

IT@CBOOK



파이썬 *for* Beginner 자료구조와 알고리즘

우재남 지음

연습문제 해답은 제공하지 않습니다.

한빛아카데미
HANBIT ACADEMY

IT@CBOOK

쉽게 배우는 자료구조 with 파이썬

문병로 지음

한빛아카데미
HANBIT ACADEMY



쉽게 배우는 자료구조 with 파이썬

문병로 지음

연습문제 해답은 제공하지 않습니다.

한빛아카데미
HANBIT ACADEMY

워밍업

입문자도 부담 없이 시작할 수 있도록 기본 개념과 파이썬 기초 문법을 먼저 살펴봅니다.

자료구조와
알고리즘 소개

파이썬 기본 문법과
데이터 형식

기본기 다지기

가장 기본적인 자료구조와 알고리즘을 다룹니다. 개념 → 구현 → 응용 순으로 체계적으로 학습할 수 있습니다.

순차 자료구조

순차/연결 리스트

스택과 큐

이진 트리/그래프

재귀 호출

정렬(선택·삽입·버블·퀵 정렬)

검색

동적 계획법

응용력 기르기

중간중간 이해도를 확인하고 응용하며 실력을 기를 수 있는 다양한 학습 장치가 효율적으로 학습을 돕습니다.

SELF STUDY

응용 예제

예제모음

단순연결, 원형연결

01

CHAPTER

자료구조와 알고리즘 소개

학습목표

- 자료구조의 개념을 파악할 수 있다.
- 네 가지 자료구조의 종류를 알 수 있다.
- 알고리즘의 정의를 이해하고, 알고리즘과 자료구조의 관계를 알 수 있다.
- 알고리즘을 표현하는 다양한 방법을 이해하고, 성능 측정 방법을 확인한다.
- 파이썬을 설치해서 실행 환경을 구축한다.

SECTION 00 생활 속 자료구조와 알고리즘

SECTION 01 자료구조의 개념과 종류

SECTION 02 알고리즘

SECTION 03 파이썬 소개와 설치

연습문제



Section 00 생활 속 자료구조와 알고리즘

■ 자료구조(Data Structure)는?

- 그릇(자료)을 효율적으로 관리하는 방법



■ 알고리즘(Algorithm)은?

- 목적지까지 최적의 이동 경로를 찾는 방법



Section 01 자료구조의 개념과 종류

■ 자료구조의 개념

- 컴퓨터 프로그래밍 언어에서 효율적인 자료(데이터)의 형태



(a) 자료구조가 없는 동물원



(b) 자료구조가 있는 동물원

그림 1-1 동물원 자료구조 예

동물들을 커다란 울타리 하나에 가두어 하나하나 찾기도 어렵고 관리가 혼란스러움
동물별로 울타리에 나누어 가두어 효율적으로 관리 가능
→ 데이터는 동일하지만 효율적인 자료구조에 저장함으로써 효율적으로 관리 가능

Section 01 자료구조의 개념과 종류

■ 자료구조의 종류



그림 1-2 자료구조의 종류

주	해당 장	주제
2	2장	자료구조와 알고리즘 소개
3	3장	파이썬 기초 문법과 데이터 형식
4	4장	선형 리스트
5	5장	단순 연결 리스트
6	6장	원형 연결 리스트
7	7장	스택
8		중간고사
9	8장	큐
10	9장	이진 트리
11	10장	그래프
12	11장	재귀 호출
13	12장	정렬 기본
14	13장	정렬 고급
15	14장	검색/동적 계획법
16	15장	기말고사

Section 01 자료구조의 개념과 종류

- 단순 자료구조

- 프로그래밍 언어의 데이터 형식에 해당하는 정수, 실수, 문자, 문자열 등

- 정수

정수 0, 100, 1234, -27 등

int 또는 integer 형태로 소수점이 없음

- 실수

실수 0.1, 3.14, 1.234567 등

소수점이 있는 형태로 float 등으로 표현

- 문자

문자 'A', '채', '3' 등

한 글자를 의미하며, *char*로 표현
주로 작은따옴표(' ')로 묶어 줌

- 문자열

문자열 "안녕", "1234", "한" 등

글자 여러 개를 연결한 것으로, *string*으로 표현
주로 큰따옴표(" ")로 묶어 준다,

Section 01 자료구조의 개념과 종류

■ 선형 자료구조

- 데이터를 한 줄로 순차적으로 표현한 형태. 선형 리스트, 연결 리스트, 스택, 큐 등



그림 1-3 선형 자료구조의 형태

■ 비선형 자료구조

- 하나의 데이터 뒤에 여러 개가 이어지는 형태. 트리와 그래프 등

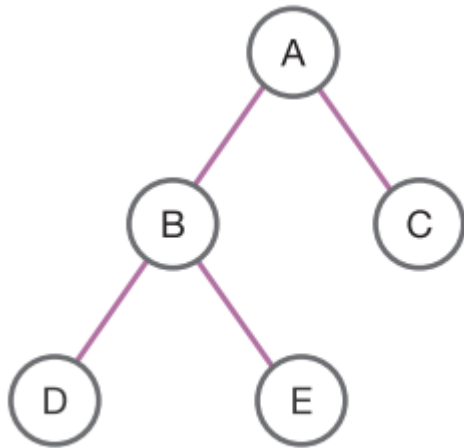


그림 1-4 비선형 자료구조의 형태

Section 01 자료구조의 개념과 종류

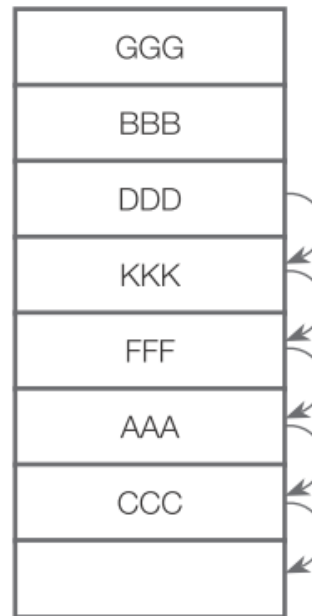
■ 파일 자료구조

- 파일 내용이 디스크에 저장되는 방식에 따라 순차 파일과 직접 파일로 구분
- 순차 파일(Sequential File)
 - 파일 내용을 논리적인 처리 순서에 따라 연속해서 저장하는 것
 - 구조가 간단하기에 저장되는 공간 효율이 높지만, 다른 내용을 추가하거나 삭제할 경우에는 파일 내용을 재구성해야 하므로 상당히 시간이 오래 걸림



(a) 구성이 완료된 순차 파일

PPP →



(b) 중간에 데이터가 추가된 경우

PPP 데이터가 세 번째 위치에 추가된 경우 세 번째 이후의 모든 데이터가 한 칸씩 아래로 밀려야 하므로 시간이 오래 걸리고 데이터를 검색하려면 처음부터 끝까지 모두 찾아야 하므로 비효율적

그림 1-5 순차 파일에서 중간에 내용이 추가된 경우 예

Section 01 자료구조의 개념과 종류

■ 파일 자료구조

■ 직접 파일(Direct File)

- 파일 내용을 임의의 물리적 위치에 기록하는 방식으로 직접 접근 방식(Direct Access Method)

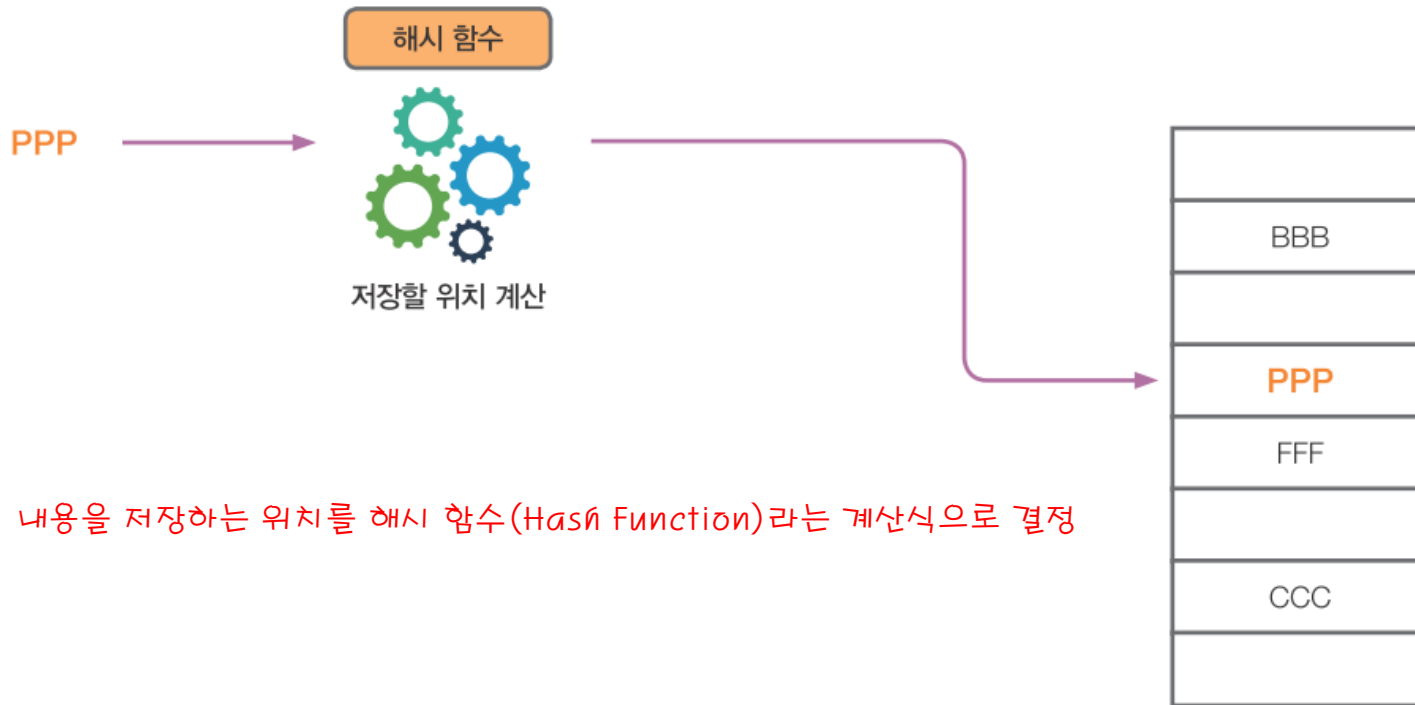
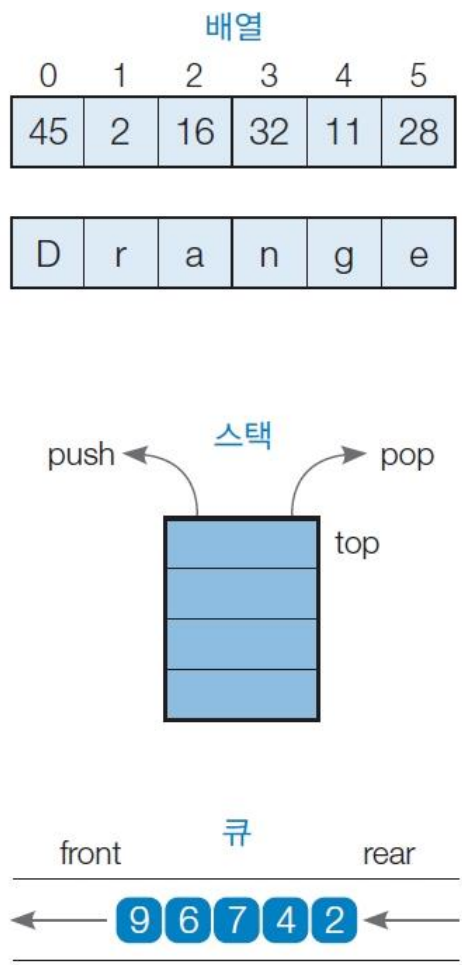


그림 1-6 해시 함수로 내용을 저장할 위치를 계산하는 직접 파일 예

■ 색인 순차 파일

- 순차 파일과 직접 파일이 결합된 형태

자료구조 종류



행렬

0	1	2	3	4	5
0					
1					
2					
3					
4					

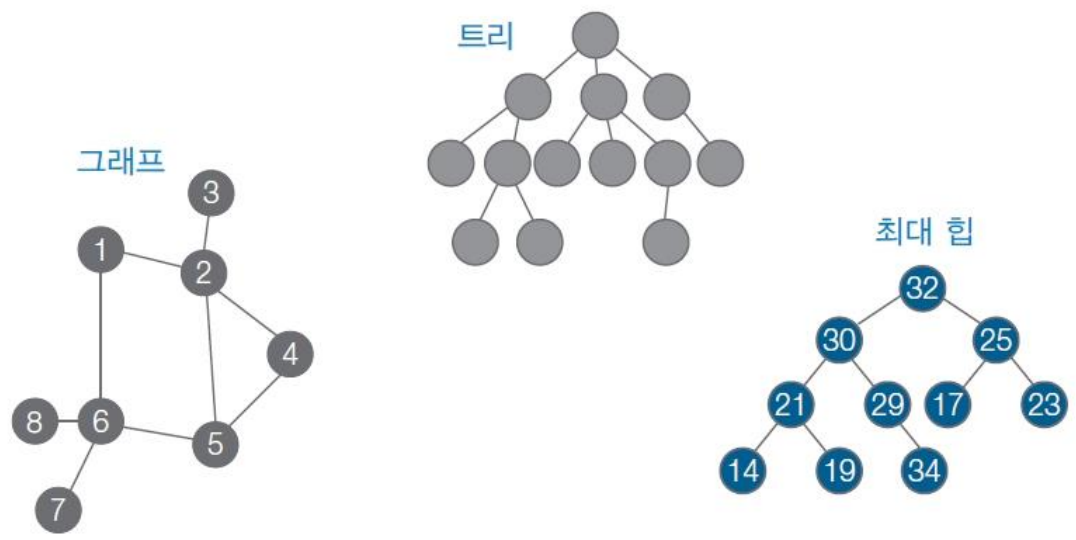


그림 1-7 자료구조의 종류

Section 02 알고리즘

■ 알고리즘

- 어떤 문제를 해결해 가는 논리적인 과정
- 알고리즘 예
 - 트럭에는 최대 7톤의 무게를 실을 수 있고 단 1회만 운송할 수 있다면, 선호도 합이 최대가 되도록 동물을 태우는 방법은?

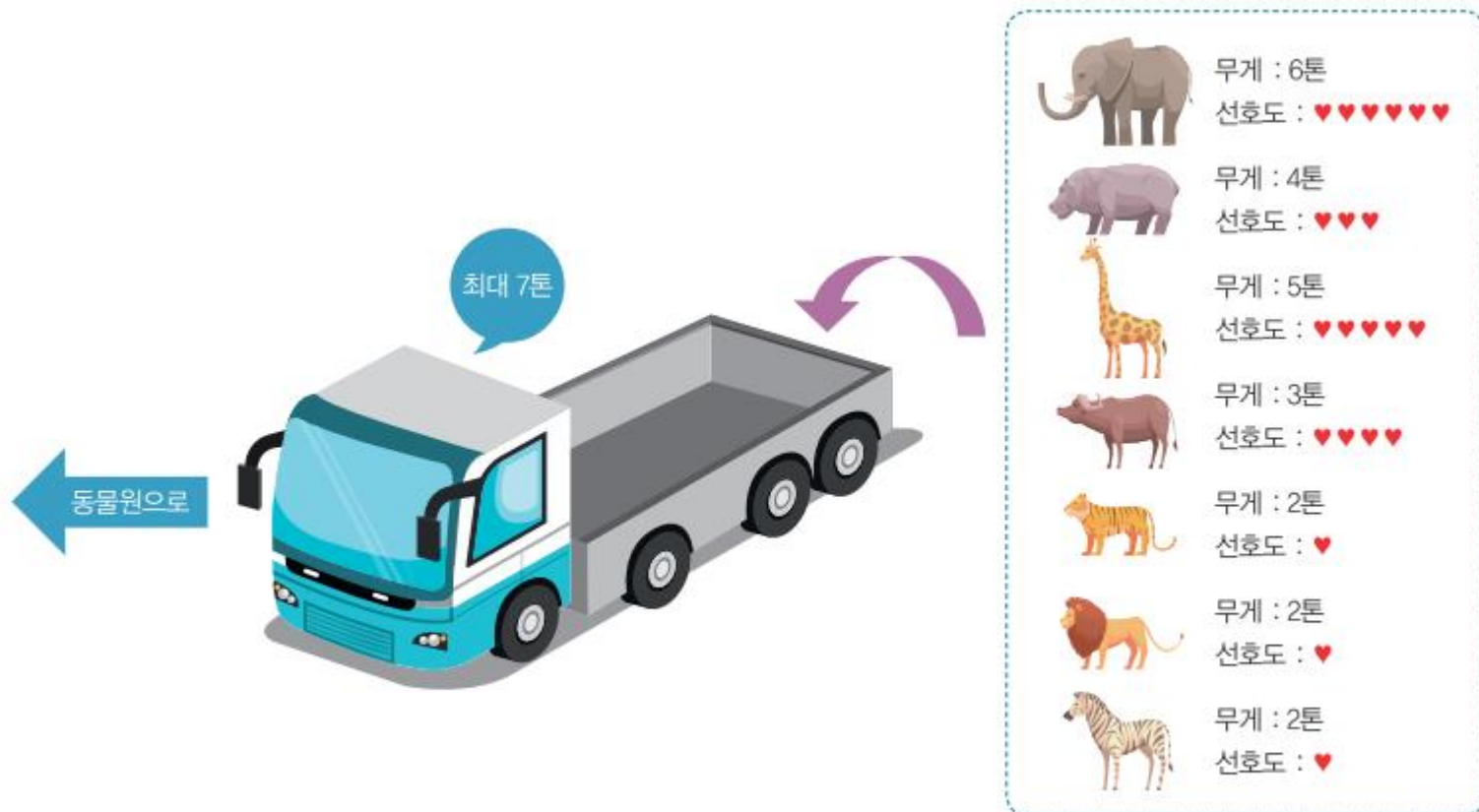


그림 1-7 동물원으로 동물 이동시키기

Section 02 알고리즘

- 동물 코끼리(6톤, 선호도 6), 하마(4톤, 선호도 3), 기린(5톤, 선호도 5), 물소(3톤, 선호도 4), 호랑이(2톤, 선호도 1), 사자(2톤, 선호도 1), 얼룩말(2톤, 선호도 1)

- 동물을 트럭에 태우기

- ① 선호도가 가장 높은 코끼리를 먼저 태운다. 현재 트럭의 무게는 6톤이 되었다.
- ② 다음으로 선호도가 높은 기린을 태운다. 그런데 트럭의 최대 중량 7톤이 넘어서 태울 수 없다.
- ③ 다음으로 선호도가 높은 물소를 태운다. 그런데 트럭의 최대 중량 7톤이 넘어서 태울 수 없다.
- ④ 다음으로 선호도가 높은 하마를 태운다. 그런데 트럭의 최대 중량 7톤이 넘어서 태울 수 없다.
- ⑤ 다음으로 선호도 높은 호랑이, 사자, 얼룩말을 차례대로 태우지만 최대 중량 7톤이 넘어서 태울 수 없다.
- ⑥ 최종적으로 코끼리 한 마리만 태워서 총 선호도는 6이 되었다.
- ⑦ 트럭이 동물원으로 출발한다.

동물이 자료, 울타리에 들어 있는 형태가 자료구조, 동물을 트럭에 태우는 방법이 알고리즘 → 자료구조와 알고리즘은 데이터와 그 데이터를 처리하는 방법의 관계

■ 자료구조

- 컴퓨터 분야에서 효율적으로 접근하고 수정할 수 있도록 자료를 구성·관리·저장하는 것

■ 알고리즘

- 컴퓨터 분야나 수학 등 관련 분야에서 어떤 문제를 해결하기 위해 정해진 일련의 단계적인 절차나 방법

SELF STUDY 1-1

[그림 1-7]에서 트럭 무게에 제한이 없다면, 동물을 태우는 경우가 몇 가지나 될지 생각해 보자. 예를 들어 트럭에 동물은 태우지 않는 경우(한 가지), 동물을 한 마리씩만 태우는 경우(일곱 가지), 동물을 두 마리씩 태우는 경우(21가지), 동물을 세 마리씩 태우는 경우 등을 생각해 보자.

■ 알고리즘 표현법

■ 일반 언어 표현

- 일반적인 자연어를 사용해서 설명하듯이 알고리즘을 표현
- 일반 사람이 이해하기 쉽게 표현할 수 있으나, 최종적으로 코드로 변경하는 데는 한계가 있음
- 어떤 알고리즘을 사용해야 할지 아이디어가 떠오르지 않는 시점에서, 생각 범위를 넓히는 단계 정도에 사용하면 무난



그림 1-8 일반 언어 표현

■ 알고리즘 표현법

■ 순서도를 이용한 표현

- 여러 종류의 상자와 상자를 이어 주는 화살표를 이용해서 명령 순서를 표현
- 간단한 알고리즘은 쉽게 표현할 수 있지만, 복잡한 알고리즘은 표현하기 어려운 경우가 많음

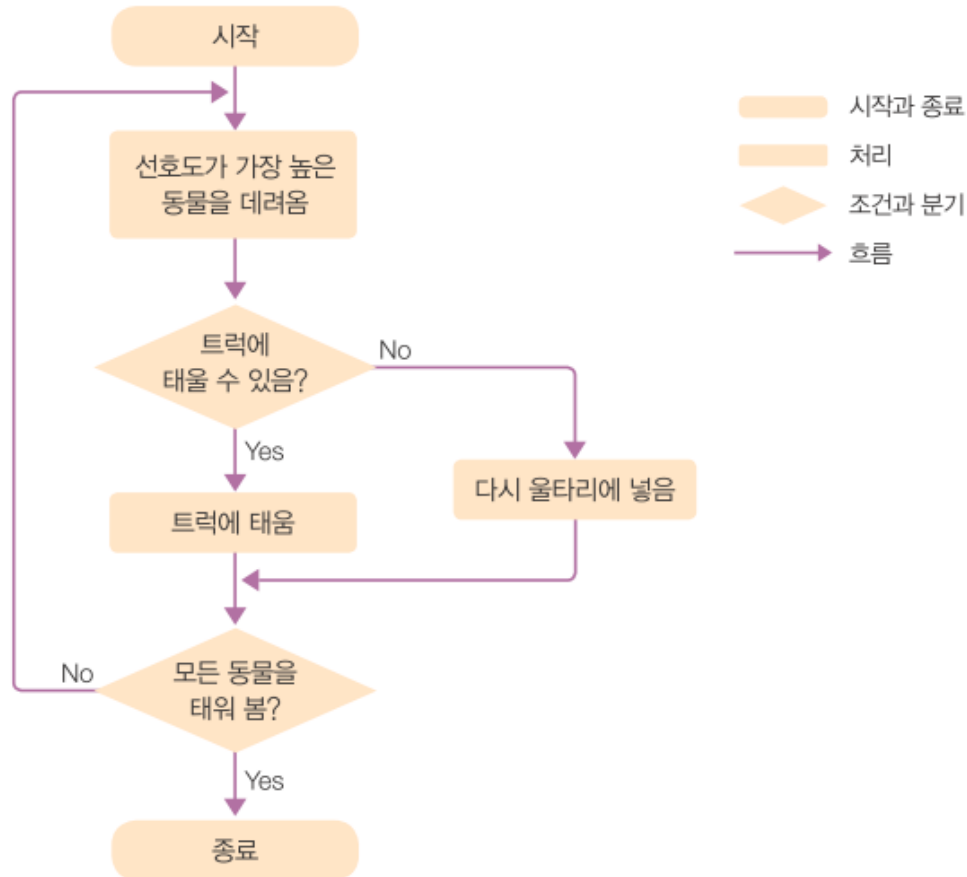


그림 1-9 순서도 표현

■ 알고리즘 표현법

■ 의사코드를 이용한 표현

- 프로그래밍 언어보다는 좀 더 인간의 언어에 가까운 형태
- 프로그램 코드와 일반 언어의 중간 형태
- 프로그램 코드를 직접 코딩하는 것보다 알고리즘을 좀 더 명확하게 정립하는 데 도움이 되고 코드에 설명을 달지 않아도 이해하는 데 무리 없음

```
: 레이블1
animal ← get animal
IF 트럭에 태울 수 있으면 THEN
    Put on a truck ← animal
ELSE
    Put in the fence ← animal
ENDIF

IF 아직 태워 볼 동물이 남았으면 THEN
    GOTO 레이블1
ELSE
    finish program
ENDIF
```

그림 1-10 의사코드로 표현

■ 알고리즘 표현법

- 프로그램 코드로 표현
 - 실제로 사용하는 프로그래밍 언어의 코드로 바로 작성 가능

```
def main() :  
    while(true)  
        animal = getAnimal()  
        if truck_weight + animal.weight <= 7 :  
            truck[put_count] = animal  
            put_count += 1  
        else :  
            fence[wait_count] = animal  
            wait_count += 1  
  
        if getAnimal() != None :  
            continue  
        else :  
            exit()  
  
    return
```

그림 1-11 프로그램 코드로 표현(파이썬 예)

- 세세한 부분을 표현하지 않고 전체적인 이미지를 표현한 것

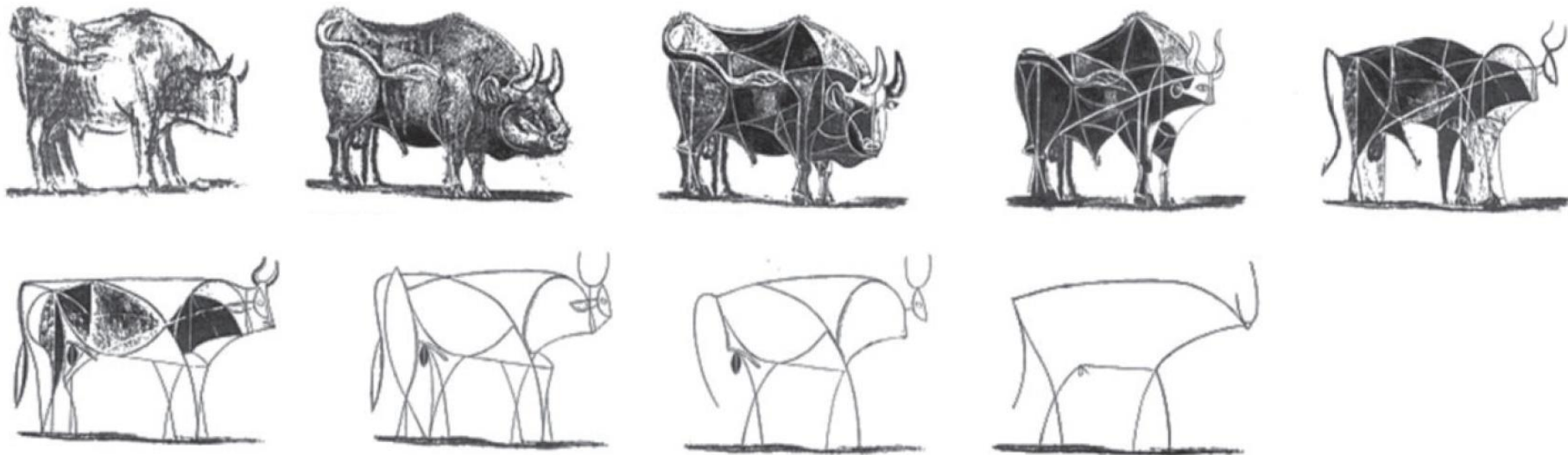


그림 1-12 추상적 묘사

Genetic Algorithm and Graph Partitioning

T. N. Bui, B. Moon • Published 1996 • Computer Science • IEEE Trans. Computers

Hybrid genetic algorithms (GAs) for the graph partitioning problem are described. The algorithms include a fast local improvement heuristic. One of the novel features of these algorithms is the schema preprocessing phase that improves GAs' space searching capability, which in turn improves the performance of GAs. Experimental tests on graph problems with published solutions showed that the new genetic algorithms performed comparable to or better than the multistart Kernighan-Lin algorithm and the simulated annealing algorithm. Analyses of some special classes of graphs are also provided showing the usefulness of schema preprocessing and supporting the experimental results. Collapse

그림 1-13 논문 초록

- 세부 사항에서 벗어나 추상적으로 정의한 데이터 타입
- 즉, 어떤 데이터 타입이 어떤 작업으로 이루어지는지만 표현한 것

(a) 원소를 첫 번째, 두 번째, ..., i 번째 원소로 가리킬 수 있는 자료구조

i 번째 자리에 새 원소 넣기

(b) 원소 x 찾기

i 번째 자리의 원소 삭제하기

그림 1-14 리스트의 ADT 표현 예

ADT '리스트'

작업:

- (a)
 - i번째 자리에 새 원소 넣기
 - 원소 x 찾기
 - i번째 자리의 원소 삭제하기

ADT '리스트'

작업:

- (b)
 - i번째 자리에 새 원소 넣기
 - 원소 x 찾기
 - i번째 자리의 원소 삭제하기
 - 비어 있는지 확인하기
 - 깨끗이 청소하기

그림 1-15 ADT 리스트

■ 알고리즘 표현법

■ 혼합 형태

- 간단한 알고리즘은 직접 코드로 작성
- 복잡한 알고리즘은 일반 언어, 의사코드, 순서도, 그림 등을 종합적으로 활용해서 표현
- 일반적으로 알고리즘의 수행시간은 최악경우 분석으로 표현
- **최악경우 분석**: '어떤 입력이 주어지더라도 알고리즘의 수행시간이 얼마 이상은 넘지 않는다'라는 상한(Upper Bound)의 의미
- **평균경우 분석**: 입력의 확률 분포를 가정하여 분석하는데, 일반적으로 균등분포(Uniform Distribution)를 가정
- **최선경우 분석**: 가장 빠른 수행시간을 분석
 - 최적(Optimal) 알고리즘을 찾는데 활용

등교 시간 분석

- 집을 나와서 지하철역까지는 5분, 지하철을 타면 학교까지 30분, 강의실까지는 걸어서 10분 걸린다
- **최선경우:** 집을 나와서 5분 후 지하철역에 도착하고, 운이 좋게 바로 열차를 탄 경우를 의미한다. 따라서 최선경우 시간은 $5 + 20 + 10 = 35$ 분
- **최악경우:** 열차에 승차하려는 순간, 열차의 문이 닫혀서 다음 열차를 기다려야 하고 다음 열차가 10분 후에 도착한다면, 최악경우는 $5 + 10 + 20 + 10 = 45$ 분

Section 03 알고리즘

- 평균 시간: 대략 최악과 최선의 중간이라고 가정했을 때, 40분이 된다.



(a) 최선 경우



(b) 최악 경우



(c) 평균 경우

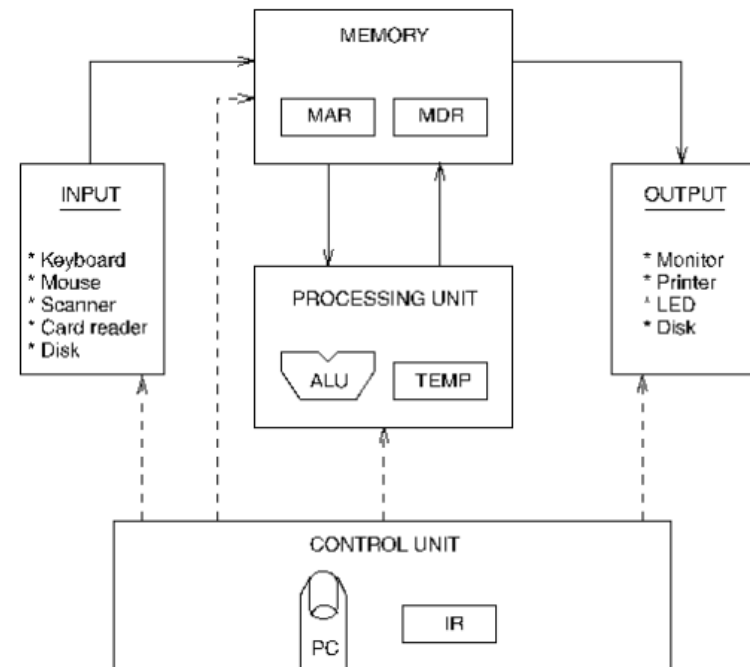
Section 03 알고리즘

■ 알고리즘의 성능 : 가상컴퓨터(RAM)+가상언어+가상코드

- 자료구조 + 알고리즘 > 코드(C, Python) : H/W, S/W 환경에 따라 다름
 - 가상컴퓨터 + 가상언어(Pseudo language)+가상코드(pseudo Code)
 - 가상컴퓨터 : Turing Machine > von Neumann Machine(Ram Access Machine)
 - RAM = CPU + Memory + 기본연산(단위시간 수행 연산)

Von Neumann Model

- 기본연산의 정의 : 1단위 시간
 - » 배정, 대입, 복사 : $A=B$ (1 시간)
 - » 산술연산 : $+, -, *, /$ (1 시간)
 - » 비교연산 : $>, >=, <, <=, ==, !=$ (1 시간)
 - » 논리연산 : AND, OR, NOT (1 시간)
 - » 비트연산 : bit-AND, OR, NOT (1 시간)



Section 03 알고리즘

■ 알고리즘의 성능 : 가상컴퓨터(RAM), 가상언어, 가상코드

■ 알고리즘 성능 측정

- 알고리즘을 소요 시간을 기준으로 알고리즘 성능을 분석 방법이 '시간 복잡도(Time Complexity)'

합계 ← 0

for 숫자가 1부터 100까지 반복 :
 합계 ← 합계 + 숫자

합계 출력

그림 1-12 알고리즘 1

합계 ← 0

합계 ← (1 + 100) * (100) / 2

합계 출력

그림 1-13 알고리즘 2

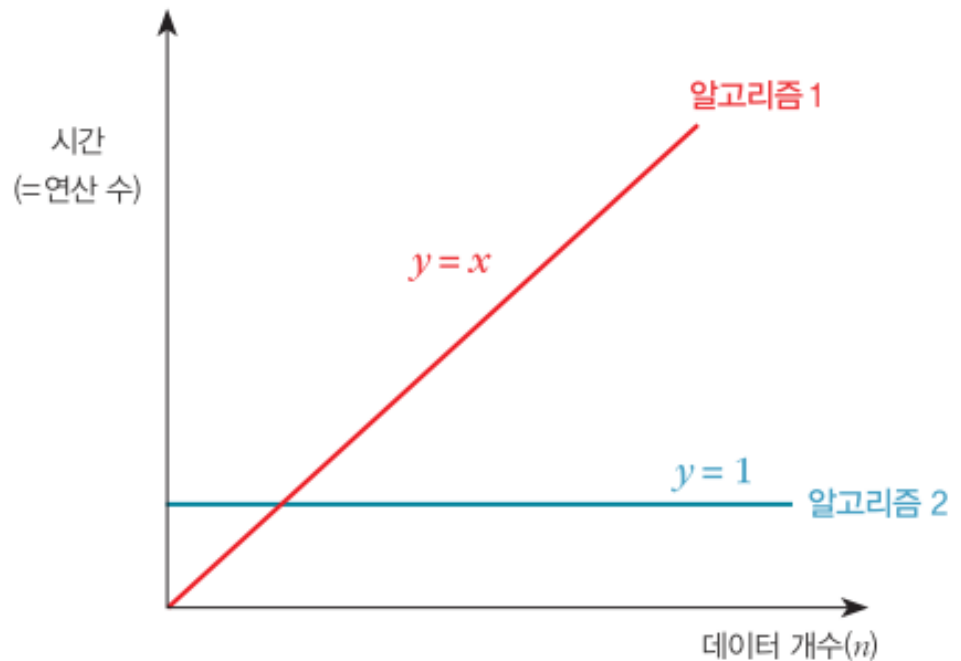


그림 1-14 두 알고리즘의 연산 시간 비교

알고리즘 1은 데이터 개수에 비례해서 시간이 늘어나지만, 알고리즘 2는 데이터 개수에 관계없이 시간이 동일하게 1

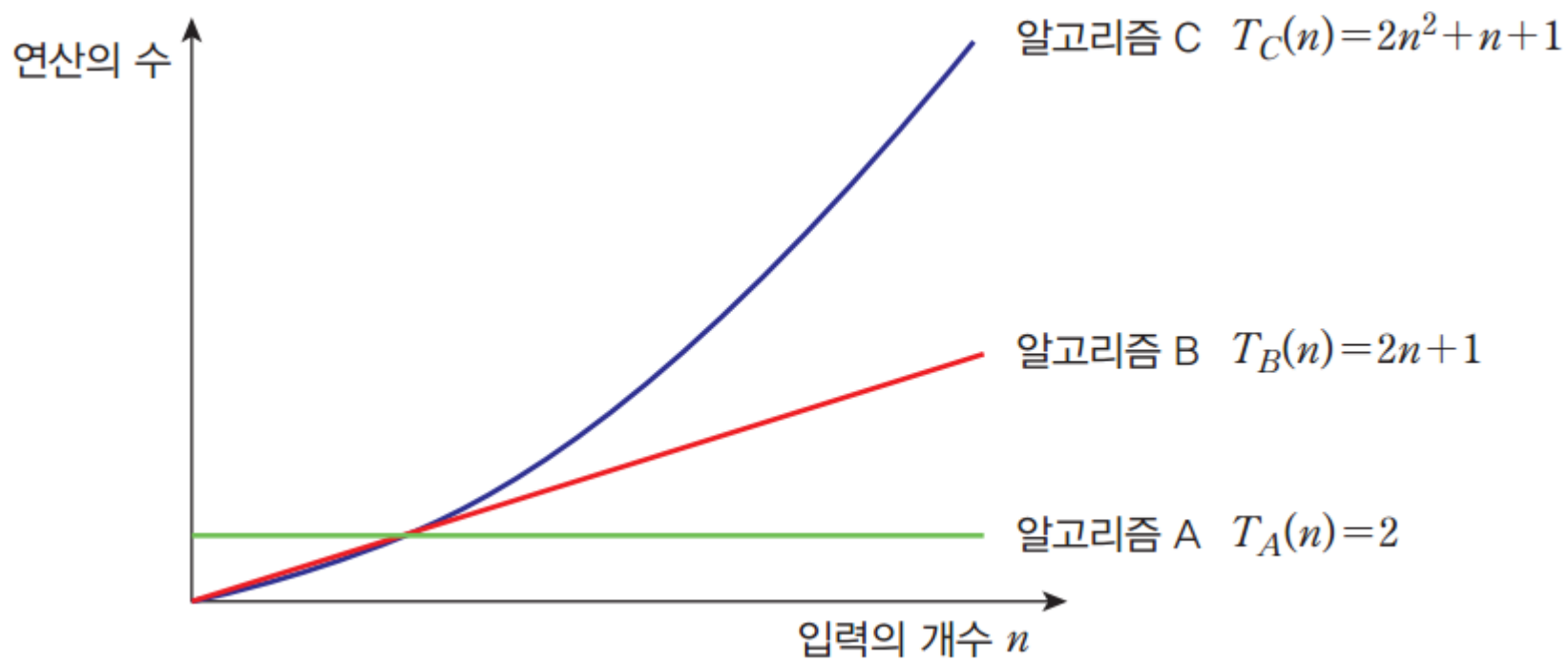
■ 알고리즘의 복잡도

n^2 을 구하는 문제

알고리즘	A	B	C
유사 코드	$\text{sum} \leftarrow n * n$	<pre>for i ← 1 to n do sum ← sum + n</pre>	<pre>for i ← 1 to n do for j ← 1 to n do sum ← sum + 1</pre>
연산 횟수	대입연산: 1 곱셈연산: 1	대입연산: $n+1$ 덧셈연산: n	대입연산: $n^2 + n + 1$ 덧셈연산: n^2
복잡도 함수	$T_A(n) = 2$	$T_B(n) = 2n + 1$	$T_C(n) = 2n^2 + n + 1$

Section 03 알고리즘

n^2 을 구하는 세 알고리즘 비교



빅오(Big-O) 표기법

- 차수가 가장 큰 항이 절대적인 영향
 - 다른 항들은 상대적으로 무시
 - 예: $T(n) = n^2 + n + 1$

- $n=1$ 일때 : $T(n) = 1 + 1 + 1 = 3$ (n^2 항이 33.3%)
- $n=10$ 일때 : $T(n) = 100 + 10 + 1 = 111$ (n^2 항이 90%)
- $n=100$ 일때 : $T(n) = 10000 + 100 + 1 = 10101$ (n^2 항이 99%)
- $n=1,000$ 일때 : $T(n) = 1000000 + 1000 + 1 = 1001001$ (n^2 항이 99.9%)

$n=100$ 인 경우

$$T(n) = n^2 + n + 1$$

99%

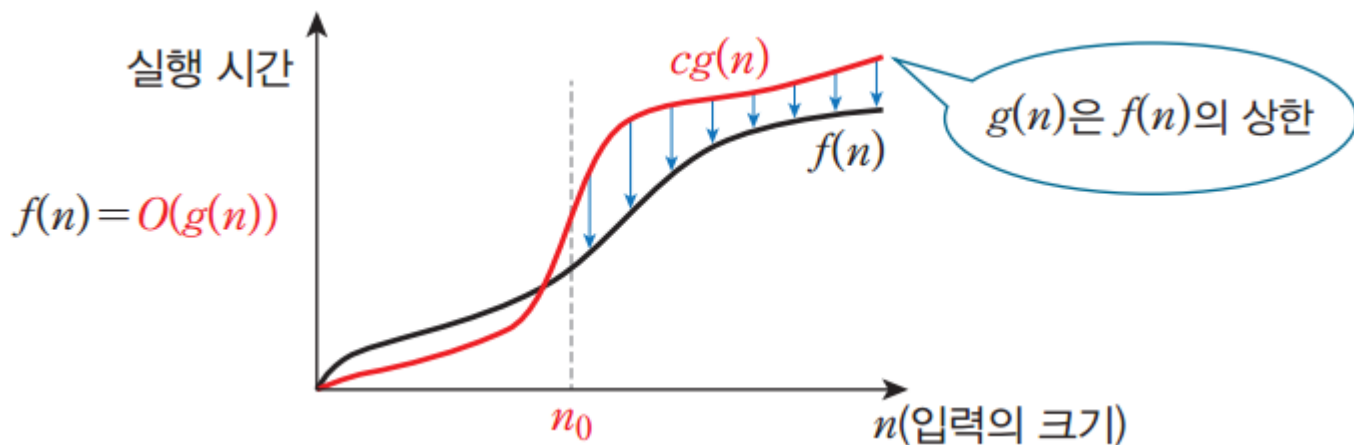
1%

빅오 표기법의 정의

정의 빅오 표기법

두개의 함수 $f(n)$ 과 $g(n)$ 이 주어졌을 때 모든 $n > n_0$ 에 대해 $|f(n)| \leq c |g(n)|$ 을 만족하는 상수 c 와 n_0 가 존재하면 $f(n) = O(g(n))$ 이다.

- 연산의 횟수를 대략적(점근적)으로 표기한 것



빅오 표기법의 예

- $f(n)=5$ 이면 $O(1)$ 이다. 왜냐하면 $n_0=1$, $c=10$ 일 때, $n \geq 1$ 에 대하여 $5 \leq 10 \cdot 1$ 이 되기 때문이다.
- $f(n)=2n+1$ 이면 $O(n)$ 이다. 왜냐하면 $n_0=2$, $c=3$ 일 때, $n \geq 2$ 에 대하여 $2n+1 \leq 3n$ 이 되기 때문이다.
- $f(n)=3n^2+100$ 이면 $O(n^2)$ 이다. 왜냐하면 $n_0=100$, $c=5$ 일 때, $n \geq 100$ 에 대하여 $3n^2+100 \leq 5n^2$ 이 되기 때문이다.
- $f(n)=5 \cdot 2^n + 10n^2+100$ 이면 $O(2^n)$ 이다.
왜냐하면 $n_0=1000$, $c=10$ 일 때, $n \geq 1000$ 에 대하여 $5 \cdot 2^n + 10n^2+100 < 10 \cdot 2^n$ 이 되기 때문이다.

[요약]

- 가장 빠르게 증가하는 항만을 고려하는 표기법입니다.
 - 함수의 상한만을 나타내게 됩니다.
- 예를 들어 연산 횟수가 $3N^3 + 5N^2 + 1,000,000$ 인 알고리즘이 있다고 합시다.
 - 빅오 표기법에서는 차수가 가장 큰 항만 남기므로 $O(N^3)$ 으로 표현됩니다.

Section 03 알고리즘

■ 알고리즘의 성능

■ 알고리즘 성능 표기

: 최악의 경우의 기본연산 횟수

- 빅-오 표기법(Big-Oh Notation)으로 $O(f(n))$ 형태
- 대표적인 함수는 $O(1)$, $O(\log n)$, $O(n)$, $O(n \log n)$, $O(n^2)$, $O(n^3)$, $O(2^n)$ 정도가

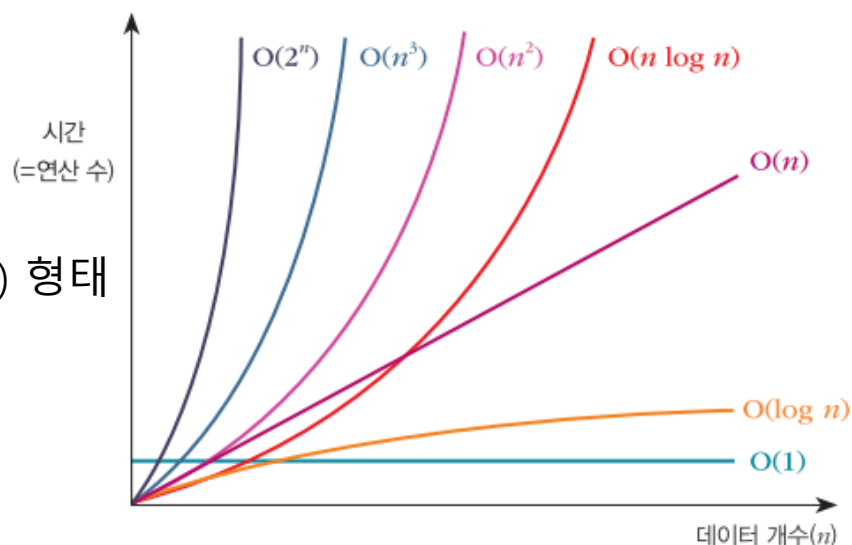


그림 1-15 시간 복잡도 함수의 그래프

표 1-1 입력 데이터에 따른 알고리즘 연산 횟수

입력 데이터(n)	알고리즘 연산 횟수					
	$O(\log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$	$O(n^3)$	$O(2^n)$
10	4	10	40	100	1000	1024
30	5	30	150	900	27000	1.07E+09
50	6	50	300	2500	125000	1.13E+15
100	7	100	700	10000	1000000	1.27E+30
1,000	10	1000	10000	1000000	1E+09	1.1E+301
10,000	14	10000	140000	1E+08	1E+12	
100,000	17	100000	1700000	1E+10	1E+15	
1,000,000	20	1000000	20000000	1E+12	1E+18	

Section 03 알고리즘

■ 자료구조와 알고리즘과 프로그램의 관계

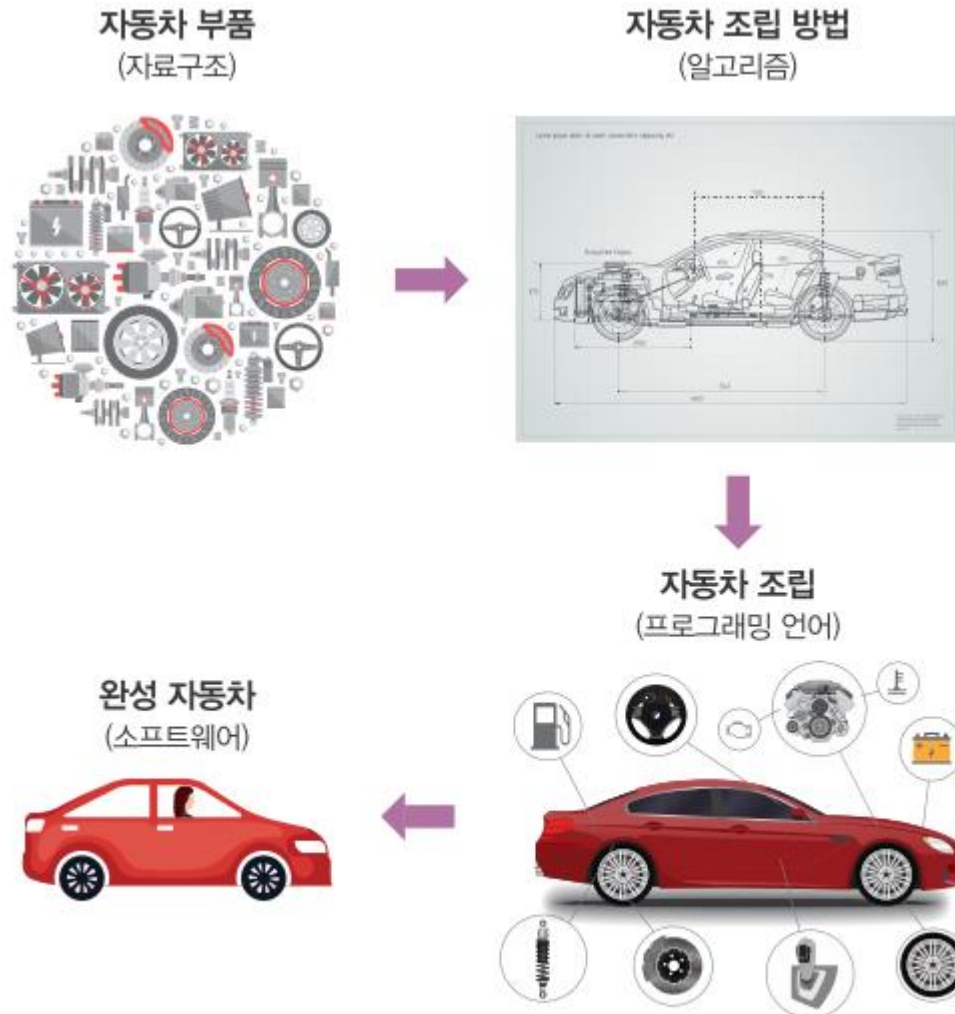


그림 1-16 자료구조, 알고리즘, 프로그래밍 언어, 소프트웨어의 관계

SELF STUDY 1-2

11~12장에서 데이터를 순서대로 나열하는 정렬 알고리즘을 배운다. 그중 선택 정렬은 $O(n^2)$ 의 연산 횟수가 필요하고, 퀵 정렬은 평균 $O(n \log n)$ 의 연산 횟수가 필요하다. 데이터가 15개라면 각각 얼마의 연산 횟수가 필요할까?

$$15^2 = 225$$

$$15 \cdot \log 15 = 58.60336$$

Section 03 파이썬 소개와 설치

■ 파이썬 소개

- 배우기도 쉽고 결과도 바로 확인할 수 있어 초보자에게 적합한 프로그래밍 언어
- 귀도 반 로섬(1956년~)이라는 프로그래머가 C 언어로 제작해 1991년에 공식으로 발표
- 사전적인 의미는 비단뱀으로 로고도 파란색과 노란색 비단뱀 두 마리가 서로 얹혀 있는 형태



그림 1-17 파이썬 로고

- [파이썬 소개와 설치 동영상 강좌](#)

Section 03 파이썬 소개와 설치

■ 파이썬 다운로드

- <http://www.python.org/>에 접속 → [Downloads]-[Python 3.x.x] 클릭
→ 설치 파일인 python-3.x.x.exe를 원하는 위치에 저장

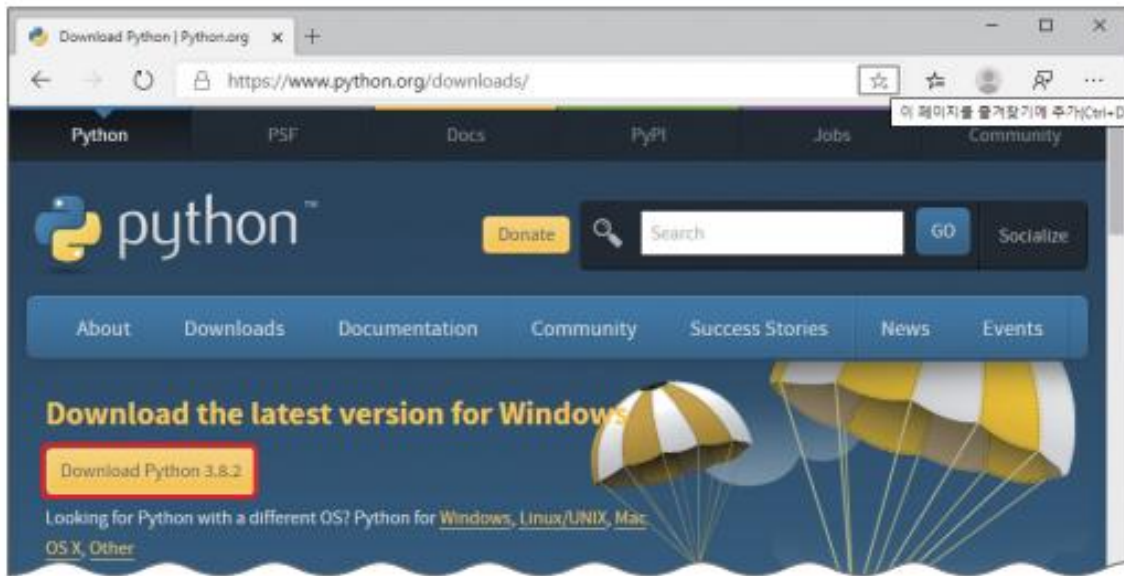


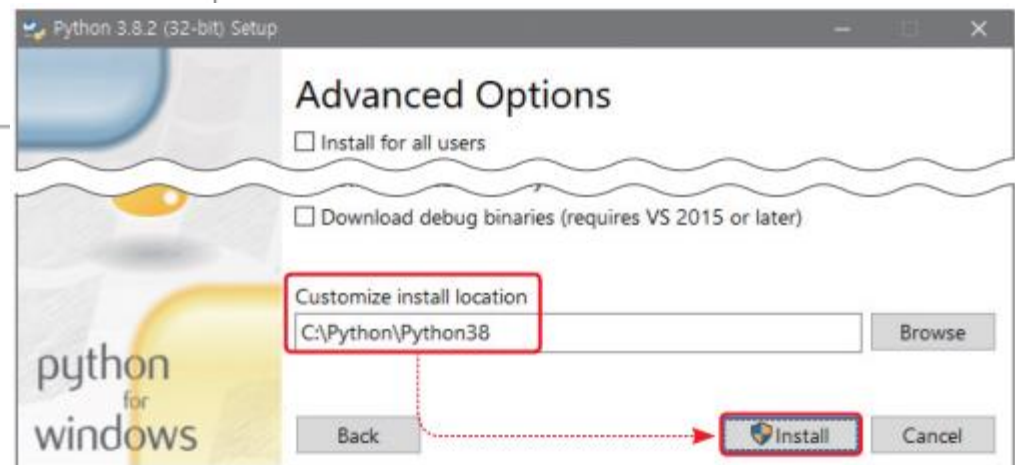
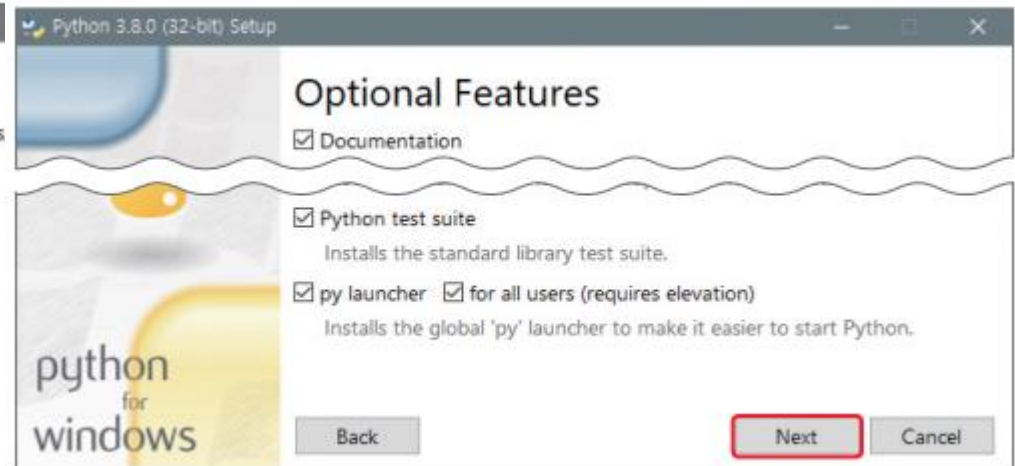
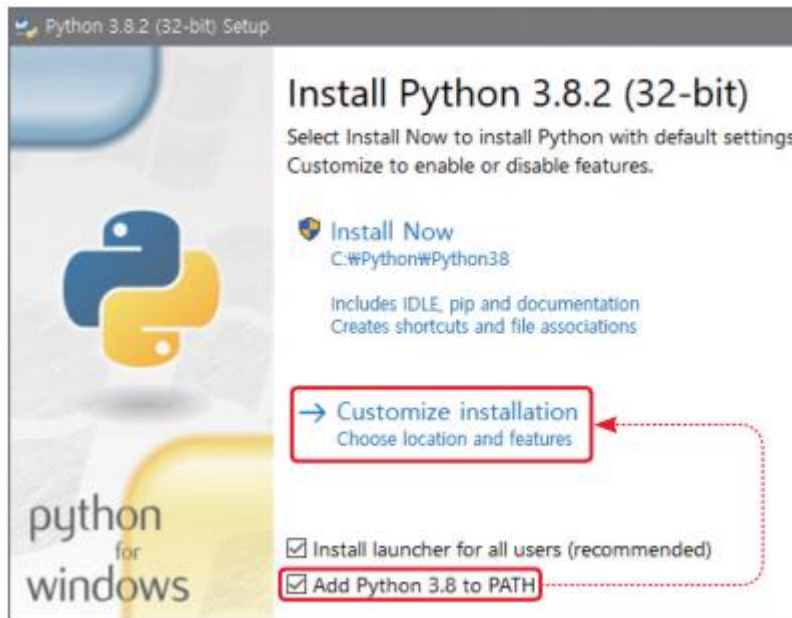
그림 1-18 <https://www.python.org> 웹 사이트에서 다운로드

TIP • 파이썬 3.x는 32bit용과 64bit용이 있다. 어떤 것을 사용해도 상관없지만, 이 책은 호환성이 좋은 32bit용을 사용한다. 참고로 파일명이 python-3.8.2.exe인 것은 32bit용이고, python-3.8.2-amd64.exe인 것은 64bit용이다.

Section 03 파이썬 소개와 설치

■ 파이썬 설치

- python-3.x.x.exe를 더블클릭 실행 → Add Python 3.8 to PATH에 체크
→ <Install Now> 버튼 클릭(<Customize Installation> 버튼으로 설치 폴더 변경 가능)
→ 설치 진행 → 설치를 마치면 <Close> 버튼 클릭



Section 03 파이썬 소개와 설치

■ 파이썬 설치

- python-3.x.x.exe를 더블클릭 실행 → Add Python 3.8 to PATH에 체크
→ <Install Now> 버튼 클릭(<Customize Installation> 버튼으로 설치 폴더 변경 가능)
→ 설치 진행 → 설치를 마치면 <Close> 버튼 클릭

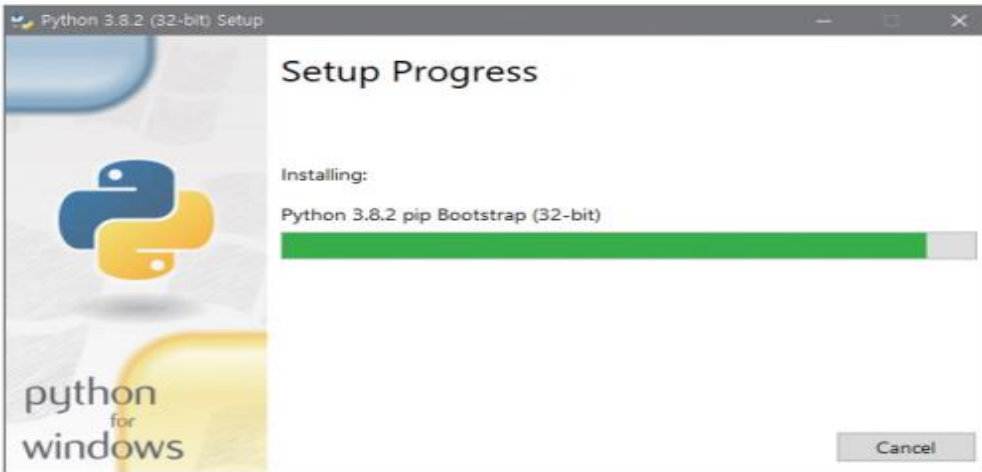


그림 1-19 파이썬 설치

Section 03 파이썬 소개와 설치

■ 파이썬 기본 사용법

■ 파이썬 실행

- 윈도우의 <시작> 버튼

→ [모든 프로그램]-[Python 3.6]-[IDLE (Python 3.6 32-bit)] 메뉴 선택

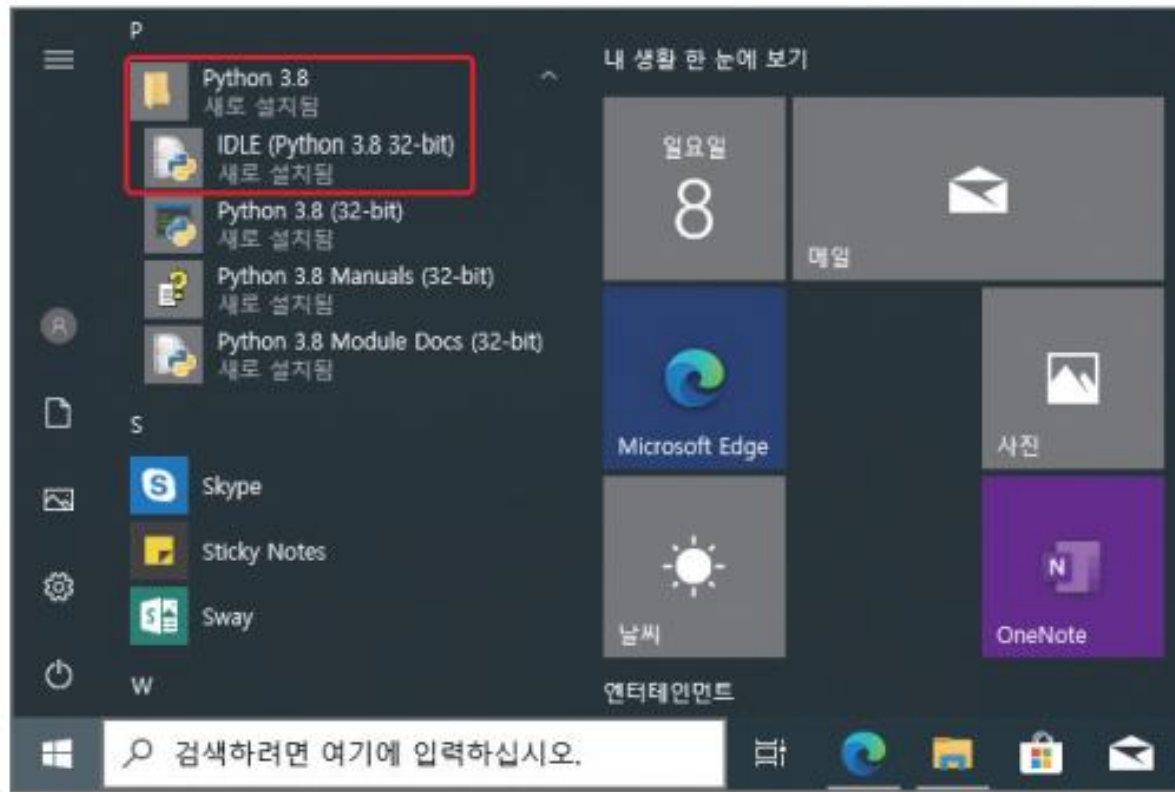
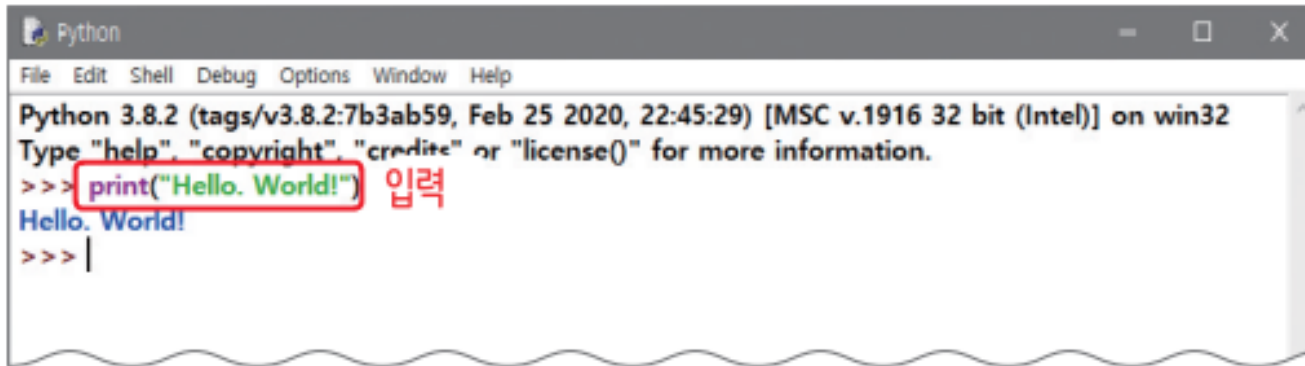


그림 1-20 파이썬 개발 환경 실행

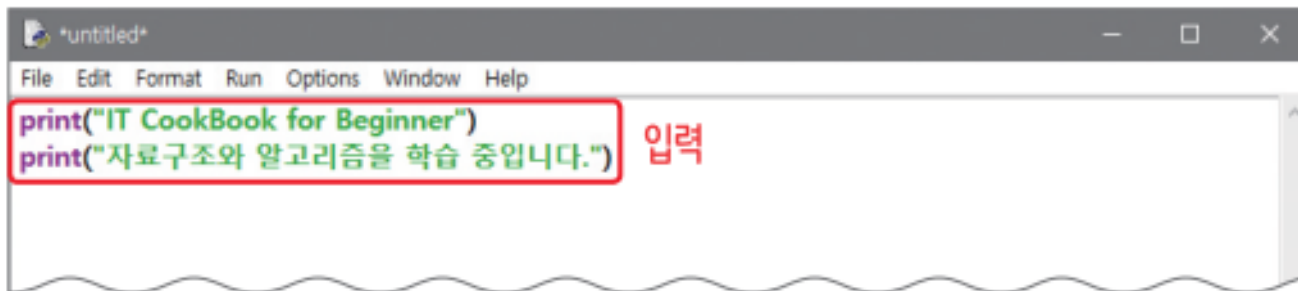
Section 03 파이썬 소개와 설치

- 파이썬 기본 사용법
 - 파이썬 코드 입력과 실행 예



A screenshot of a Python 3.8.2 Shell window. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Window', and 'Help'. The main text area shows the Python version and build information: 'Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32'. Below this, it says 'Type "help", "copyright", "credits" or "license()" for more information.' The prompt '>>>' is followed by the code 'print("Hello. World!")' which is highlighted with a red box. To the right of the code is the Korean text '입력' (Input). The output 'Hello. World!' is displayed below the code. The prompt '>>>' is followed by a vertical bar '|'.

그림 1-21 한 행씩 파이썬 코딩 및 실행



A screenshot of a Python 3.8.2 Shell window. The window has a menu bar with 'File', 'Edit', 'Format', 'Run', 'Options', 'Window', and 'Help'. The main text area shows the Python version and build information: 'Python 3.8.2 (tags/v3.8.2:7b3ab59, Feb 25 2020, 22:45:29) [MSC v.1916 32 bit (Intel)] on win32'. Below this, it says 'Type "help", "copyright", "credits" or "license()" for more information.' The prompt '>>>' is followed by two lines of code: 'print("IT CookBook for Beginner")' and 'print("자료구조와 알고리즘을 학습 중입니다.")'. Both lines are highlighted with a red box. To the right of the code is the Korean text '입력' (Input). The output 'IT CookBook for Beginner' and '자료구조와 알고리즘을 학습 중입니다.' is displayed below the code. The prompt '>>>' is followed by a vertical bar '|'.

그림 1-22 여러 행의 파이썬 코딩 및 실행

Section 03 파이썬 소개와 설치

■ 파이썬 기본 사용법

■ 파이썬 소스 파일 저장

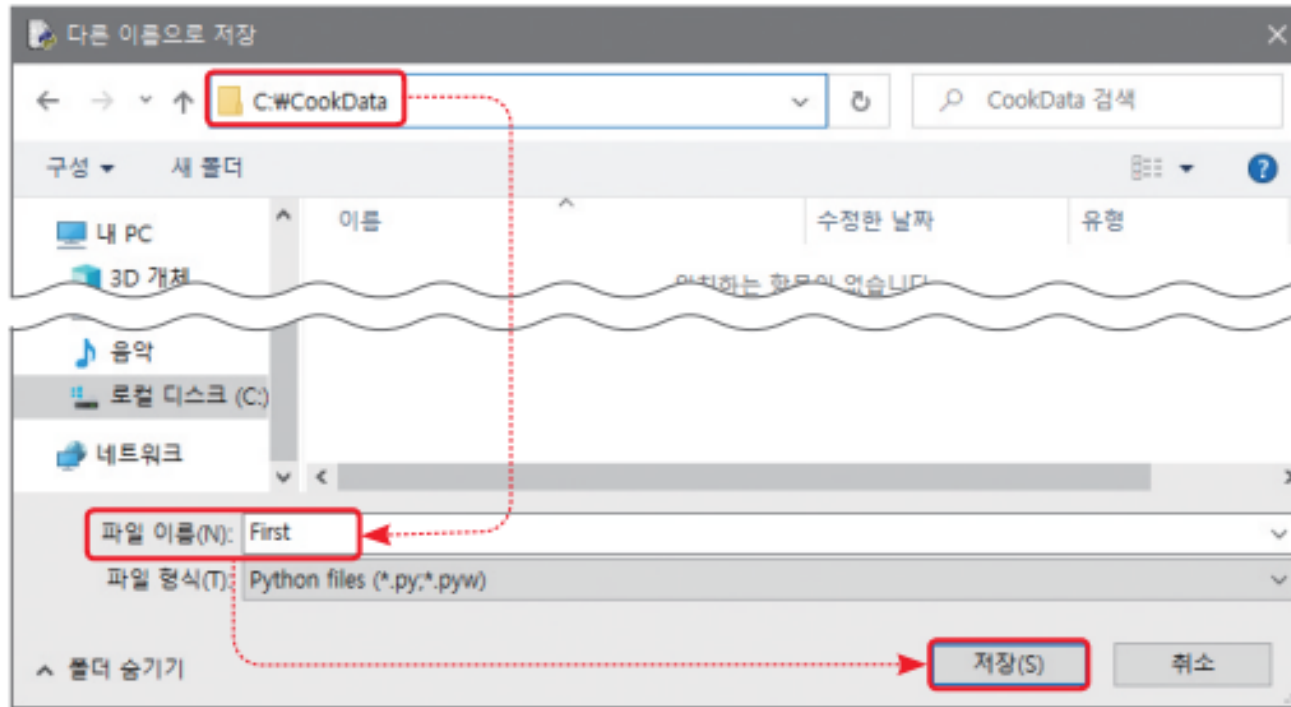


그림 1-23 파이썬 소스 파일 저장

Section 03 파이썬 소개와 설치

■ 파이썬 기본 사용법

■ 파이썬 소스 파일 실행

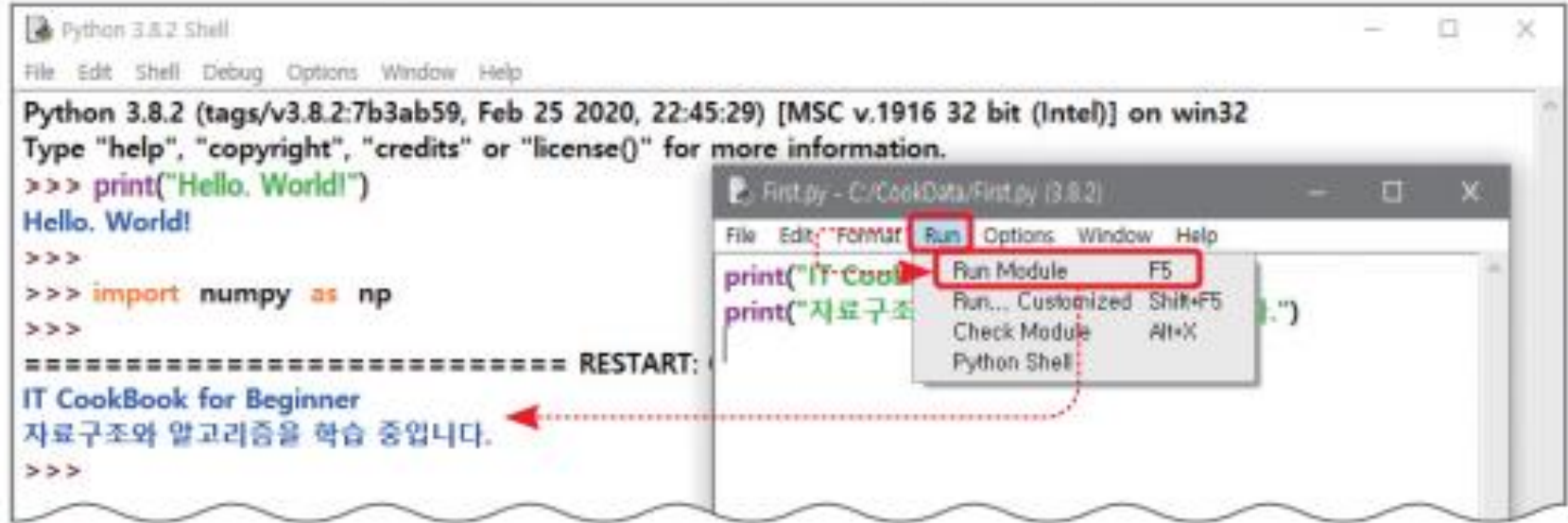


그림 1-24 파이썬 소스 파일 실행

■ 파이썬 종료

- [File]-[Exit]

Section 03 파이썬 소개와 설치

SELF STUDY 1-3

다음 결과가 나오도록 파이썬 코드를 작성한 후 Second.py로 저장하고 실행해 보자.

실행 결과

안녕하세요?

자료구조는 어렵지 않습니다.

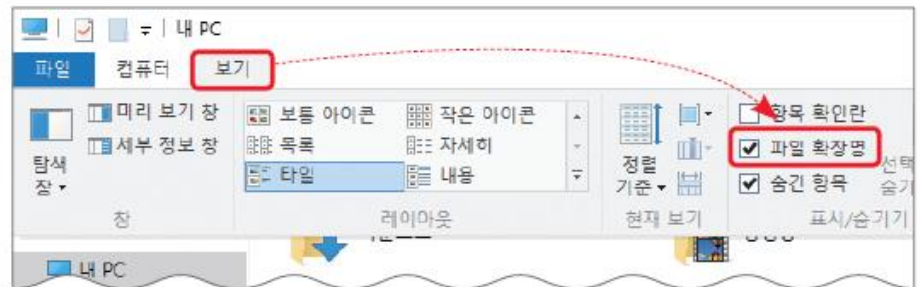
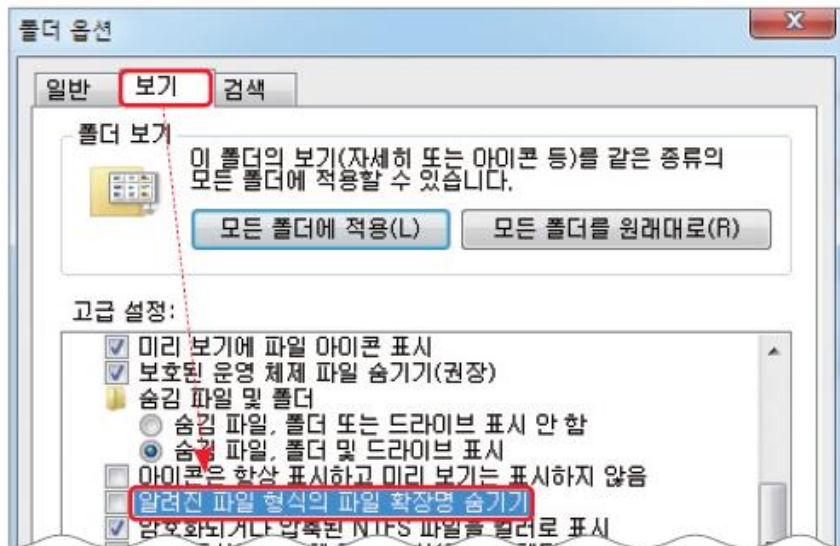
열공하겠습니다. ^^

```
print("안녕하세요?")  
print("자료구조는 어렵지 않습니다.")  
print("열공하겠습니다.^^")
```

Section 03 파이썬 소개와 설치

■ (참고) 윈도우의 확장명 표시

- 윈도우 10은 파일 탐색기 실행→[보기] 메뉴 선택 → '파일 확장명' 체크
- 윈도우 7은 [구성]-[폴더 및 검색 옵션] 메뉴 선택 → [폴더 옵션] 대화상자의 [보기] 탭 클릭 → '알려진 파일 형식의 파일 확장명 숨기기'의 체크 표시 해제 → [확인] 버튼



미리 생각해 두는 중간고사

집에서 학교오면 방법과 학교에서 집으로 가는 방법을 최악, 최선, 평균을 고려하여 알고리즘을 작성하시고, 배운 범위에서 자료구조를 사용하여 프로그램 하고 설명하시오.

Thank You
