01강

알고리즘과 자료구조

알고리즘 개요

서울과학기술대학교 신일훈 교수

0. 강의개요

- 신일훈 교수
  - 서울과학기술대학교 전자공학과
  - ilhoon.shin@seoultech.ac.kr

0. 강의개요

강의 내용

- 알고리즘의 개념 및 시간복잡도
- 다양한 문제 해결 기법 이해 및 이를 활용한 문제 해결 (파이썬)
  - brute-force
  - divide and conquer
  - dynamic programming
  - montecarlo simulation
  - greedy

0. 강의개요

강의 내용

- 자료구조와 알고리즘의 효율성 관계
- 다양한 자료구조 이해 및 구현 (파이썬)
  - linked list
  - stack
  - queue
  - hash
  - tree
  - graph

0. 강의개요

선이수 요구사항

- 파이썬 프로그래밍

강의 및 참고 교재

- 파이썬 알고리즘 (최영규, 생능출판)
- 파이썬으로 쉽게 풀어 쓴 자료구조 (최영규, 천인국, 생능출판)

- 파이썬과 함께 하는 자료구조의 이해 (양성봉, 생능출판)
- 알기 쉬운 알고리즘 (양성봉, 생능출판)

학습목표

1. 알고리즘의 개념을 이해한다.
2. 알고리즘의 조건을 이해한다.
3. 알고리즘의 표현방법을 이해한다.
4. 알고리즘의 효율성 분석 방법을 이해한다.
5. 알고리즘의 설계 기법을 이해한다.

01

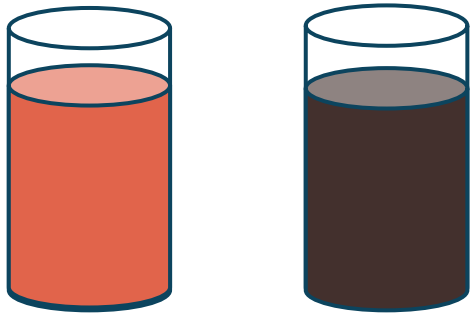
알고리즘 개념

1. 알고리즘(algorithm)

❑ 알고리즘이란?

- 어떤 문제의 해답을 구하기 위한 단계적인 절차를 순서대로 명확하게 나타낸 것.

두 컵(A, B)의 음료를 맞바꾸는 알고리즘

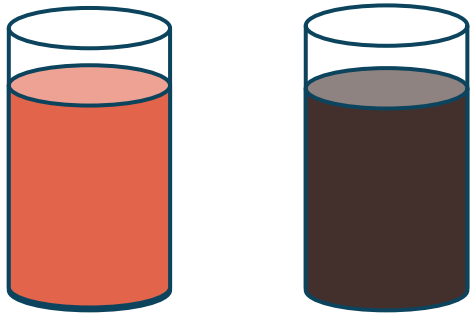


1. 알고리즘(algorithm)

❑ 알고리즘이란?

- 어떤 문제의 해답을 구하기 위한 **단계적인 절차**를 **순서대로 명확**하게 나타낸 것.

두 컵(A, B)의 음료를 맞바꾸는 알고리즘 (잘못된 예시)



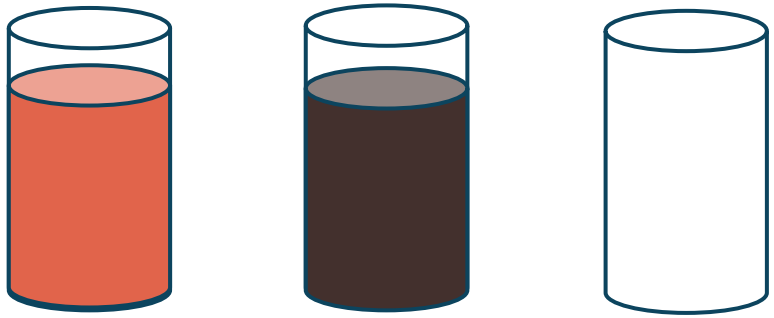
=> A, B의 음료를 맞바꿔라

1. 알고리즘(algorithm)

❑ 알고리즘이란?

- 어떤 문제의 해답을 구하기 위한 **단계적인 절차**를 **순서대로 명확**하게 나타낸 것.

두 컵(A, B)의 음료를 맞바꾸는 알고리즘

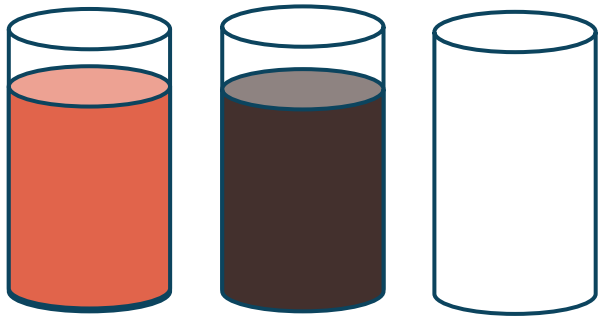


1. 알고리즘(algorithm)

❑ 알고리즘이란?

- 어떤 문제의 해답을 구하기 위한 **단계적인 절차**를 **순서대로 명확**하게 나타낸 것.

두 컵(A, B)의 음료를 맞바꾸는 알고리즘



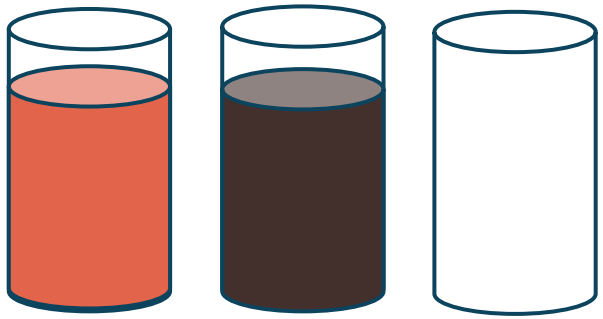
1. 컵 C를 준비한다.
2. 컵 A의 음료를 컵 C에 따른다.
3. 컵 B의 음료를 컵 A에 따른다.
4. 컵 C의 음료를 컵 B에 따른다.

1. 알고리즘(algorithm)

❑ 알고리즘이란?

- 어떤 문제의 해답을 구하기 위한 **단계적인 절차**를 **순서대로 명확**하게 나타낸 것.

두 컵(A, B)의 음료를 맞바꾸는 알고리즘 (잘못된 예시)



1. 컵 C를 준비한다.
2. 컵 B의 음료를 컵 A에 따른다.
3. 컵 C의 음료를 컵 B에 따른다.
4. 컵 A의 음료를 컵 C에 따른다.

1. 알고리즘(algorithm)

❑ 알고리즘이란?

- 어떤 문제의 해답을 구하기 위한 단계적인 절차를 순서대로 명확하게 나타낸 것.

두 변수(A, B)의 값을 맞바꾸는 알고리즘

1. 변수 C를 준비한다.
2. 변수 A의 값을 변수 C에 저장한다.
3. 변수 B의 값을 변수 A에 저장한다.
4. 변수 C의 값을 변수 B에 저장한다.

02

알고리즘 조건(특성)

2. 알고리즘의 조건(특성)

❑ 명확성

- 알고리즘을 구성하는 각 명령의 의미는 모호하지 않고 명확해야 함

두 변수(A, B)의 값을 맞바꾸는 알고리즘

1. 변수 C를 준비한다.
2. 변수 A의 값을 변수 C에 저장한다.
3. 변수 B의 값을 변수 A에 저장한다.
4. 변수 C의 값을 변수 B에 저장한다.

2. 알고리즘의 조건(특성)

❑ 유한성

- 알고리즘은 일정한 시간 내에 종료되어야 함
- 무한루프를 포함하면 안 됨

두 변수(A, B)의 값을 맞바꾸는 알고리즘

1. 변수 C를 준비한다.
2. 변수 A의 값을 변수 C에 저장한다.
3. 변수 B의 값을 변수 A에 저장한다.
4. 변수 C의 값을 변수 B에 저장한다.

2. 알고리즘의 조건(특성)

❑ 유효성

- 컴퓨터에서 실행 가능해야 함

두 변수(A, B)의 값을 맞바꾸는 알고리즘

1. 변수 C를 준비한다.
2. 변수 A의 값을 변수 C에 저장한다.
3. 변수 B의 값을 변수 A에 저장한다.
4. 변수 C의 값을 변수 B에 저장한다.

2. 알고리즘의 조건(특성)

❑ 효율성

- 효율적인 (빠르고 메모리 사용량이 적은) 알고리즘일수록 가치가 높음

1-N까지 합계를 구하는 알고리즘1

1. sum = 0
2. num = 1
3. sum = sum + num
4. num = num + 1
5. if (num <= N) then go to step 3
6. otherwise print(sum)

2. 알고리즘의 조건(특성)

❑ 효율성

- 효율적인 (빠르고 메모리 사용량이 적은) 알고리즘일수록 가치가 높음

1-N까지 합계를 구하는 알고리즘2

```
1. sum = (N * (N+1)) / 2
2. print(sum)
```

03

알고리즘의 표현 방법

3. 알고리즘의 표현 방법

- 자연어(한글, 영어 등)로 표현

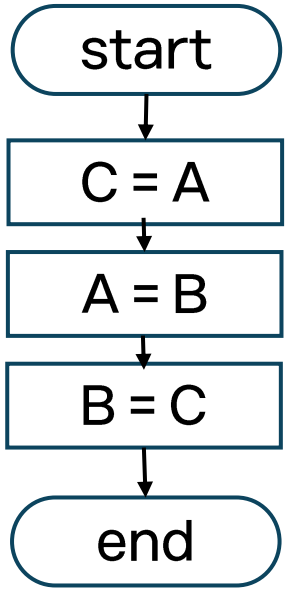
두 변수(A, B)의 값을 맞바꾸는 알고리즘

1. 변수 C를 준비한다.
2. 변수 A의 값을 변수 C에 저장한다.
3. 변수 B의 값을 변수 A에 저장한다.
4. 변수 C의 값을 변수 B에 저장한다.

3. 알고리즘의 표현 방법

- 흐름도(flowchart)로 표현
  - 누구나 이해할 수 있는 장점
  - 알고리즘이 복잡해지면 흐름도가 지나치게 비대하고 복잡해질 수 있음

두 변수(A, B)의 값을 맞바꾸는 알고리즘



3. 알고리즘의 표현 방법

- 의사코드 (pseudo-code)로 기술
  - 프로그램 코드와 유사한 형태 (하지만 상세한 detail은 생략 가능)
  - 논문 등에서 알고리즘을 표현할 때 널리 활용됨

두 변수(A, B)의 값을 맞바꾸는 알고리즘

```
C <= A
A <= B
B <= C
```

3. 알고리즘의 표현 방법

- 프로그램 언어로 기술
  - 파이썬으로 프로그래밍

두 변수(A, B)의 값을 맞바꾸는 알고리즘

```
C = A
A = B
B = C
```

04

알고리즘 정확성 검증

4. 알고리즘의 정확성 검증

실험적 분석 (test)

- 다양한 입력값을 이용해 예상된 결과가 도출되는지 검증
  - 동치(동등)분할 입력
  - 경계값 입력

점수 => 학점 변환 문제

80 – 100 : A

60 – 79 : B

40 – 59 : C

20 – 39 : D

0– 19 : F

4. 알고리즘의 정확성 검증

실험적 분석 (test)

- 다양한 입력값을 이용해 예상된 결과가 도출되는지 검증
  - 동치(동등)분할 입력
  - 경계값 입력

동치분할입력

- 90 => A
- 70 => B
- 50 => C
- …

경계값입력

- 101 => 에러
- 100 => A
- 80 => A
- 79 => B
- …

4. 알고리즘의 정확성 검증

- 증명적 분석
  - 수학적인 방법 활용
    - 수학적 귀납법
    - …

05

알고리즘 효율성 분석

5. 알고리즘 효율성 분석

- 시간 효율성
  - 알고리즘이 얼마나 빠른 시간 안에 결과를 도출하는가.

- 공간 효율성
  - 알고리즘이 얼마나 많은 메모리를 사용하는가.

5. 알고리즘 효율성 분석

- 시간 효율성
  - 일반적으로 데이터의 입력 크기에 따른 시간 복잡도(time complexity)로 표현

- 시간 복잡도
  - 데이터의 입력 크기에 따라, 필요한 연산의 수
  - 데이터 크기 N이 증가함에 따라 필요한 연산의 수가 얼마만큼 증가하는가?

5. 알고리즘 효율성 분석

- 시간 복잡도 예시 (빅오 표기)
  - O(1) ⇒연산의수가데이터크기N과관계없이일정함
  - O(logN) ⇒연산의수가데이터크기N이증가함에따라logN에비례하여증가함
  - O(N) ⇒연산의수가데이터크기N이증가함에따라N에비례하여증가함
  - O(NlogN) ⇒…
  - $O(N^2)$ ⇒…
  - $O(2^N)$ ⇒…
  - …

5. 알고리즘 효율성 분석

효율성 비교

- $O(1) > O(\log N) > O(N) > O(N\log N) > O(N^2) > O(2^N)$

5. 알고리즘 효율성 분석

- 시간 복잡도 종류
  - 최선(best case) 시간복잡도
  - 평균(average) 시간복잡도
  - 최악(worst case) 시간복잡도

5. 알고리즘 효율성 분석

O(1) 알고리즘 예시

1-N까지 합계를 구하는 알고리즘

```
sum = (N * (N+1)) / 2
print(sum)
```

5. 알고리즘 효율성 분석

O(N) 알고리즘 예시

1-N까지 합계를 구하는 알고리즘

1. sum = 0
2. num = 1
3. sum = sum + num
4. num = num + 1
5. if (num <= N) then go to step 3
6. otherwise print(sum)

5. 알고리즘 효율성 분석

- 전체 알고리즘이 여러 알고리즘으로 구성된 경우…
  - 1단계 알고리즘 시간복잡도 : O(1)
  - 2단계 알고리즘 시간복잡도 : O(N)
  - 3단계 알고리즘 시간복잡도 : O(logN)

  - 전체 알고리즘의 시간복잡도 => O(N)
    - 가장 복잡도가 높은 알고리즘에 의해 전체 복잡도가 결정됨.

06

알고리즘 설계 기법

6. 알고리즘 설계 기법

- 100개의 숫자가 저장된 파이썬 리스트에서 큰 숫자를 찾으시오.
- 피보나치(1000)의 값은?
  - fibo(1000) = fibo(999) + fibo(998)
- 1000000이하의 숫자 중 가장 큰 소수는?
- sqrt(3)은 얼마일까 (근사값).
- 원주율은 얼마일까 (근사값).

6. 알고리즘 설계 기법

- brute-force (억지) 기법
- decrease and conquer
- divide and conquer
- dynamic programming
- montecarlo simulation
- greedy
- …

정리하기

- 알고리즘 개념
- 알고리즘 조건
- 알고리즘 정확성 검증
- 알고리즘 효율성 분석
- 알고리즘 설계 기법

02강

다음시간안내 ▶▶▶

Brute-force 전략