;
}
```

# 순차검색 알고리즘 (1.1.seqsearch.py)

- ◆ 문제:
  - 크기가  $n$ 인 배열  $S$ 에  $x$ 가 있는가?
- ◆ 입력 (파라미터):
  - 양수  $n$ , 배열  $S[0..n-1]$ , 키  $x$
- ◆ 출력:
  - $x$ 가  $S$ 의 어디에 있는지의 위치. 만약 없으면  $-1$

Week-1 > 1.1.seqsearch.py > seqsearch

```
1  from typing import List
2
3  def seqsearch(n: int, S: List[int], x: int) -> int:
4      location = 0
5
6      # Complete the code here
7      return location
```

```
test_cases = [
    {"S": [10, 7, 11, 5, 13, 8], "x": 10, "expected": 0,
    {"S": [10, 7, 11, 5, 13, 8], "x": 6, "expected": -1,
    {"S": [10, 7, 11, 5, 13, 8], "x": 5, "expected": 3,
```

# 알고리즘 1.2: 배열의 수 더하기

- ◆ 문제:
  - $n$ 개의 수로 된 배열  $S$ 에 있는 모든 수를 더하라
- ◆ 입력 (파라미터):
  - 양의 정수  $n$ , 수의 배열  $S[1..n]$
- ◆ 출력:
  - $S$ 에 있는 수의 합,  $sum$

```
number sum(int n, const number S[ ])
{
    index i;
    number result;

    result = 0;
    for (i=1; i<=n; i++)
        result = result+S[i];
    return result;
}
```

# 배열의 수 더하기 (1.2.arrsum.py)

- ◆ 문제:
  - n개의 수로 된 배열 S에 있는 모든 수를 더하라
- ◆ 입력 (파라미터):
  - 양의 정수 n, 수의 배열 S[0..n-1]
- ◆ 출력:
  - S에 있는 수의 합, sum

Week-1 > 1.2.arrsum.py > ...

```
1 from typing import List
2
3 def arrsum(n: int, S: List[int]) -> int:
4     # Complete the code here
```

```
test_cases = [
    {"S": [10, 7, 11, 5, 13, 8], "expected": 54, "desc": "일반적인 배열"},
    {"S": [], "expected": 0, "desc": "빈 배열"},
    {"S": [100], "expected": 100, "desc": "배열 크기 1"},
    {"S": [-1, -2, -3, -4], "expected": -10, "desc": "음수 배열"}
```



# 알고리즘 1.3: 교환 정렬

- ◆ 문제:
  - 비내림차순(nondecreasing order)으로  $n$ 개의 키를 정렬하라
- ◆ 입력 (파라미터):
  - 양의 정수  $n$ , 키의 배열  $S[1..n]$
- ◆ 출력:
  - 키가 비내림차순으로 정렬된 배열  $S$

```
void exchangesort(int n, keytype S[ ])
{
    index i, j;

    for (i=1; i<=n-1; i++)
        for (j=i+1; j<=n; j++)
            if(S[j] < S[i])
                exchange S[i] and S[j];
}
```

# 교환 정렬 알고리즘 (1.3.exchangesort.py)

- ◆ 문제:
  - 비내림차순(nondecreasing order)으로  $n$ 개의 키를 정렬하라
- ◆ 입력 (파라미터):
  - 양의 정수  $n$ , 키의 배열  $S[0..n-1]$
- ◆ 출력:
  - 키가 비내림차순으로 정렬된 배열  $s$

Week-1 > 1.3.exchangesort.py > ...

```
1  from typing import List
2
3  def exchangesort(n: int, S: List[int]) -> None:
4      # Complete the code here
```

```
{"S": [10, 7, 11, 5, 13, 8], "expected": [5, 7, 8, 10, 11, 13],
{"S": [], "expected": [], "desc": "빈 배열"},
{"S": [100], "expected": [100], "desc": "배열 크기 1"},
{"S": [5, 4, 3, 2, 1], "expected": [1, 2, 3, 4, 5], "desc": "역"},
{"S": [1, 2, 3, 4, 5], "expected": [1, 2, 3, 4, 5], "desc": "이"}
```

# Session 2

- ◆ Leetcode 팀 실습
  - 팀원과 함께 의논하면서 문제를 풀니다 (채점 없음, 점수 반영 x)
  
- ◆ 실습 마무리 후 퇴실 방법
  - 자유 퇴실: 17:55부터 자유 퇴실 가능
  - 조기 퇴실: 17:25부터 조기 퇴실 가능
    - ✓ session 2까지 마쳤으면, 조교님에게 작성 글 보여드린 후 조기 퇴실 가능
    - ✓ e-campus – 게시판 – leetcode 게시판 – 글쓰기
    - ✓ 제목: “[Week1] 이름1, 이름2”
    - ✓ 내용: 문제별 작성한 코드 작성
  
- ◆ [136. Single Number](#)
- ◆ [121. Best Time to Buy and Sell Stock](#)
- ◆ [1480. Running Sum of 1d Array](#)
- ◆ [1365. How Many Numbers Are Smaller Than the Current Number](#)