자료구조

Chap 1. Introduction

2018년 1학기

컴퓨터과학과
민 경 하

Contents

1. **Introduction**
2. Analysis
3. Array
4. List
5. Stack/Queue
6. Sorting
7. Tree
8. Search
9. Graph
10. STL

1. Introduction

- Data structure (자료 구조)
  - A particular way of storing and organizing data in a computer so that it can be used efficiently

데이터를 효율적으로 관리하는 기법

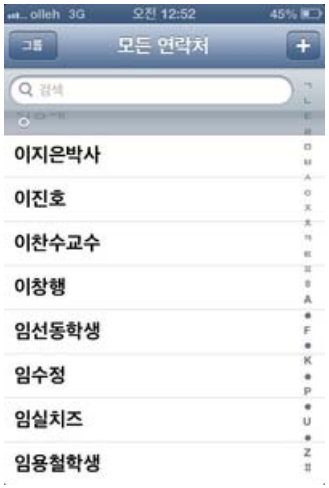
데이터 (data) == 자료?

1. Introduction

- Data structure (자료 구조)

데이터를 효율적으로 관리하는 기법

– 예) 모바일 앱



Phone number

Apps

Music

Memo

Bus

1. Introduction

- Data structure (자료 구조)

데이터를 효율적으로 관리하는 기법

  - 예) 기차표 예매/수강 신청

1. Introduction

- Data structure (자료 구조)

  데이터를 효율적으로 관리하는 기법

  ① 데이터 (data)

  ② 관리 (manipulation)

  ③ 기법 (technique)

  ④ 효율적 (efficiency)

2. Data

- Data

> A collection of organized information, usually the results of experience, observation or experiment, or a set of premises.

- Data type

> A set of data with values having predefined characteristics.

1. System-defined data type

2. User-defined data type

2. Data

1. System-defined data type

(1) primitive data type

- Bit
- Byte
  - 8bits
- Word
  - 4 bytes or 8 bytes

2. Data

1. System-defined data type

(2) char

- BCD code
- EBCDIC code
- ASCII code
  - 8 bit system = 7 bits for code + 1 bit for parity check
- Uni-code
  - 16 bit system

2. Data

1. System-defined data type

(3) int

- 4 bytes (depends on the system)
  - precision: $-2^{31} \sim 2^{31} - 1$
  - _int8, _int16, _int32, _int64: 1 ~ 8 bytes
- Extended to short (2 bytes), long (4 bytes) or unsigned int (4 bytes)
  - The precision of long → 2^{32}: $-2^{31} \sim 2^{31} - 1$
  - The precision of unsigned int → 2^{32}: $0 \sim 2^{32} - 1$
- Three types of integer implementation
  - Sign bit
  - 1's complement
  - 2's complement

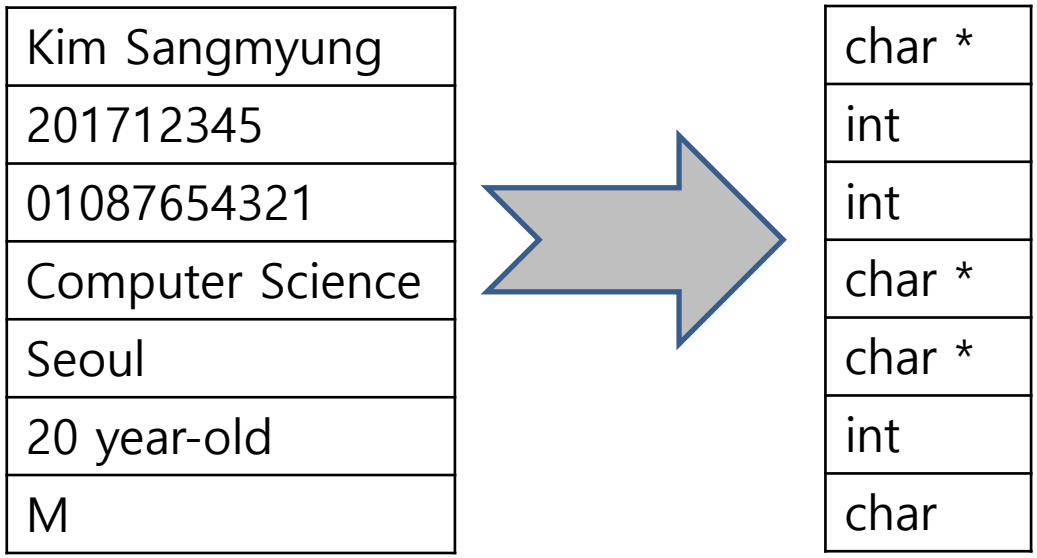
2. Data

1. System-defined data type

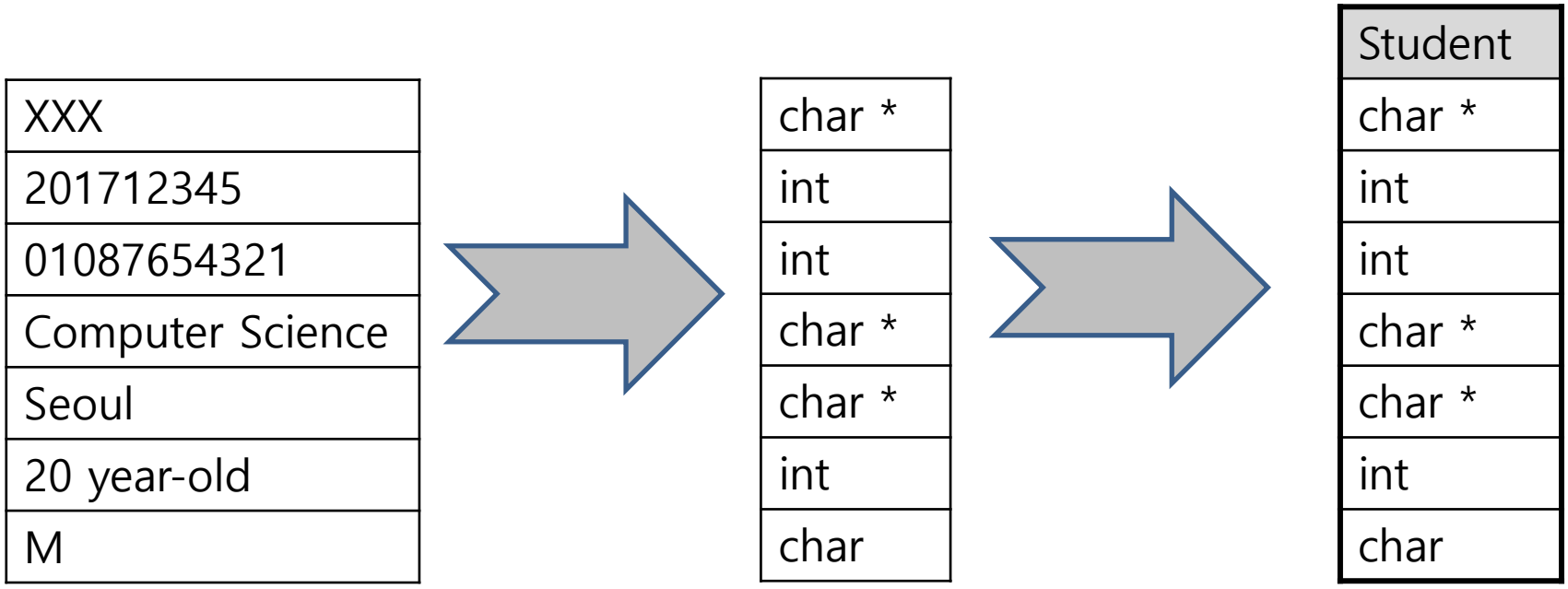
(4) float

- What is floating point? (→← fixed point)
  - In fixed point, 321.05 is stored as 321.05
  - In floating point, 321.05 is stored as 0.32105×10^3
    - » 0.32105: value
    - » 10: base
    - » 3: exponent
- Example: 4-byte floating point
  - Sign: 1 bit
  - Exponent: 7 bits
  - Value: 24 bits

2. User-defined data type

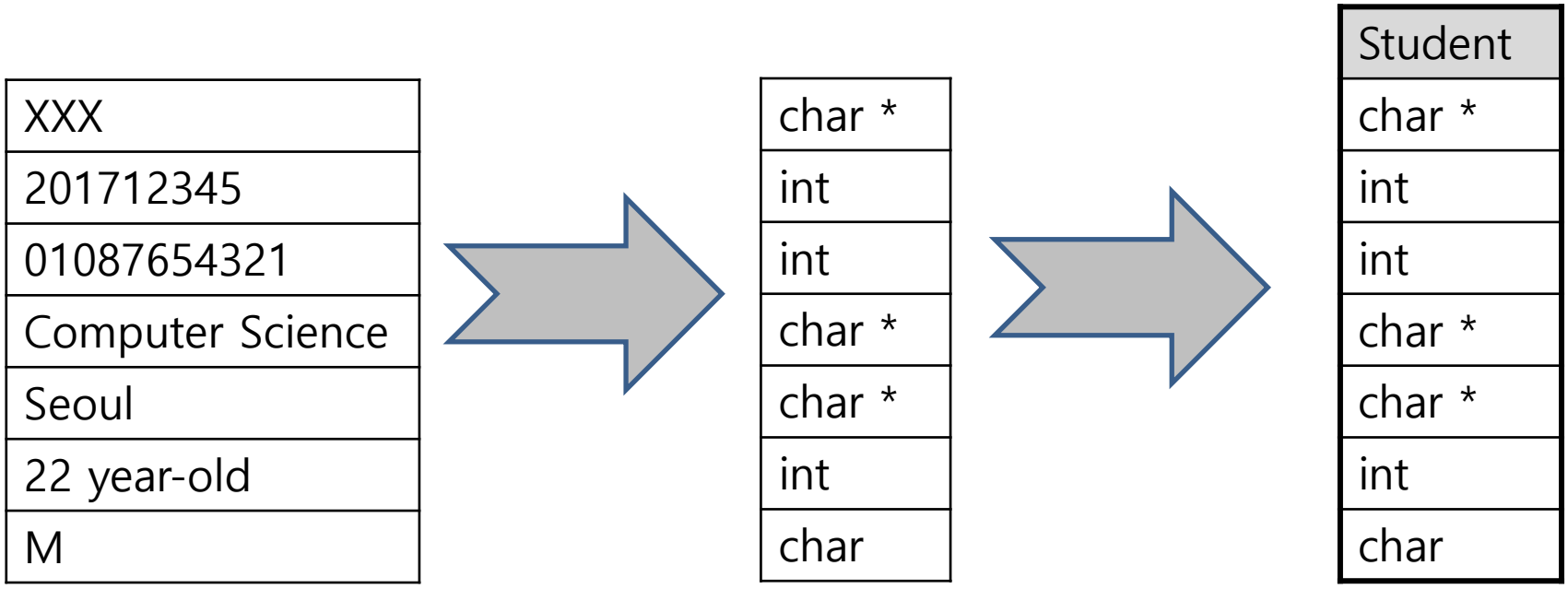


2. User-defined data type



2. User-defined data type

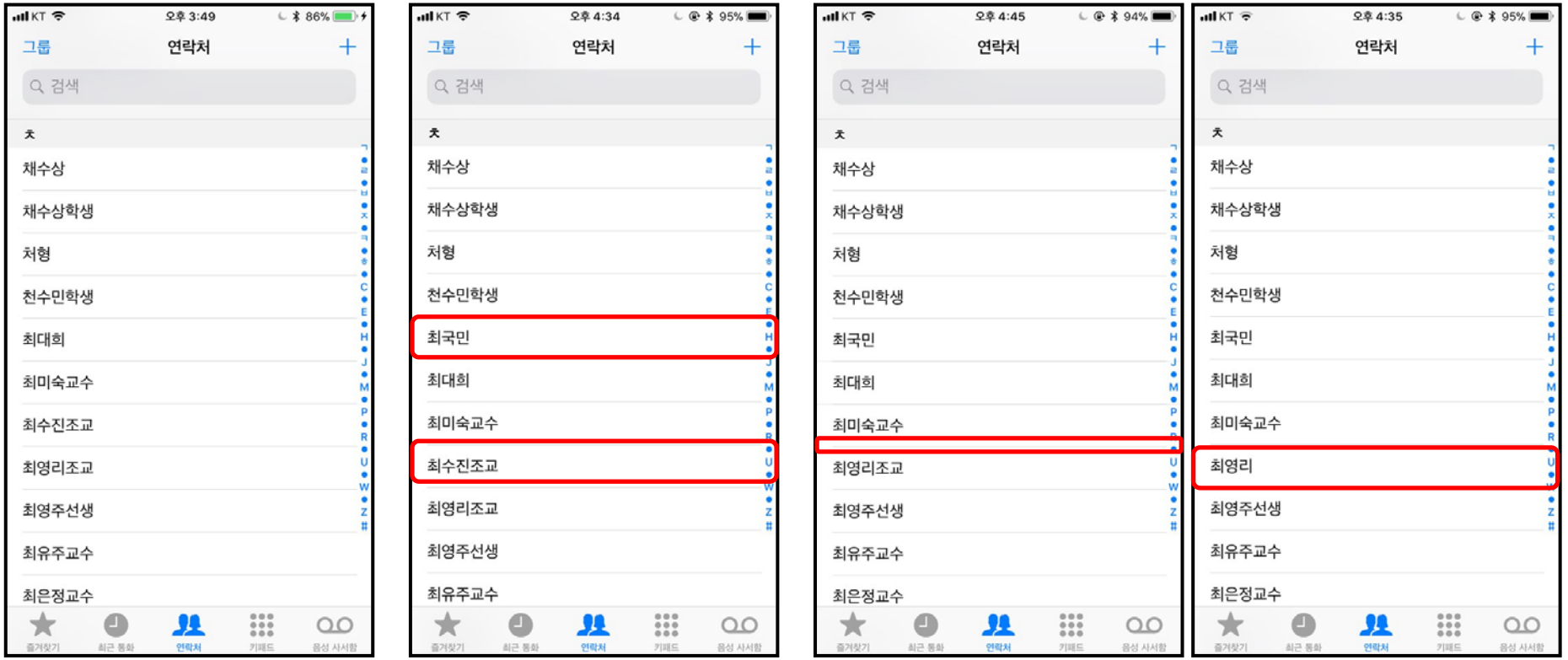
- struct in C
- class in C++



3. Manipulation

- Manipulation?

Insert 최국민 → Delete 최수진 → Modify 최영리조교 → 최영리



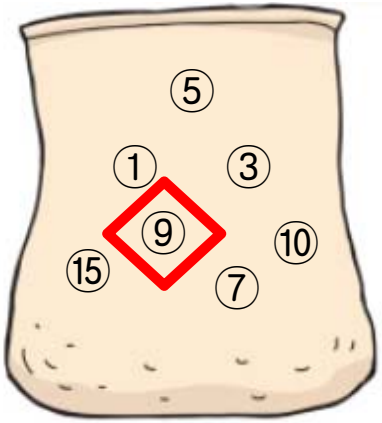
3. Manipulation

- Manipulation?
  - Insert, delete, modify, ……
  - Insert, delete는 몇 번 사용? 1번
  - 가장 많이 사용하는 연산? search

  - 가장 중요한 연산: search

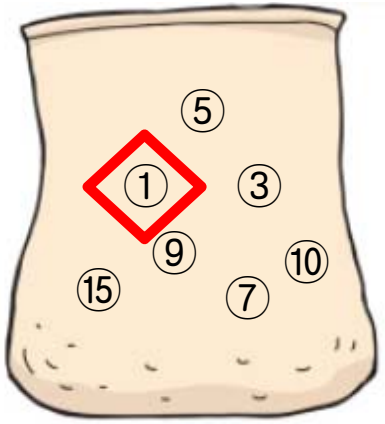
3. Manipulation

- 3 types of search
  - Find an **arbitrary** element in a given set
  - Find the **earliest / last** element in a give set
  - Find the **top** (maximum/minimum) element in a given set



Find **arbitrary**

Find **earliest/last**



Find **top**

4. Technique

- Technique
  - Organization + operations
- Organization
  - Data structure
- Operations
  - How to implement insert, delete, search, …
  - abstraction

4. Technique

- Organization (1): list

철이 2914

영이 3165

훈이 1211

순이 9801

돌이 7812

옥이 4151

Data

(Phone list)

철이 2914
영이 3165
훈이 1211
순이 9801
돌이 7812
옥이 4151

Organization

(list)

4. Technique

- Organization (2): sorted list

철이 2914

영이 3165

훈이 1211

순이 9801

돌이 7812

옥이 4151

Data

(Phone list)

철이 2914
영이 3165
훈이 1211
순이 9801
돌이 7812
옥이 4151

돌이 7812
영이 3165
옥이 4151
순이 9801
철이 2914
훈이 1211

Organization

(sorted list)

4. Technique

- Organization (3): queue

철이 (4)

영이 (1)

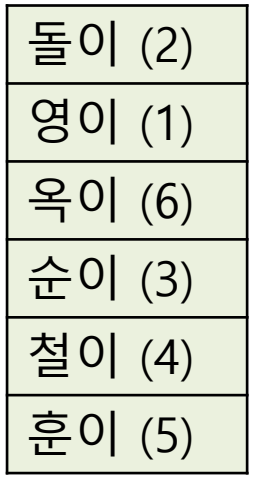
훈이 (5)

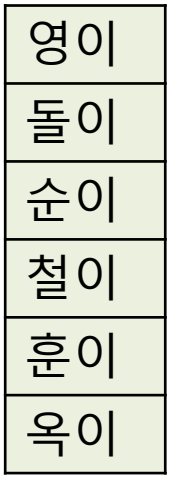
순이 (3)

돌이 (2)

옥이 (6)

Data
(Waiting list)

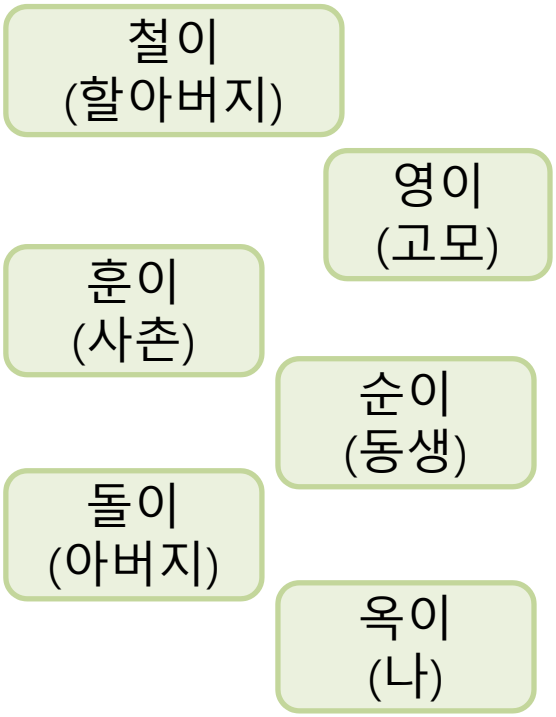




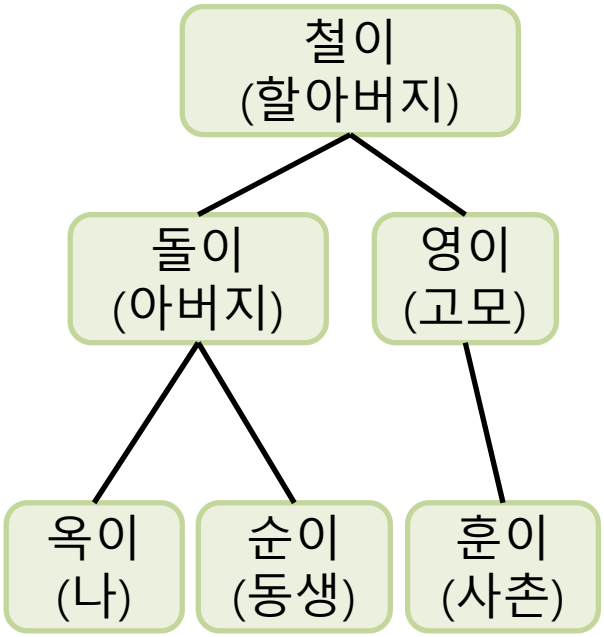
Organization
(queue)

4. Technique

- Organization (4): hierarchy



철이 (할아버지)
돌이 (아버지)
영이 (고모)
옥이 (나)
순이 (동생)
훈이 (사촌)



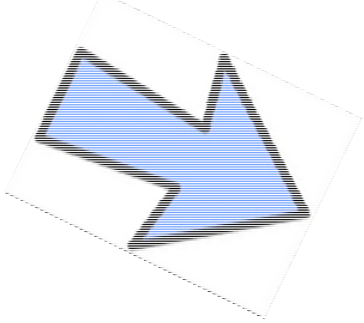
Data
(family list)

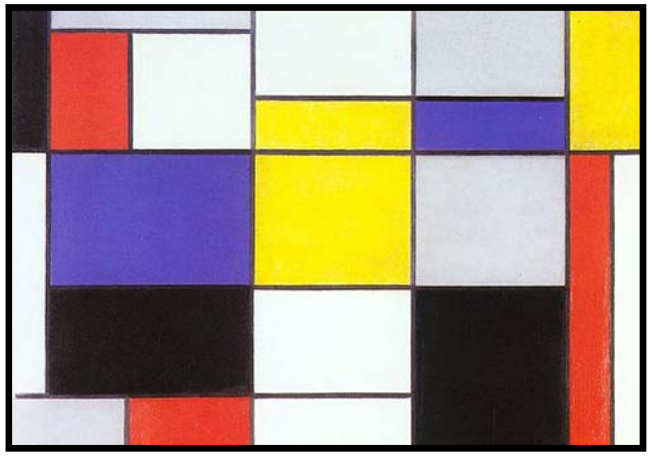
Organization
(tree)

4. Technique

- Abstraction

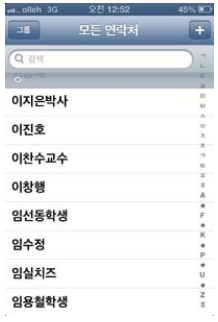
A general idea rather than one relating to a particular object, person, or situation





4. Technique

- Abstraction in computer science
  - Have a lot of applications
  - Remove the specifics of the applications
  - Find common properties



.....

Apps				
Data	**Phone #**	**App Name**	**Bus #**	**.....**
Structure	**List**			
Operation	**Search, Insert, Delete, ...**			

5. Efficiency

- Efficient
  - Efficiency = solution / **resource**

- Effective
  - Effective = **solution** / resource

5. Efficiency

- Performance
  - Best case
  - Average case
  - Worst case

- Resource in computer science
  - Time → CPU
  - Space → Memory

5. Efficiency

- Performance
  - Not how much time it takes for an input
  - But how much time it increases as an input increases

```
i = 0;
while ( i < n ) {
    printf(“hello”);
    i++;
}
```

```
for ( i = 0; i < n; i++ ) {
    for ( j = 0; j < n; j++ ) {
        printf(“hello”);
    }
}
```

n = 10	f(n) = 10	g(n) = 100
n = 100	f(n) = 100	g(n) = 10,000
n = 1,000	f(n) = 1,000	g(n) = 1000,000
n = 10,000	f(n) = 10,000	g(n) = 100,000,000
n = 100,000	f(n) = 100,000	g(n) = 10,000,000,000

Contents

1. **Introduction**
2. Analysis
3. Array
4. List
5. Stack/Queue
6. Sorting
7. Tree
8. Search
9. Graph
10. STL