

소프트웨어와 인공지능 (유형3)

Ch 2. 프로그램 설계, 알고리즘과 자료구조, 파이썬 프로그램의 기본 구조



영남대학교 정보통신공학과
교수 김 영 탁

(Tel : +82-53-810-2497; E-mail : ytkim@yu.ac.kr)

Outline

- ◆ 알고리즘의 표현, 유사코드(pseudo code)
- ◆ 대표적인 알고리즘
- ◆ 대표적인 자료구조
- ◆ 파이썬 프로그램의 기본 구성
- ◆ 파이썬 프로그램의 식별자 (identifier), 기본 연산
- ◆ 파이썬 프로그램의 입력과 출력
- ◆ 파이썬 프로그램의 실행 제어 기초 – 조건문과 반복문
- ◆ 객체지향형 프로그래밍 개요 (기본 용어)
- ◆ 터틀 그래픽 개요 – 별 (star) 그리기



알고리즘의 표현과 유사코드

알고리즘이란?

◆ 알고리즘 (Algorithm)

- 컴퓨터 프로그램으로 문제를 해결하는 절차와 방법에 대한 정리
- 큰 문제를 작은 문제로 나누어 해결하는 절차도 포함됨
- 문제해결에 포함되는 절차와 방법을 가능한 한 세부적으로 표현하여야 함
- 순서도 (flow chart)나 유사코드 (pseudo code)로 표현하여 쉽게 수정 및 편집할 수 있게 함
- 실제 프로그램 소스코드 작성 이전에 알고리즘과 자료구조에 대한 확실한 구상이 있어야 함



알고리즘의 설계

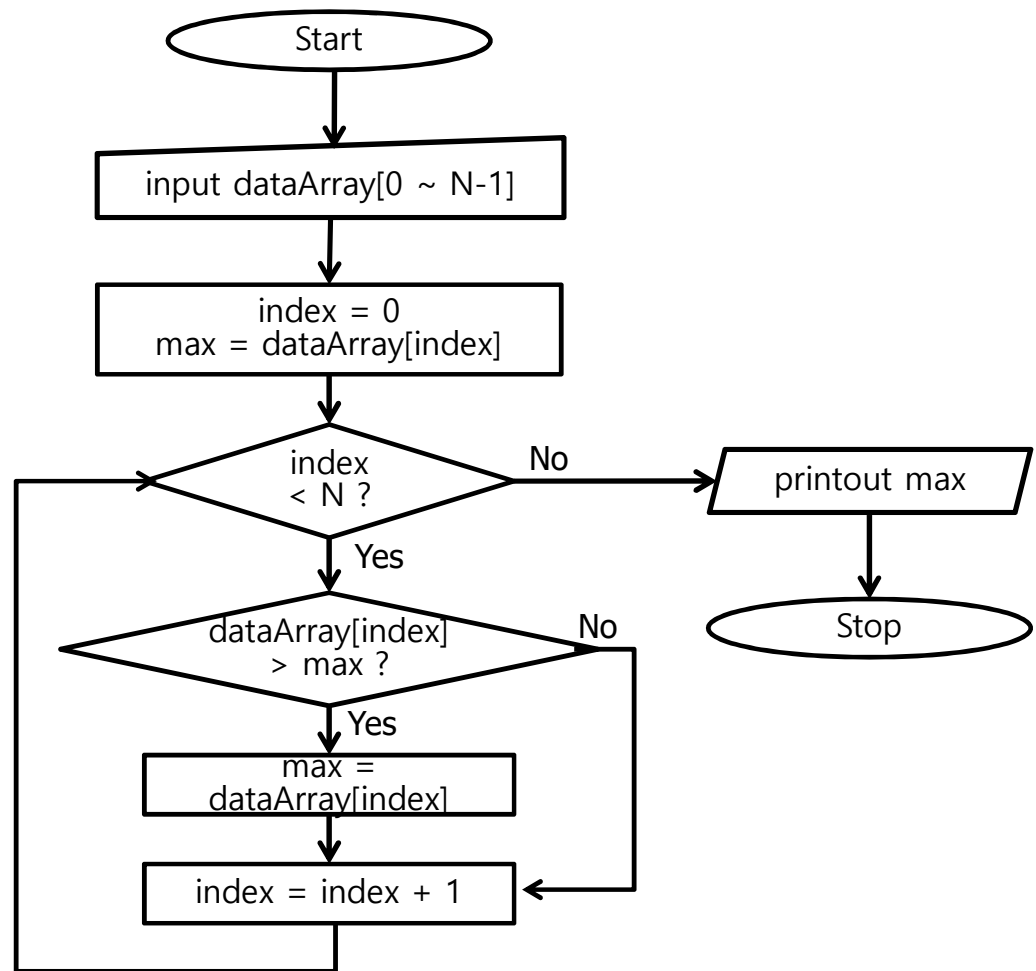
◆ 알고리즘의 설계

- 프로그램/소프트웨어 개발에서 가장 핵심적인 부분
- 어떤 절차와 단계를 밟아서 어떤 순서로 작업을 처리할 것인지를 설계
- 순서도 (flow chart) 또는 유사 코드/의사코드 (pseudo code)로 표현
- 알고리즘은 프로그래밍 언어와는 무관
- 알고리즘은 원하는 결과를 얻기 위하여 밟아야 하는 단계에 집중적으로 초점을 맞추는 것

순서도 (flow chart)

◆ 순서도 (Flow Chart)

- 프로그램에서의 논리순서 또는 작업순서 등을 그래픽으로 표현하기 위한 형식
- 알고리즘이 복잡하면 그래픽으로 표현하기가 힘들어진다.



순서도에서 사용되는 기호와 의미

◆ Flow Chart Symbols

순서도 기호	의미	순서도 기호	의미
	단말(단자): 알고리즘의 시작과 끝을 표시		순서 흐름선 (flow line)
	데이터의 입력과 출력		판단, 분기
	데이터의 수동입력		지연 (delay)
	준비		논리접합, 가산접합
	데이터 처리		정렬, 분류
	미리 준비된 처리 (서브 루틴)		대조
	문서, 서류		다른 페이지 연결자
	디스플레이		페이지 내 연결자

유사코드/의사코드 (Pseudo Code)

◆ 유사코드/의사코드 (Pseudo Code)

- 프로그래밍언어가 가지고 있는 모든 문법적인 상세한 내용들을 다 포함하는 것이 아니라 기본적인 골격만 이해할 수 있도록 소스코드와 비슷하지만 간략화된 표현 방법

◆ 유사코드 사용의 장점

- 프로그램의 구조와 실행 상세 내용을 잘 표현할 수 있으며, 프로그램 소스코드 작성 단계 이전에 **설계 문서**로서 작성
- 프로그램의 실행 순서를 조리있게 **설명**할 수 있고, 프로그램 구조와 실행 내용을 검토할 수 있음
- 프로그램의 실행 내용과 절차를 procedure(), sub_procedure() 등으로 **그룹화**하여 **체계적/계층적으로 설계**할 수 있음
- 프로그램의 구조와 실행 내용의 설계 문서를 일반 문서 작성기로 **쉽게 변경 및 보완**할 수 있음

유사코드 작성 예

◆ 유사코드 작성 예 – Draw_Rectangle()

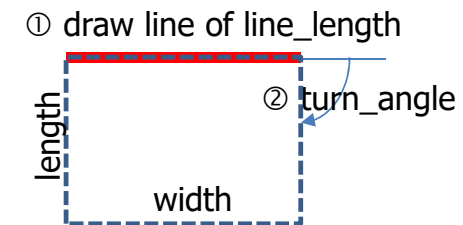
Procedure Draw_Rectangle()

```
1.  /* input argument: None */
2.  /* output result: Rectangle drawn on canvas */

3.  width = 200
4.  length = 100

5.  draw_line(width)
6.  turn_right(90)
7.  draw_line(length)
8.  turn_right(90)
9.  draw_line(width)
10. turn_right(90)
11. draw_line(length)
12. turn_right(90)

13. end procedure /* end of procedure Draw_Rectangle() */
```



유사코드 (pseudo code) 작성법

◆ Pseudo code 작성에서 많이 사용되는 단어들

- 입력(Input): READ, OBTAIN, GET
- 출력(Output): PRINT, DISPLAY, SHOW
- 계산(Compute): COMPUTE, CALCULATE, DETERMINE
- 초기화(Initialize): SET, INIT
- 요소를 추가(Add one): INCREMENT, BUMP
- 선형적으로 증가할 때(linear progression): SEQUENCE
- 반복: WHILE, FOR, LOOP
- 조건문: IF-THEN-ELSE, IF-ELIF-ELSE
- 마지막에 조건문이 있는 반복문: REPEAT-UNTIL
- IF-THEN-ELSE 대신 조건 분기처리: CASE
- 부울 : TRUE / FALSE
- 그 외 : RETURN, BEGIN, EXCEPTION, END



유사코드 (pseudo code) 작성법

◆ Pseudo code 작성법

- 유사코드 작성에서는 C, C++, Java, Python 등의 프로그래밍 언어 문법을 정확하게 준수하지 않아도 허용되나, 국제적으로 많이 통용되는 용어와 표현을 사용하는 것이 권장됨
 - 국제 학술 논문
 - 국제 협력 작업
- 시스템 소프트웨어의 핵심구조와 핵심 내용을 잘 표현할 수 있게 작성
- 조건문, 반복문은 if, while 다음에 조건식을 표시
- 복잡한 조건식 표현을 위하여 괄호()를 사용
- 조건문, 반복문 등이 수행되는 블록 단위 표현
 - ‘:’ 다음 줄에 들여쓰기로 구분 – Python에서 사용
 - 또는, 중괄호 {}로 표현 – C, C++, Java에서 사용
- 프로그램 실행을 체계적으로 구분하여 procedure와 sub_procedure의 계층적인 구조로 작성
- 각 procedure에서 실행되는 내용을 이해할 수 있는 이름을 사용
 - draw_line(), turn_right()
 - draw_rectangle()
 - print(), input()

유사코드 작성 예

◆ 유사코드의 예 – DrawStar()

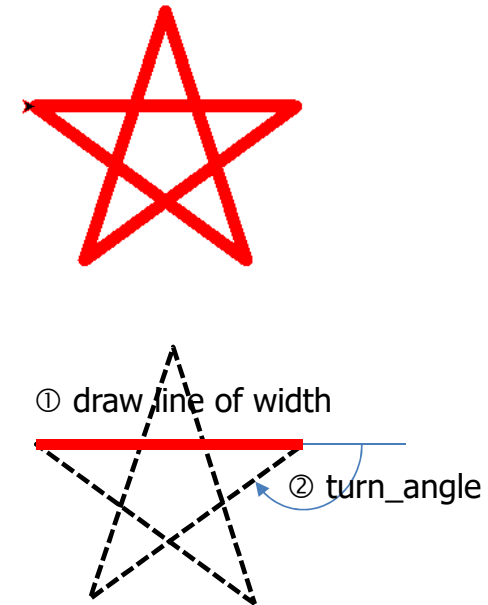
Procedure Draw_Star()

```
1.  /* input argument: None */
2.  /* output result: Star drawn on canvas */

3.  width = 200
4.  turn_angle = (360 * 2) / 5
5.  count = 0
6.  move_to_start_position()

7.  while (count < 5) :
8.      draw_line(width)
9.      turn_right(turn_angle)
10.     count = count + 1

11. end procedure /* end of procedure Draw_Star() */
```



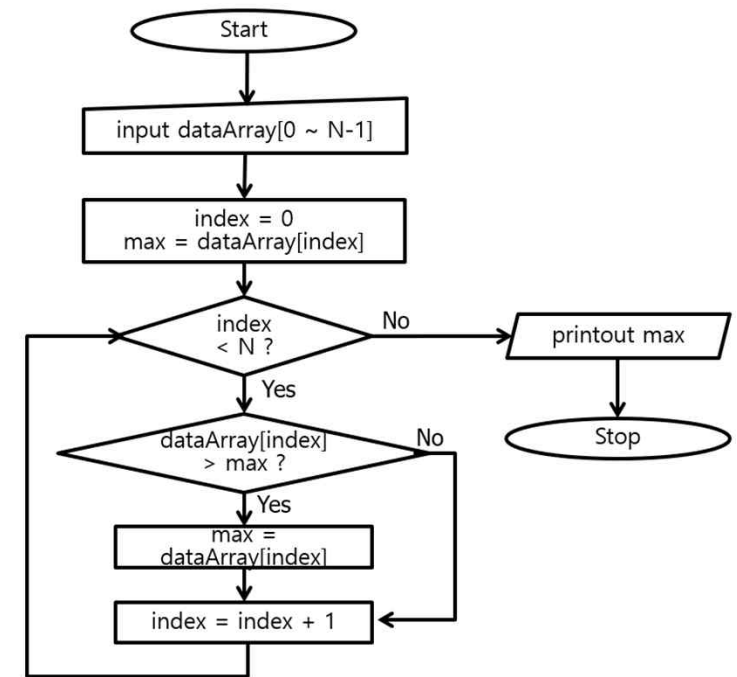
유사코드 (Pseudo Code) 작성

◆ Find_Max()

- N개의 원소를 가진 배열에서 가장 큰 값 (최대값)을 찾아 출력

Procedure **Find_Max(dataArray, N)**

1. /* input argument: dataArray[0..N-1]; */
2. /* array of data with N elements */
3. /* output result: the maximum of the given data array */
4. index = 0 /* define and initialize index */
5. max = dataArray[index]
6. while (index < N):
7. if (dataArray[index] > max):
8. max = dataArray[index]
9. index = index + 1
10. printout max
11. end procedure /* end of Algorithm Find_Max() */



유사코드를 사용한 알고리즘의 변경

◆ Find_MaxMinAvg()

- N개의 원소를 가진 배열에서 최대값, **최소값**, **평균값**을 출력

```
Procedure Find_MaxMinAvg(dataArray, N)
    /* input argument: dataArray[0..N-1]; */
    1. /* array of data with N elements */
    2. /* output result: the maximum, minimum and average values */
    /* of the given data array */
    3. index = 0 /* define and initialize index */
    4. sum = avg = 0.0
    5. max = min = dataArray[index]
    6. while (index < N):
    7.     if (dataArray[index] > max):
    8.         max = dataArray[index]
    9.     if (dataArray[index] < min):
    10.        min = dataArray[index]
    11.    sum = sum + dataArray[index]
    12.    index = index + 1
    13. avg = sum / N
    14. printout max, min, avg
    15. end procedure /* end of Algorithm Find_MaxMinAvg() */
```

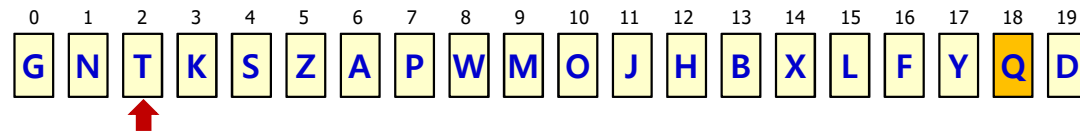
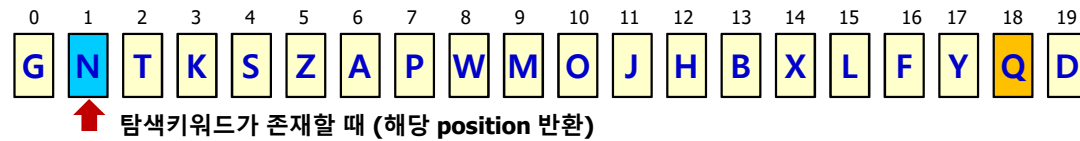
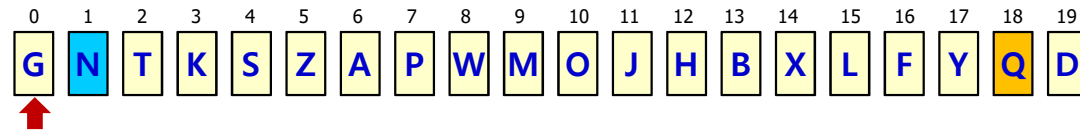


대표적인 알고리즘

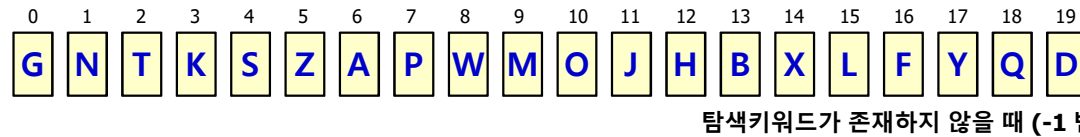
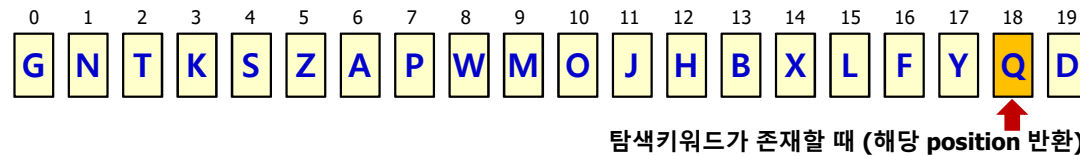
대표적인 알고리즘

알고리즘	주요 기능
순차 탐색	배열이나 연결형리스트에 포함되어 있는 항목을 차례대로 탐색
이진 탐색	배열에 포함되어 있는 항목의 탐색 구간을 절반씩 줄여가며 탐색
선택 정렬 (selection sort)	배열에 포함되어 있는 항목의 탐색 구간을 하나씩 줄여가며, 가장 작은(또는 큰) 항목을 찾아 그 구간의 맨 앞에 두는 동작을 반복하여 전체 항목들을 정렬. 배열에
퀵 정렬 (quick sort)	배열에 포함되어 있는 항목의 탐색 구간을 중간 지점의 피벗 (pivot)을 중심으로 큰 항목들과 작은 항목들로 구분하여 정돈하며, (평균적으로) 절반씩 줄어드는 탐색 구간에 대하여 재귀 함수 실행
병합정렬 (merge sort)	입력 배열을 같은 크기의 2개의 부분 배열로 분할 (partition)한 후, 각 부분 배열을 정렬하며, 각 부분 배열의 크기가 충분히 작지 않은 경우 분할 정복 (divide-and-conquer) 방법을 반복 적용함. 각 부분 배열의 정렬이 완료되면 이 정렬된 부분 배열들을 하나의 배열로 합병함. 추가적인 배열이 필요하며, 실제로 정렬이 이루어지는 시점은 2개의 부분 배열을 합병 (merge)하는 단계에서 이루어 짐.
힙 정렬 (heap sort)	Heap 자료구조를 사용하여 정렬. 전체 자료 중 일부만 정렬 순서로 추출하는 경우에 효율적임. 예를 들어 500명의 학생 중 상위 성적 10명만을 찾고자 할 때 가장 신속하게 처리할 수 있음.
해시	문자열과 같이 길이가 일정하지 않는 키 (key)를 사용하여 해시 값을 생성하고, 이를 배열의 인덱스나 데이터 항목이 저장되어 있는 그룹(버킷)을 찾는 데 사용
그래프 깊이 우선 탐색 (DFS)	그래프에서 지정된 시작 정점으로부터 목적지 정점까지의 경로를 깊이 우선 방식으로 탐색. 목적지까지의 탐색 시간은 상대적으로 빠를 수 있으나, 탐색된 경로가 최단 거리를 보장하지는 않음
그래프 넓이 우선탐색 (BFS)	그래프에서 지정된 시작 정점으로부터 목적지 정점까지의 경로를 넓이 우선 방식으로 탐색. 탐색된 경로는 경유하는 간선수가 최소인 것을 보장함
최단거리 경로 탐색	간선 (edge)에 가중치 (weight)이 지정된 그래프에서 시작 정점으로부터 목적지 정점까지의 경로 중에서 전체 거리가 가장 짧은 경로를 탐색. 탐색된 경로는 최단 거리인 것을 보장함
최소비용 신장트리 탐색	주어진 그래프에서 사용되는 간선들의 총 거리 (또는 가중치)가 가장 작으면서 모든 노드 (정점)들을 연결할 수 있는 최소비용 신장 트리 (minimum spanning tree)를 탐색

순차 탐색 (Sequential Search)



...



Sequential Search

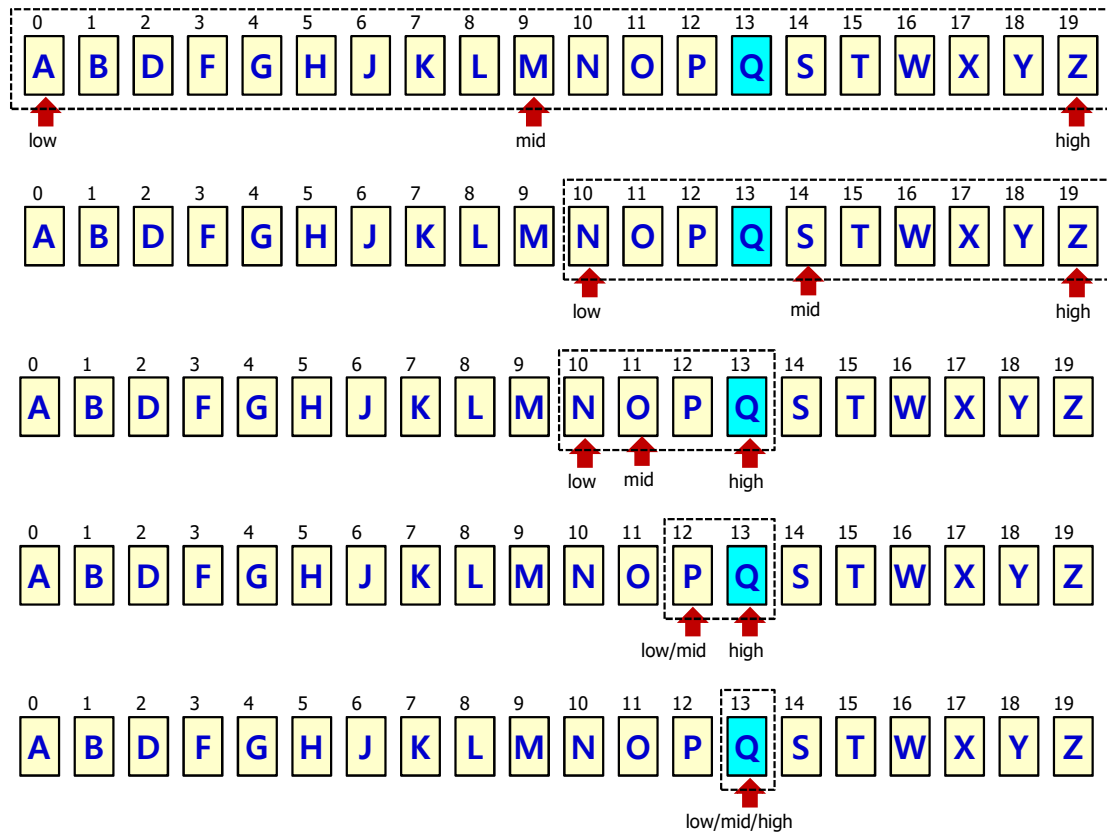
◆ 순차 탐색 (sequential search) 알고리즘

Procedure **Sequential_Search(L, N, Key)**

1. /* input argument: non-sortedList L of Length N, Key to find */
2. /* output result: the position (index) of the Key, -1 if not exists */
3. idx = 0 /* index */
4. while (idx < N) :
5. if (L[idx] == Key):
6. return idx
7. else:
8. idx = idx + 1
9. return -1 /* -1 means the given key was not found in the L */
10. end procedure /* end of Procedure Sequential_Search() */



이진 탐색 (Binary Search)



Binary Search

◆ 이진 탐색 알고리즘

- 이진 탐색 알고리즘은 주어진 리스트/배열이 정렬되어 있는 상태에서 실행

Procedure **Binary_Search(L, N, key)**

```
1.  /* input argument: SortedList L of Length N, key to find */
2.  /* output result: the position of the Key, -1 if not exists */

3.  left = 0; /* left index */
4.  right = N-1; /* right index */

5.  while (left <= right):
6.      mid = (left + right) / 2
7.      if (L[mid] > key):
8.          right = mid - 1
9.      else if (L[mid] < key):
10.         left = mid + 1
11.     else: /* L[mid] == key */
12.         return mid;
13. return -1

14. end procedure /* end of Procedure Binary_Search() */
```



순차 탐색과 이진 탐색의 비교

◆ 순차 탐색 (Sequential Search)

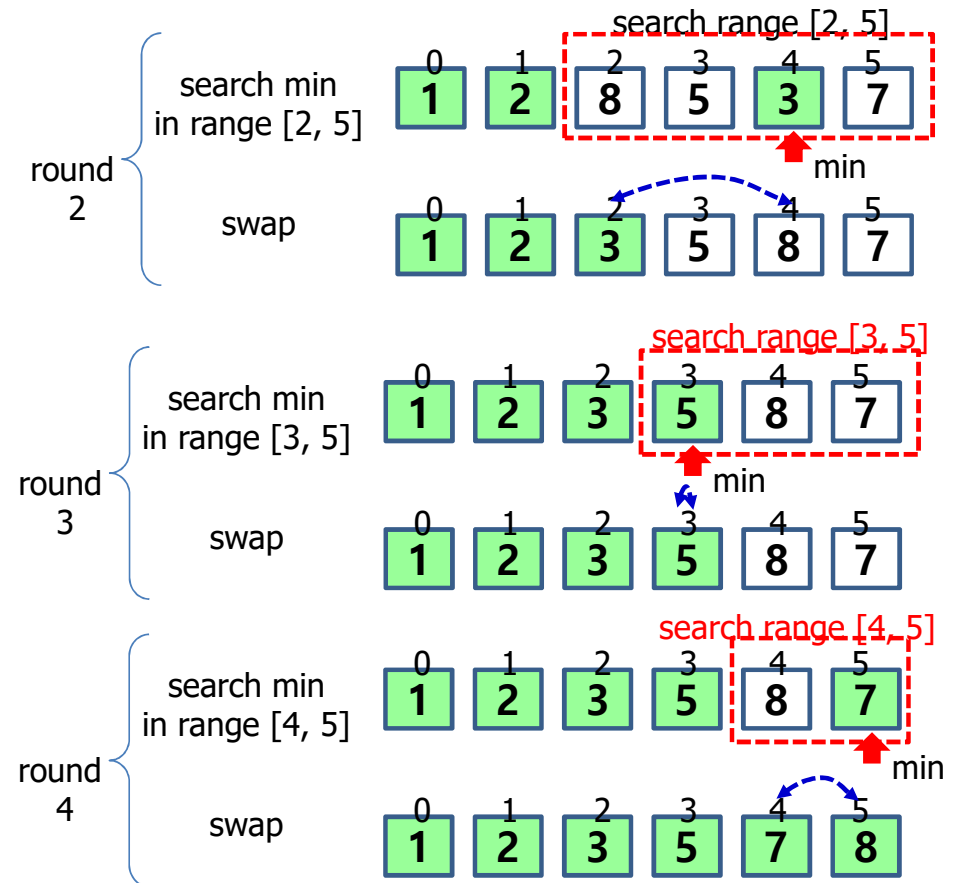
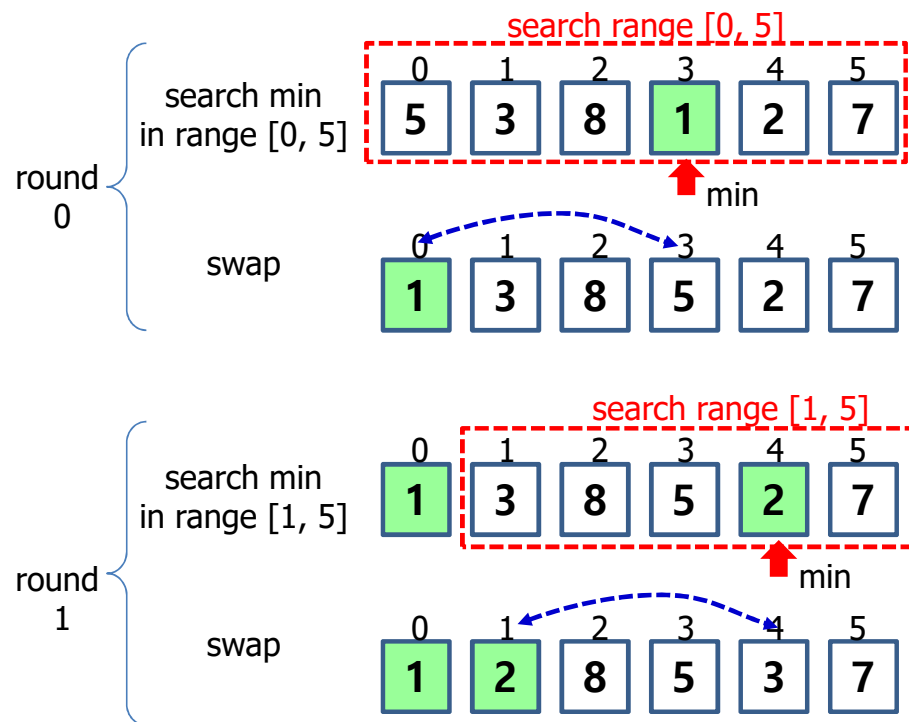
- 탐색 대상 데이터 베이스가 정렬되어 있지 않은 상태에서 맨 처음 항목부터 차례대로 비교하여 탐색 기준 값이 동일한 항목을 찾아냄
- 탐색에 걸리는 시간은 탐색 대상 자료에 포함된 데이터 베이스에 포함된 항목의 개수에 비례하며, 최대 $O(N)$ 가 될 수 있고, 평균적으로 $O(N/2)$ 정도의 시간이 걸림

◆ 이진 탐색 (Binary Search)

- 탐색 대상 데이터 베이스가 정렬되어 있는 경우에 사용되며, 탐색 구간을 절반씩 줄여가며 탐색 기준 값이 동일한 항목을 찾아냄
- 탐색에 걸리는 시간은 탐색 대상 자료에 포함된 데이터 베이스에 포함된 항목 개수에 $O(\log_2 N)$ 에 비례함
- 정렬에 걸리는 시간이 추가적으로 필요함
- 한번 정렬한 후, 많은 이진 탐색이 수행되는 경우에 효율적임

선택정렬(selection sort)

- ◆ 선택정렬(selection sort): 정렬이 안된 숫자들 중에서 최소값을 선택하여 배열의 첫 번째 요소와 교환
- ◆ 몇 개의 단계만 살펴보자.



Selection Sorting

◆ 선택 정렬 알고리즘

Procedure **Selection_Sort(L, N)**

```
1.  /* input argument: List L of Length N */
2.  /* output result: sorted List */

3.  i = 0
4.  while (i < N-1):
5.      min = L[i]
6.      min_idx = i
7.      j = i + 1
8.      while (j < N):
9.          if (min > L[j]):
10.             min = L[j]
11.             min_idx = j
12.         j = j + 1
13.     if (min_idx != i):
14.         L[min_idx] = L[i]
15.         L[i] = min
16.     i = i + 1
17. end procedure /* end of Procedure Selection_Sort() */
```



대표적인 자료구조

자료구조 란?

◆ 자료구조 (Data Structure)

- 컴퓨터 프로그램으로 문제를 해결하는 알고리즘에서 데이터/정보를 저장(store)하는 기능을 제공하며, 관련 정보를 신속하게 탐색(search)할 수 있는 기능을 제공하여야 함
- 알고리즘과 긴밀한 연관성을 가짐



정보의 정리와 검색 (Search)

◆ 사전에서의 단어 검색과 도서관에서의 도서 검색

- 사전에서 어떤 어휘 (vocabulary)의 뜻과 이 단어와 동일한 의미의 유사어 (thesaurus)를 신속하게 찾기 위하여 어떤 자료구조와 어떤 알고리즘을 사용할 수 있는가?
- 도서관에서 특정 정보를 가진 도서를 신속하게 찾기 위하여 어떤 자료구조와 어떤 알고리즘을 사용할 수 있는가 ?



(a) 사전에서의 단어 검색



(b) 도서관에서의 도서 검색

대표적인 자료구조

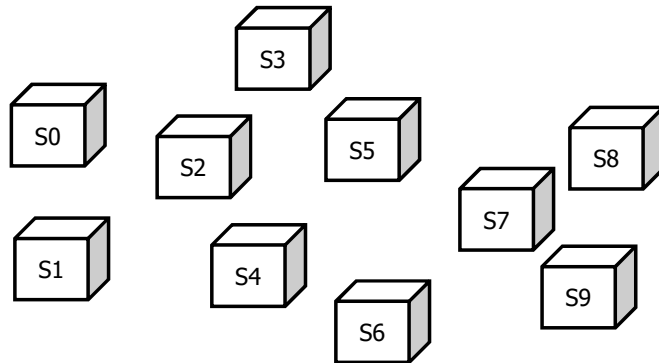
자료구조	특성
단순 배열	컴파일 단계에서 크기가 지정되어 변경되지 않는 크기의 배열
동적 배열	프로그램 실행 단계에서 크기가 지정되며, 프로그램 실행 중에 크기를 변경할 수 있는 배열
구조체 배열	구조체로 배열 원소가 지정되는 배열
연결형 리스트	확장성이 있으며, 단일연결형 또는 이중 연결형으로 구성 가능. 다양한 컨테이너 자료구조의 내부적인 자료구조로도 활용됨
스택 (stack)	Last In First Out (LIFO) 특성을 가짐. 배열 또는 연결형 리스트로 구현할 수 있음.
큐 (queue)	First In First Out (FIFO) 특성을 가짐. 배열 또는 연결형 리스트로 구현할 수 있음.
힙 우선 순위 큐 (heap priority queue)	컨테이너 내부에 가장 우선순위가 높은 데이터 항목을 추출할 수 있도록 관리하며, 배열 또는 자기 참조 구조체로 구현할 수 있음
이진 탐색 트리	컨테이너 내부에 포함된 데이터 항목들을 정렬된 상태로 관리하여야 할 때 매우 효율적임. 단순 이진 탐색 트리의 경우 편중될 수 있으며, 편중된 경우 검색 성능이 저하되기 때문에, 밸런싱이 필요함.
해시 테이블 (Hash Table)	컨테이너 자료구조에 포함된 항목들을 문자열 (string) 또는 긴 숫자를 키 (key)로 사용하여 관리하여야 하는 경우, key로부터 해시 값을 구하고, 이 해시 값을 배열의 인덱스로 사용함.
Map	key와 항목 간에 1:1 관계가 유지되어야 하는 경우에 사용되며, 해시 테이블을 기반으로 구현할 수 있음
Dictionary	key와 항목 간에 1:N 관계가 유지되어야 하는 경우에 사용되며, 해시 테이블을 기반으로 구현할 수 있음
그래프	정점 (vertex)/노드 (node)로 개체 (object)가 표현되고, 간선 (edge)/링크(link)들을 사용하여 개체 간의 관계를 표현하는 경우에 적합함. 그래프를 기반으로 경로 탐색, 최단 거리 경로 탐색, 신장 트리 (spanning tree) 탐색 등에 활용됨.



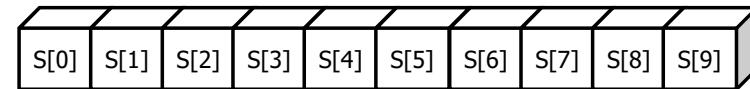
배열 (Array)

◆ 배열(array)

- 동일한 자료형의 데이터가 여러 개 저장되어 있는 데이터 저장 장소
- 배열 안에 들어있는 각각의 데이터들은 정수로 되어 있는 인덱스(첨자)에 의하여 접근
- 배열을 이용하면 동일한 자료형의 여러 개의 데이터 들을 하나의 이름으로 관리할 수 있다.



(a) 각 원소들이 개별적인 이름을 사용하는 경우

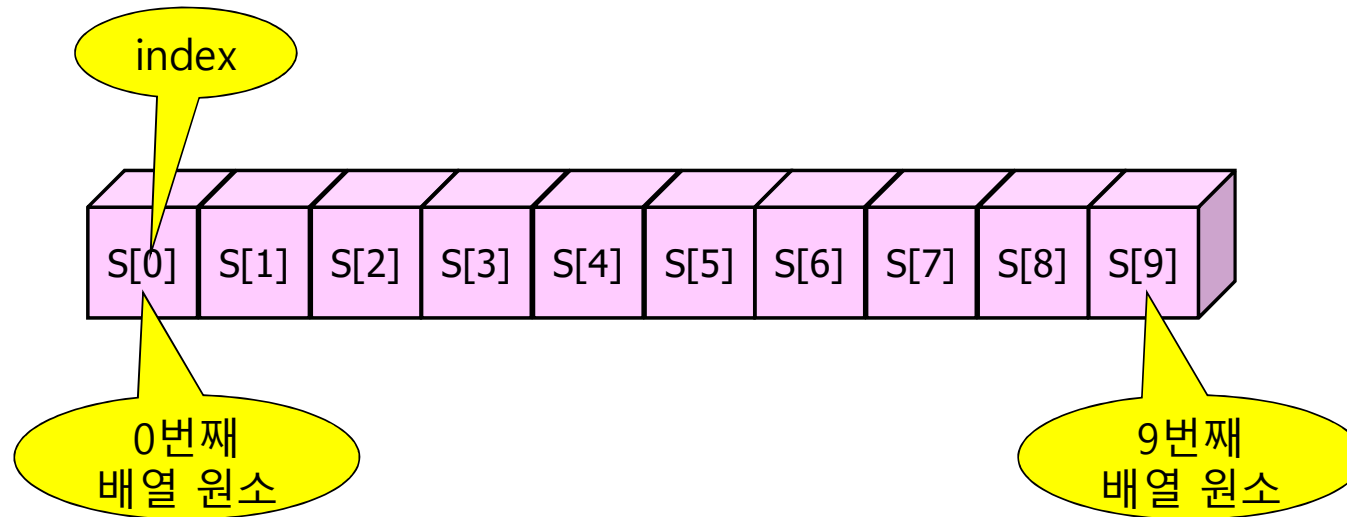


(b) 하나의 이름을 사용하며, 연속적인 메모리 공간에 저장되어 인덱스로 각 원소를 개별적으로 사용하는 배열의 경우

배열 원소와 인덱스

◆ 인덱스(index)

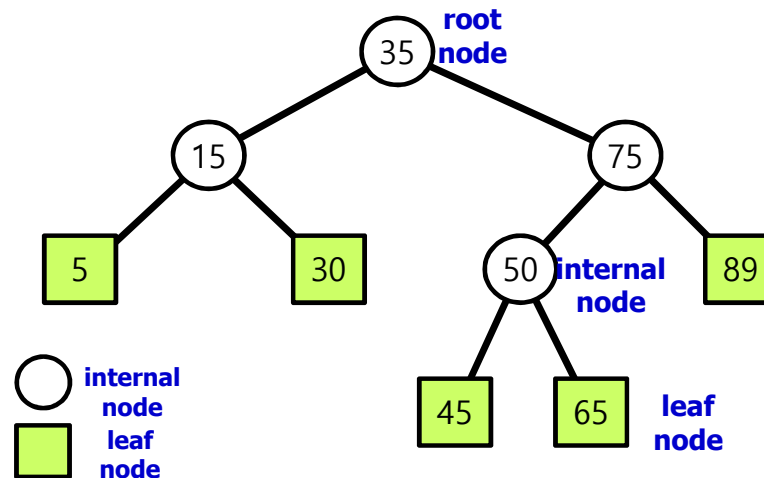
- 배열은 하나의 이름을 사용하며, 연속적인 메모리 공간에 저장되어 있는 배열 원소를 인덱스 (첨자, 색인 번호)로 개별적인 접근이 가능
- 총 N개의 배열 원소가 포함되어 있을 경우 인덱스는 $0 \sim N-1$



이진 탐색 트리 (Binary Search Tree)

◆ 이진 탐색 트리 (Binary search tree):

- 데이터 탐색을 쉽게 수행할 수 있도록 구성된 이진 트리
- 각 노드의 왼쪽 서브 트리는 그 노드의 값보다 작은 데이터를 가지는 노드들로만 구성
- 각 노드의 오른쪽 서브 트리는 그 노드의 데이터 값보다 같거나 큰 데이터 값을 가지는 노드들로만 구성



파이썬 프로그램의 기본 구성

MyFirstPythonProgram.py

```
# MyFirstPythonProgram.py (1)
"""
Project : My first Python Program
Author: Cul-Soo Kim
Date of last update: March 5, 2022
Update list:
    - v1.0 : March 2, 2022
        . My first Python program with simple
          prints of variables
    - v1.1 : March 5, 2022
        . Include str and mathematical operations
"""

print("This is my first Python program !")
my_name = "Chul-Soo Kim" # string variable
print("My name is", my_name, ".")
print("I am really happy to learn\
Python Programming !!")

# definitions of variables
x = 5 # integer variable
y = 2 # integer variable
print("x : ", x)
print("y : ", y)
```

```
# MyFirstPythonProgram.py (2)

# operations
sum_xy = x + y
sub_xy = x - y
mul_xy = x * y
div_xy = x / y
int_div_xy = x // y
print("x + y : ", sum_xy)
print("x - y : ", sub_xy)
print("x * y : ", mul_xy)
print("x / y : ", div_xy)
print("x // y : ", int_div_xy)
```

```
This is my first Python program !
My name is Chul-Soo Kim .
I am really happy to learn Python Programming !!
x : 5
y : 2
x + y : 7
x - y : 3
x * y : 10
x / y : 2.5
x // y : 2
```



기본 주석문 (basic comments)

◆ 주석문 (Comments)

- 프로그램의 목적 및 주요 기능
- 프로그램 작성자 및 작성 일자, 보완 일자
- 프로그램의 주요 수정/보완 내용

파이썬 프로그램 주석문 형식	설 명
<pre>""" comment_line_1 comment_line_2 """</pre>	여러 줄에 걸친 주석문
<pre># comment to the end of line</pre>	'#' 문자로부터 그 줄 끝까지 주석문

```
"""
Project : My first Python Program
Author: Cul-Soo Kim
Date of last update: Jan. 5, 2021
Update list:
- v1.0 : Jan. 2, 2021
    . My first Python program with simple
      prints of variables
- v1.1 : Jan. 5, 2021
    . Include str and mathematical operations
"""

# First Greetings
x = 5  # variable
```



데이터 입력 기능을 가지는 파이썬 프로그램 예제

◆ 데이터 입력 및 연산 예제

```
"""
    Basic Comments (same as before)
"""

# definitions of variables
x_str = input("input x = ") # variable
y_str = input("input y = ") # variable
x, y = int(x_str), int(y_str)
print("x, y = {}, {}".format(x, y))

# operations
sum_xy = x + y
sub_xy = x - y
mul_xy = x * y
div_xy = x / y
int_div_xy = x // y
print("x + y : ", sum_xy)
print("x - y : ", sub_xy)
print("x * y : ", mul_xy)
print("x / y : ", div_xy)
print("x // y : ", int_div_xy)
```

```
input x = 7
input y = 3
x, y = 7, 3
x + y : 10
x - y : 4
x * y : 21
x / y : 2.3333333333333335
x // y : 2
```



변수 (variable)의 선언 및 사용

◆ 변수 (variable) 선언 및 사용 예

```
# Python program variables
```

```
x = 1 # integer
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = 2.345 #float
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = [1, 2, 3, 4] # list
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = (7, 8, 9, 10) # tuple
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = {'A':1, 'B':2, 'C':3} # dict
print("x = ", x)
print("type(x) = ", type(x))
```

```
x = {1, 2, 3, 4} # set
print("x = ", x)
print("type(x) = ", type(x))
```

```
--
x = 1
type(x) = <class 'int'>
x = 2.345
type(x) = <class 'float'>
x = [1, 2, 3, 4]
type(x) = <class 'list'>
x = (7, 8, 9, 10)
type(x) = <class 'tuple'>
x = {'A': 1, 'B': 2, 'C': 3}
type(x) = <class 'dict'>
x = {1, 2, 3, 4}
type(x) = <class 'set'>
```



상수 (constant)의 선언 및 사용

◆ 상수 (constant)의 선언 및 사용 예

```
# variables and constants in Python program

PI = 3.141592

radius = 5.0
circle_area = radius * radius * PI
print("Area of circle (radius : {}) : {}".format(radius, circle_area))

width = 5.0
length = 4.0
rectangle_area = width * length
print("Area of rectangle (width: {}, length: {}) : {}".format(width, length, rectangle_area))

base = 5.0
height = 4.0
triangle_area = (base * height) / 2.0
print("Area of triangle (base: {}, height: {}) : {}".format(base, height, triangle_area))

Area of circle (radius : 5.0) : 78.5398
Area of rectangle (width: 5.0, length: 4.0) : 20.0
Area of triangle (base: 5.0, height: 4.0) : 10.0
```



과학 기술 계산에서 많이 사용되는 상수들

◆ 기호상수 (symbolic constant) 정의

기호 상수의 예	설 명
$\text{PI} = 3.141592653589793238$	원주율의 값을 의미하며, 원면적, 원둘레, 삼각함수 등에서 사용
$\text{INT_MAX} = 2147483647$	32비트로 표현되며 부호가 포함된 정수 중 가장 큰 값
$\text{INT_MIN} = -2147483638$	32비트로 표현되며 부호가 포함된 정수 중 가장 작은 값
$\text{RAND_MAX} = 32767$	16비트값으로 생성되는 난수 (random number)의 최대 값

**파이썬 프로그램의 식별자 (Identifier),
기본 수학연산, 함수호출**

파이썬 프로그램 식별자

◆ 식별자 (identifier)

- 변수 (variable), 상수 (constant), 함수 (function) 이름
- 의미를 전달할 수 있는 영어 단어 사용

◆ 식별자의 예

구분		식별자 예
기 하, 도 형	원	radius (반지름), diameter (지름), circumference(원둘레), height (원기둥 높이)
	삼각형	base(밑 변), side (대 각 선), vertex (꼭 지 점), height (삼 각 형 높 이), prism (삼 각 기 둥), prism_height(삼각기둥 높이)
	사각형	width(가로), length(세로), height(사각기둥 높이)
도형 그리기		draw(그리기), rotate(회전), move(이동)
산술 계산		add (덧셈), subtract(뺄셈), multiply(곱셈), divide(나눗셈), modulo(모듈로)

식별자 (Identifier) 설정 원칙

◆ 식별자 설정 규정 (Rule of identifier)

- 영문자(대문자, 소문자), 밑줄 문자, 숫자로 구성
 - 대문자 'A' ~ 'Z', 소문자 'a' ~ 'z', 밑줄 문자 ('_'), 숫자 ('0' ~ '9')
- 숫자는 식별자의 첫 문자가 될 수 없음 ('0' ~ '9')
- 파이썬의 키워드 (keyword)는 식별자가 될 수 없음
- 식별자의 길이에는 제한이 없음
- 대문자와 소문자를 구분함

```
>>> import keyword
>>> keyword.kwlist
['False', 'None', 'True', 'and', 'as', 'assert', 'break', 'class', 'continue', '
def', 'del', 'elif', 'else', 'except', 'finally', 'for', 'from', 'global', 'if',
'import', 'in', 'is', 'lambda', 'nonlocal', 'not', 'or', 'pass', 'raise', 'retu
rn', 'try', 'while', 'with', 'yield']
>>> |
```


식별자 설정 방법

◆ 식별자 설정 방법

- Camel casing: 단어의 첫 문자를 대문자로 구분
- GNU 작명 관례: 단어 사이에 밑줄 문자 사용

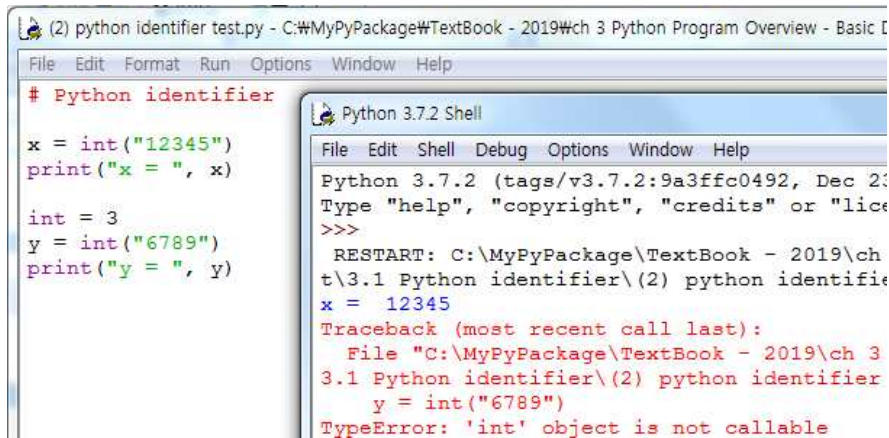
식별자 이름 작명 방법	설 명
camel casing	단어들을 연속적으로 나열하며, 각 단어의 첫 문자를 대문자로 표시 (단 첫 번째 단어는 소문자로 사용가능) 예) printComplex(), drawRectangle(), drawPolygon()
GNU 작명 관례	단어들을 연속적으로 나열하며, 각 단어사이에 밑줄 문자를 삽입 예) print_complex(), draw_rectangle(), draw_polygon()

식별자의 예

◆ 파이썬 식별자의 예

파이썬 식별자 예	사용 가능 여부	설명
average, avg, avg1	사용 가능	문자와 숫자
num_data, numData,	사용 가능	문자, 밑줄
ch, ch1, ch_1	사용 가능	문자, 밑줄
i, d, f	사용 가능	
_avg, __area	사용 가능	밑줄, 문자
NUM_DATA	사용 가능	대문자, 밑줄
auto, char, int, class, while, for	사용 불가능	keyword
3avg	사용 불가능	숫자로 시작
avg!, avg%, &avg, dollar\$	사용 불가능	특수문자 포함

잘못 설정된 파이썬 식별자의 예



```
(2) python identifier test.py - C:\MyPyPackage\TextBook - 2019\ch 3 Python Program Overview - Basic I
File Edit Format Run Options Window Help

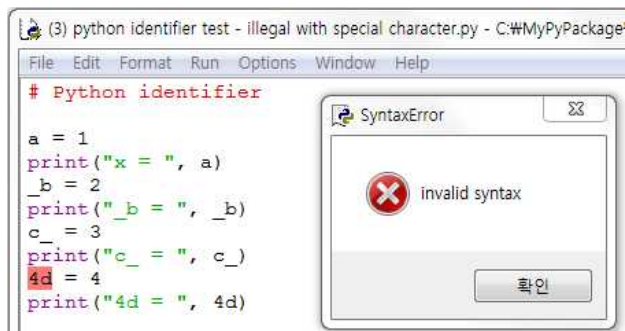
# Python identifier

x = int("12345")
print("x = ", x)

int = 3
y = int("6789")
print("y = ", y)
```

Python 3.7.2 Shell

```
Python 3.7.2 (tags/v3.7.2:9a3ffc0492, Dec 23 2019)
Type "help", "copyright", "credits" or "license()" for more
>>>
RESTART: C:\MyPyPackage\TextBook - 2019\ch 3 Python Program Overview - Basic I
x = 12345
Traceback (most recent call last):
  File "C:\MyPyPackage\TextBook - 2019\ch 3 Python Program Overview - Basic I
  File "C:\MyPyPackage\TextBook - 2019\ch 3 Python Program Overview - Basic I
    y = int("6789")
TypeError: 'int' object is not callable
```



```
(3) python identifier test - illegal with special character.py - C:\MyPyPackage\
File Edit Format Run Options Window Help

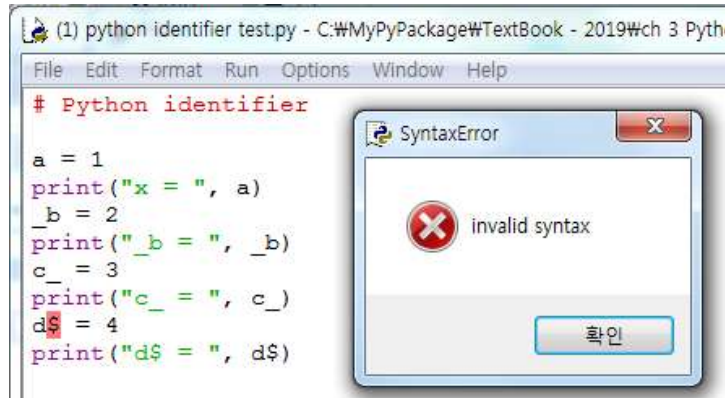
# Python identifier

a = 1
print("x = ", a)
_b = 2
print("_b = ", _b)
c_ = 3
print("c_ = ", c_)
4d = 4
print("4d = ", 4d)
```

SyntaxError

invalid syntax

확인



```
(1) python identifier test.py - C:\MyPyPackage\TextBook - 2019\ch 3 Python Program Overview - Basic I
File Edit Format Run Options Window Help

# Python identifier

a = 1
print("x = ", a)
_b = 2
print("_b = ", _b)
c_ = 3
print("c_ = ", c_)
d$ = 4
print("d$ = ", d$)
```

SyntaxError

invalid syntax

확인

기본 수학 연산

◆ 기본 수학 연산

파이썬 기본 연산자	예제 및 설명
+	주어진 데이터가 숫자 인 경우: 덧셈 주어진 데이터가 문자열일 경우: 문자열 병합
-	주어진 숫자 데이터에 대한 뺄셈
*	주어진 데이터가 숫자 인 경우: 곱셈 주어진 데이터가 문자열일 경우: 문자열 병합
/	주어진 숫자 데이터에 대한 실수형 나눗셈 (소수점 이하 값 유지)
//	주어진 숫자 데이터에 대한 정수형 나눗셈 (소수점 이하 값 생략)
%	주어진 숫자 데이터에 대한 모듈로 계산

함수의 호출 및 실행

◆ 파이썬 함수 호출의 예

파이썬 기본 함수 예시	예제 및 설명
<code>x_str = input("prompt string");</code>	표준 입력 장치로부터 문자열 입력 prompt 문자열이 지정되어 있으면 이를 먼저 출력하고, 문자열 입력
<code>x = int(x_str)</code>	주어진 숫자 문자열을 정수 (integer)로 변환
<code>y = float(y_str)</code>	주어진 숫자 문자열을 실수 (float)로 변환
<code>print("output string", data);</code>	<code>print("sum = ", sum)</code>

파이썬 프로그램의 입력과 출력

파이썬 프로그램의 데이터 입출력

◆ 파이썬 프로그램의 입력 및 출력 예

```
# Python input and output
```

```
a = input("input a data : ")  
print("input data a = ", a)  
print("type of a is ", type(a))
```

```
b = input("input an integer data : ")  
print("input data b = ", b)  
print("type of b is ", type(b))  
sum = a + b  
print("sum is : ", sum)
```

```
input a data : 10  
input data a = 10  
type of a is <class 'str'>  
input an integer data : 20  
input data b = 20  
type of b is <class 'str'>  
sum is : 1020
```



입력 문자열의 숫자 데이터 변환

◆ 숫자 데이터 입력

```
# Simple Python program with number input and output
```

```
a_str = input("input a_str : ")
print("input a_str = ", a_str)
print("type of a_str is ", type(a_str))
a = int(a_str) # a = int(input("input a = "))
print("type of a is ", type(a))
```

```
b_str = input("input b_str : ")
print("input b_str = ", b_str)
print("type of b_str is ", type(b_str))
b = int(b_str) # b = int(input("input b = "))
sum = a + b
print("sum is : ", sum)
```

```
input a_str : 10
input a_str = 10
type of a_str is <class 'str'>
type of a is <class 'int'>
input b_str : 20
input b_str = 20
type of b_str is <class 'str'>
sum is : 30
```



포맷 지정 데이터 출력

◆ 파이썬 출력 포맷 지정 방법 (1) - %

문자열 서식 지정	서식 지정 예
%d, %s, %f의 형식 지정자를 사용한 출력 형식 지정	<pre>print("%d %5d %10s, %7.3f" % (123, 456, "Python", 123.456))</pre> <p>%d : 10진수를 출력 %5d : 5칸을 확보하여 오른쪽으로 정렬된 10진수로 출력 %05d : 5칸을 확보하여 오른쪽으로 정렬된 10진수로 출력하며, 왼쪽에 남은 공간에 0을 출력 %c : 문자 출력 %s : 문자열 (string) %10s : 10칸을 확보하여 오른쪽으로 정렬된 문자열을 출력 %-10s : 10칸을 확보하여 왼쪽으로 정렬된 문자열을 출력 %f : 실수 출력 %7.3f : 7칸을 확보하여 오른쪽으로 정렬된 실수를 출력하며, 소수점 아래 3자리까지 출력하며, 오른쪽 빈칸은 0으로 채움</p>
format() 함수를 사용한 서식 지정	format(123456789, 'd') : 1000단위로 comma (,)를 삽입한 형식으로 출력

포맷 지정 데이터 출력

◆ 파이썬 출력 포맷 지정 방법 (2) - .format()

문자열 서식 지정	서식 지정 예
{n:x}와 .format 메소드를 사용한 서식 지정	<pre>print("{0:d} {1:5d} {2:05d} {3:10s} {4:7.3f}".format(123, 456, 678, "Python", 123.456))</pre> <p>{n:d} : n번째 항을 10진수 형식으로 출력</p> <p>{n:5d} : n번째 항을 5칸을 확보하여 오른쪽으로 정렬된 10진수로 출력</p> <p>{n:05d} : n번째 항을 5칸을 확보하여 오른쪽으로 정렬된 10진수로 출력 (왼쪽 남은 공간 0으로 채움)</p> <p>{n:#010b} : n번째 항을 10칸의 공간에 접두어 (0b)를 포함하여 이진수로 출력 (왼쪽 남은 공간 0으로 채움)</p> <p>{n:#05o} : n번째 항을 5칸의 공간에 접두어 (0o)를 포함하여 이진수로 출력 (왼쪽 남은 공간 0으로 채움)</p> <p>{n:#04X} : n번째 항을 10칸의 공간에 접두어 (0X)를 포함하여 이진수로 출력 (왼쪽 남은 공간 0으로 채움)</p> <p>{n:c} : n번째 항을 문자 자료형으로 출력</p> <p>{n:s} : n번째 항을 문자열 (string) 자료형으로 출력</p> <p>{n:10s} : 10칸을 확보하여 오른쪽으로 정렬된 문자열을 출력</p> <p>{n:-10s} : 10칸을 확보하여 왼쪽으로 정렬된 문자열을 출력</p> <p>{n:f} : 실수 (float) 자료형으로 출력</p> <p>{n:7.3f} : 7칸을 확보하여 오른쪽으로 정렬된 실수를 출력하며, 소수점 아래 3자리까지 출력하며, 오른쪽 빈칸은 0으로 채움</p> <p>{n:>} n번째 항을 오른쪽 정렬</p> <p>{n:^} n번째 항을 중앙정렬</p> <p>{n:<} n번째 항을 왼쪽 정렬</p>

출력 포맷 지정 예제

◆ 출력 포맷 지정 예제

```
# Python print() with formatting

print("%5d %7d %07d %10s %7.3f" % (123, 456, 678, "Python", 123.456789))
print("%5d %7d %07d %-10s %10.6f" % (12, 34, 5, "Python", 123.456789))

print("{0:5d} {1:7d} {2:07d} {3:10s} {4:7.3f}" \
      .format(123, 456, 678, "Python", 123.456789))
print("{0:5d} {1:7d} {2:07d} {3:<10s} {4:10.6f}" \
      .format(12, 34, 5, "Python", 123.456789))
print("{0:5d} {1:7d} {2:07d} {3:^10s} {4:10.6f}" \
      .format(12, 34, 5, "Python", 123.456789))
print("{0:5d} {1:7d} {2:07d} {3:>10s} {4:10.6f}" \
      .format(12, 34, 5, "Python", 123.456789))
print("Long integer with comma at each 1000 unit : " \
      , format(123456789, ',d'))
```

```
123      456 0000678      Python 123.457
12       34 0000005 Python   123.456789
123      456 0000678 Python   123.457
12       34 0000005 Python   123.456789
12       34 0000005 Python   123.456789
12       34 0000005 Python   123.456789
12       34 0000005 Python   123.456789
Long integer with comma at each 1000 unit : 123,456,789
```



파이썬 프로그램 실행 제어 기초

- 조건문과 반복문 개요

조건문 개요 – if, if-else

◆ 조건문 if

```
# simple program to input two integers and compare
x = int(input("input first integer (x) : "))
y = int(input("input second integer (y) : "))
if x == y:
    print("x(%d) is equal to y(%d)"%(x, y))
if x < y:
    print("x(%d) is less than y(%d)"%(x, y))
if x > y:
    print("x(%d) is greater than y(%d)"%(x, y))
```

```
input first integer (x) : 7
input second integer (y) : 5
x(7) is greater than y(5)
```

```
input first integer (x) : 3
input second integer (y) : 3
x(3) is equal to y(3)
```

```
input first integer (x) : 4
input second integer (y) : 8
x(4) is less than y(8)
```



조건문 – if-elif-else

◆ 조건문 if-elif-else

```
# conditional branch with if - elif - else
score = int(input('course score [0..99] = '))
if 90 <= score <= 100:
    grade = 'A'
elif 80 <= score:
    grade = 'B'
elif 70 <= score:
    grade = 'C'
elif 60 <= score:
    grade = 'D'
else:
    grade = 'F'
print("score = %d, grade = %s" %(score, grade))
```

```
course score [0..99] = 95
score = 95, grade = A

course score [0..99] = 74
score = 74, grade = C
```



while 반복문 개요

◆ while-loop 기본 구조

```
# while-loop for input positive integers

L = list() # creation of an empty list L
print("Input positive integers (-1 to end)")
x = int(input("data : "))
n = 0
sum = 0
while x >= 0:
    L.append(x) # append x to list L
    n += 1
    sum = sum + x
    x = int(input("data : "))

print("Total", n, " integers were input: ", L)
print("Sum is : ", sum)
```

```
Input positive integers (-1 to end)
data : 1
data : 3
data : 5
data : 7
data : 9
data : 2
data : 4
data : 6
data : 8
data : 10
data : -1
Total 10 integers were input: [1, 3, 5, 7, 9, 2, 4, 6, 8, 10]
Sum is : 55
```



while-loop과 continue, break

◆ while-loop with break and continue

```
# while-loop, break, continue

MAX_NUM_DATA = 100
num_data = 0
data_sum = 0
print("Input (up to {} ) integer data.".format(MAX_NUM_DATA))
while num_data < MAX_NUM_DATA:
    data = int(input("Data (-1 to finish) = "))
    num_data = num_data + 1
    if data == -1:
        break
    elif data <= 0:
        continue
    else:
        data_sum = data_sum + data

print("Total {} data input, sum of positive data = {}".format(num_data, data_sum))
```

```
Input (up to 100 ) integer data.
Data (-1 to finish) = 10
Data (-1 to finish) = 2
Data (-1 to finish) = 7
Data (-1 to finish) = 1
Data (-1 to finish) = -5
Data (-1 to finish) = -9
Data (-1 to finish) = 2
Data (-1 to finish) = 3
Data (-1 to finish) = 13
Data (-1 to finish) = -1
Total 10 data input, sum of positive data = 38
```



입력 데이터 중의 최댓값, 최솟값 찾기

◆ Find min, max of input data list

```
# while-loop, find min and max of input data list

TARGET_NUM_DATA = 10
num_data = 0
L_data = [] # empty list
L_sum = 0
print("Input up to {} integer data."\
      .format(TARGET_NUM_DATA))
while num_data < TARGET_NUM_DATA:
    data = int(input("data = "))
    num_data = num_data + 1
    L_data.append(data)
    L_sum = L_sum + data
    if num_data == 1:
        data_min = data_max = data
        continue
    if data < data_min:
        data_min = data
    if data > data_max:
        data_max = data

print("Input data list = ", L_data)
print("Min = {}, Max = {}, Avg = {}".\
      .format(data_min, data_max, L_sum/num_data))
```

```
Input up to 10 integer data.
data = 32
data = 45
data = 1
data = 5
data = 10
data = 4
data = 3
data = 99
data = -5
data = -30
Input data list = [32, 45, 1, 5, 10, 4, 3, 99, -5, -30]
Min = -30, Max = 99, Avg = 16.4
```



range() Generator 함수

◆ range(start, stop, step)

- start로 부터 stop 직전까지 (stop은 포함하지 않음) step씩 증가하며 정수 값을 생성하여 제공
- 만약 인수가 2개만 제공되면, start와 stop의 값으로 사용되며, step은 1로 설정됨
- 만약 인수가 1개만 제공되면 stop의 값으로 사용되며, start는 0으로, step은 1로 설정됨
- 만약 step 값이 음수이면, start로 부터 stop 직전까지, step에 따라 감소시키며 정수값을 생성하여 제공

range() 사용 예	설 명
range(10)	0부터 9까지 1씩 증가하며 정수를 생성
range(1, 10)	1부터 9까지 1씩 증가하며 정수를 생성
range(1, 10, 2)	1부터 9까지 2씩 증가하며 정수를 생성
range(10, 100, 5)	10부터 99까지 5씩 증가하며 정수를 생성
range(100, 0, -1)	100부터 1까지 1씩 감소하며 정수를 생성

range()와 in을 사용하는 for 반복문

◆ for-loop

- for-loop에서는 in과 range() 를 사용하여 구성할 수 있음
- for-loop에서 사용하는 변수 (아래 예에서는 i)의 값이 range()에서 생성된 값에 차례로 대입되어 사용되며, 전체 반복 회수는 range()에서 생성된 숫자의 개수에 의해 제한됨

```
# for-loop with range()
n = int(input('Input n to calculate sum of [0..n] : '))
nSum = 0
for i in range(0, n+1): #nSum = sum(range(0, n+1))
    nSum = nSum + i
print("Sum of [0..%d] = %d" %(n, nSum))
```

```
Input n to calculate sum of [0..n] : 10
Sum of [0..10] = 55
```

```
Input n to calculate sum of [0..n] : 100
Sum of [0..100] = 5050
```



for 반복문과 continue, break

◆ Example of for-loop with break and continue

```
#for_loop with break and continue

n = int(input("Input number of data to process: "))
L = list()
sum = 0
print("Input %d non-negative integers"%(n))
for i in range(n):
    d = int(input())
    if d == 0:
        continue
    elif d < 0:
        break
    L.append(d)
    sum = sum + d
print("Input data : ", L)
print("Sum = ", sum)
```

```
===== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
3
0
4
5
Input data : [2, 3, 4, 5]
Sum = 14
>>>
===== RESTART: C:/YTK-PythonProg/3_
Input number of data to process: 5
Input 5 non-negative integers
2
-1
Input data : [2]
Sum = 2
>>> |
```



입력데이터의 통계 분석 – 평균, 분산, 표준편차

```
# while-loop, for-loop, find min and max of input data list
import math
```

```
TARGET_NUM_DATA = 10
num_data = 0
L_data = [] # empty list
L_sum = 0
print("Input {} integer data."\
      .format(TARGET_NUM_DATA))
while num_data < TARGET_NUM_DATA:
    data = int(input("data = "))
    num_data = num_data + 1
    L_data.append(data)
    L_sum = L_sum + data

avg = L_sum / num_data
diff_sq_sum = 0
for i in range(num_data):
    diff = avg - L_data[i] # difference from avg
    diff_sq_sum = diff_sq_sum + diff * diff
var = diff_sq_sum / num_data
std = math.sqrt(var)

print("Input data list = ", L_data)
print("avg = {}, var = {}, std = {}".format(avg, var, std))
```

```
Input 10 integer data.
```

```
data = 1
data = 2
data = 3
data = 4
data = 5
data = 6
data = 7
data = 8
data = 9
data = 10
```

```
Input data list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
avg = 5.5, var = 8.25, std = 2.8722813232690143
```

```
Input 10 integer data.
```

```
data = -5
data = -4
data = -3
data = -2
data = -1
data = 0
data = 1
data = 2
data = 3
data = 4
```

```
Input data list = [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4]
avg = -0.5, var = 8.25, std = 2.8722813232690143
```



ASCII Characters

◆ ASCII (American Standard Code for Information Interchange)

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0x10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0x20	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0x30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x50	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
0x60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x70	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

Printing Upper-case/Lower-case Characters and Numbers in ASCII table

```
# Printing Upper_case / Lower_case Characters, and Digits of ASCII Code Table
```

```
L_upper = [] # list
for x in range(0x41, 0x5B):
    L_upper.append(chr(x))
print("Upper case alphabets : \n", L_upper)
print()
```

```
L_lower = []
for x in range(0x61, 0x7B):
    L_lower.append(chr(x))
print("Lower case alphabets : \n", L_lower)
print()
```

```
L_digits = []
for x in range(0x30, 0x3A):
    L_digits.append(chr(x))
print("Digits : \n", L_digits)
print()
```

	0x00	0x01	0x02	0x03	0x04	0x05	0x06	0x07	0x08	0x09	0x0A	0x0B	0x0C	0x0D	0x0E	0x0F
0x00	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
0x10	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
0x20	SP	!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0x30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0x40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0x50	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
0x60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0x70	p	q	r	s	t	u	v	w	x	y	z	{		}	~	DEL

```
Upper case alphabets :
['A', 'B', 'C', 'D', 'E', 'F', 'G', 'H', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X', 'Y', 'Z']

Lower case alphabets :
['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z']

Digits :
['0', '1', '2', '3', '4', '5', '6', '7', '8', '9']
```

객체 지향형 프로그래밍 개요

객체 지향형 프로그래밍 (Object-Oriented Programming)

◆ 객체 (Object)

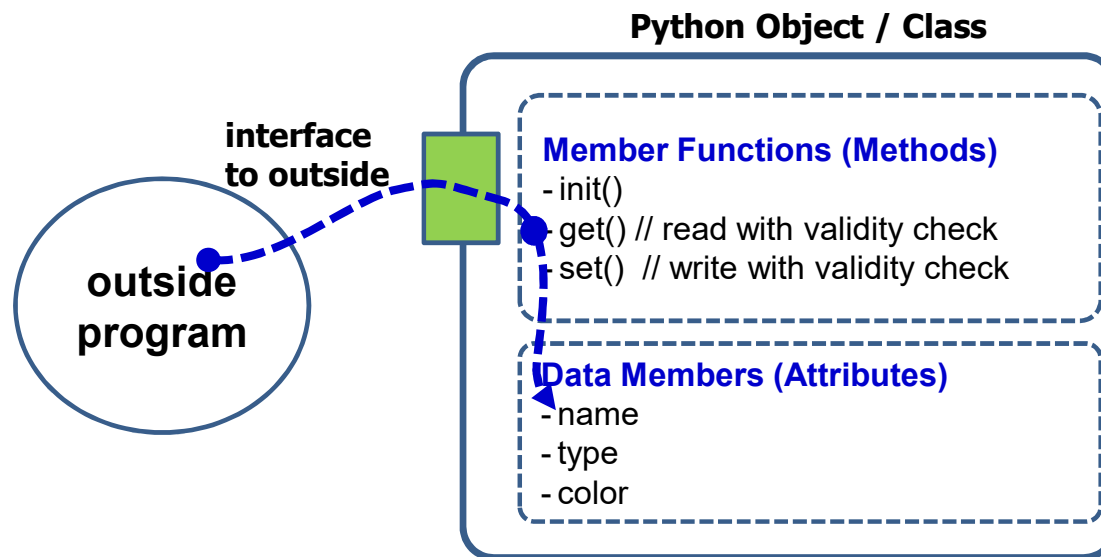
- 우리 주변에서 살펴볼 수 있는 다양한 사물과 생물들을 의미하며 크기, 색깔, 위치 등의 상태 (state) 정보와 스스로 또는 다른 객체에 의하여 이루어지는 행동 (behavior)을 가짐.
- 예를 들어 사람에게는 이름, 생년월일, 키, 몸무게, 현재 위치, 소속 기관 등의 상태 정보와 그 사람이 다른 곳으로 이동하거나, 키와 몸무게가 변하거나, 노래를 부르거나, 소속기관이 변경되는 등의 행동이 이루어질 수 있음.

◆ 객체 지향형 프로그래밍

- 소프트웨어 시스템을 객체 (object) 단위로 나누고, 각 객체는 그 내부에 데이터와 관련 함수를 함께 포함하게 함으로써 독립성을 유지하게 하는 프로그래밍 기법
- 객체 지향형 프로그래밍에서는 문제를 해결하는 함수 호출의 절차적 실행 순서보다는 각 객체가 어떻게 상호 연관성을 가지며, 각 객체 내부에서 데이터를 어떻게 정상적인 범위 내에 유지하게 하고, 그 데이터를 어떤 자료구조를 사용하여 가장 효율적으로 표현하며, 어떤 알고리즘을 사용하여 가장 효율적으로 처리할 수 있는가에 대하여 중점을 두게 됨.

파이썬 클래스

◆ 클래스 (class)



클래스와 인스턴스

◆ 클래스 (class)

- 객체를 소프트웨어에서 표현하고 구현하기 위하여 사용하는 틀/모델
- 클래스는 객체 지향형 프로그래밍으로 알고리즘 및 자료구조를 구현하기 위한 기본 단위

◆ 인스턴스 (instance)

- 클래스를 사용하여 실체를 만든 것
- 자료형 (예: int, float)을 사용하여 변수 (예: x, y)를 만드는 것과 같이 클래스 (예: list)를 사용하여 다수의 인스턴스 (예: lst_name, lst_value)를 생성할 수 있음

파이썬 클래스

◆ 클래스의 기본 멤버 함수

클래스의 기본 멤버 함수	설 명
생성자 (constructor)	클래스를 사용하여 객체가 생성될 때 실행되며, 초기화 기능 수행
데이터 멤버 접근자 (accessor)	클래스의 데이터 멤버 값을 읽기 위하여 접근하는 기능 제공. 예) getXXX()
데이터 멤버 변경자 (mutator)	클래스의 데이터 멤버 값을 변경하기 위한 기능 제공. 예) setXXX()

◆ dot (.) 연산자

- 클래스나 객체의 멤버 데이터와 멤버 함수를 사용할 때 dot(.) 연산자를 사용하여 멤버 관계를 나타냄



속성 (attribute)과 메소드 (method)

◆ 속성 (attribute)

- 클래스 (class)와 클래스로 부터 생성된 인스턴스(instance)가 가지는 멤버 데이터 또는 멤버 함수
- 클래스가 구현하는 기능과 정보를 의미
- 클래스/인스턴스의 이름에 dot(.)을 붙이고 속성을 지정
- 클래스/인스턴스의 속성 설정/변경 및 접근을 위한 멤버함수 (메소드)를 클래스가 제공
- 예)
t = turtle.Turtle()
t.color('red')

◆ 메소드 (method)

- 클래스와 인스턴스가 제공하는 멤버 함수
- 클래스/인스턴스의 이름에 dot(.)을 붙이고 메소드를 지정
- 예)
t.forward(100)

파이썬과 객체지향형 프로그래밍

◆ 파이썬은 객체지향형 프로그래밍 기반

- 파이썬에서 사용되는 모든 자료형, 모듈 및 패키지는 객체지향형으로 구현되어 있음
- 따라서 자료형, 모듈 및 패키지에 dot(.) 연산자를 사용하여 속성(멤버 함수와 멤버 데이터)를 지정하게 됨
- 가장 기본적인 객체: PyObject (파이썬 객체)
- 객체 지향형 프로그래밍을 사용함으로써 모든 자료형을 하나의 체계로 관리할 수 있고, 동적 자료형 (dynamic typing)이 가능함

터틀 그래픽 예제

터틀 그래픽의 기본 함수

터틀 그래픽 기본 함수	설 명
Turtle()	터틀 객체의 생성
window_width()	윈도우 넓이 읽기
window_height()	윈도우 높이 읽기
setup(width, height)	캔버스의 크기를 가로 (width)와 세로(height)로 설정
shape()	터틀 객체의 모양을 설정 (classic: 화살 축, circle: 원, square: 사각형, triangle: 삼각형, arrow: 화살 축)
shapeseize()	터틀의 크기를 설정
bgcolor(color)	배경색을 설정
circle(radius, steps)	주어진 반지름 (radius)의 원을 그림, steps가 주어지면 다각형을 그림
color(color)	터틀 객체의 색상을 설정
dot(size)	size 크기의 점을 그림
goto(x_coord, y_coord)	x좌표와 y좌표로 이동
forward(distance)	지정된 거리만큼 현재의 방향으로 전진
backward(distance)	지정된 거리만큼 현재의 반대 방향으로 후진
left(angle)	왼쪽 (반 시계방향)으로 지정된 각도만큼 회전
right(angle)	오른쪽 (시계방향)으로 지정된 각도만큼 회전
penup()	터틀의 펜을 위로 들어 이동에 따른 그리기가 나타나지 않게 함
pendown()	터틀의 펜을 아래로 내려 이동에 따라 그리기가 나타나게 함
setposition(x, y)	터틀의 위치를 지정된 좌표(x, y)로 설정
speed()	터틀 이동 속도를 설정: 1:가장느림~9:빠름, 0: 가장빠름
textinput()	입력창을 사용하여 문자열 입력
write(label)	현재의 터틀 위치에 label 문자열을 출력



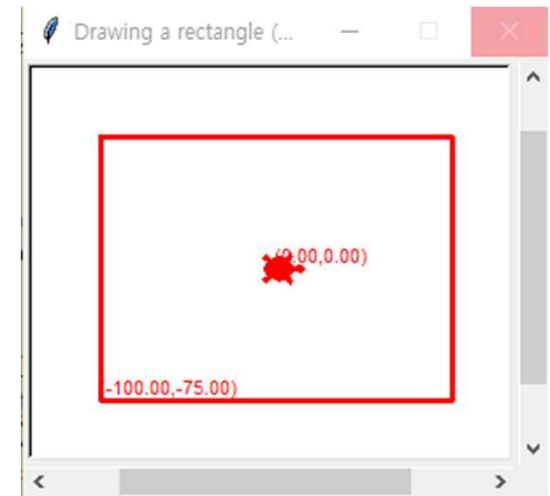
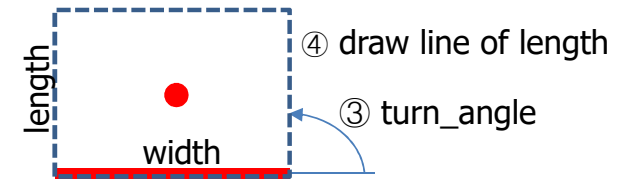
Turtle Graphic – 직 사각형 그리기

```
# draw a rectangle with Turtle graphic
import turtle
```

```
turtle.title("Drawing a rectangle")
turtle.setup(300, 250)
t = turtle.Turtle()
t.shape("turtle")
t.color("red")
t.width(3)
```

```
width, length = 200, 150
print("Drawing a rectangle(width={}, length={})".format(width, length))
```

```
turn_angle = 90
t.home(); t.write(t.position())
start_x, start_y = -width//2, -length//2
t.up(); t.goto((start_x, start_y)); t.down()
t.write(t.position())
t.forward(width); t.left(turn_angle)
t.forward(length); t.left(turn_angle)
t.forward(width); t.left(turn_angle)
t.forward(length); t.left(turn_angle)
t.up(); t.home(); t.down()
input("press any key to exit")
```



Turtle Graphic – 정삼각형 그리기

```
# draw a triangle
import math
import turtle

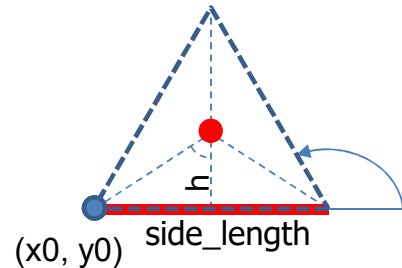
turtle.setup(400, 400) # set width and height of canvas
turtle.title("Drawing a centered triangle")
t = turtle.Turtle()
t.shape('turtle')
t.pencolor('blue')
t.width(10)

side_length = 200
turn_angle = 360 / 3
h = side_length / (2.0 * math.tan(math.pi/3))
x0, y0 = -side_length/2, -h

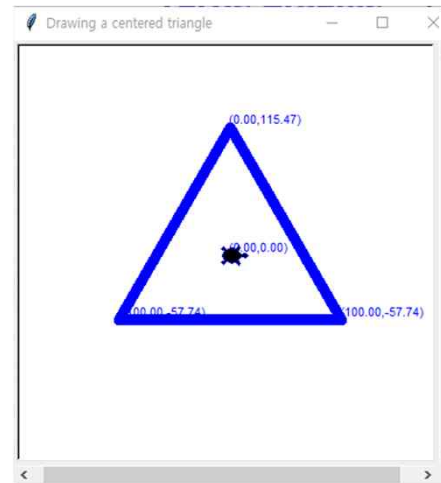
print("x0, y0 = ", x0, y0)
t.dot(5, "red")
t.write(t.pos())

t.up(); t.goto(x0, y0); t.down()
for i in range(3):
    t.write(t.pos())
    t.forward(side_length)
    t.left(turn_angle )

t.up(); t.goto(0, 0); t.down()
```



- ① calculate the start position (x0, y0)
 $\theta = 360 / 6$
 $= \pi / 3$
 $\tan(\theta) = \text{side_length} / 2 * h$
 $h = \text{side_length} / (2 * \tan(\theta))$
- ② draw line of side_length
- ③ turn left (360 / 3)
- ④ draw line of side_length



Turtle Graphic – 별 그리기

```
*2.9.1 draw_star.py - D:/YTK-Lecture(수업)-20211030/2021-2.5 예비사회인...
File Edit Format Run Options Window Help
# Draw a Star with Python turtle graphic
import turtle

turtle.setup(500, 500) #set width and height of canvas
t = turtle.Turtle()
t.shape('turtle')
t.pencolor('red')
t.width(10)
length = 200

t.up(); t.goto(-length/2, length/6); t.down()

count = 0
turn_angle = 360 * 2 / 5
while count < 5:
    t.forward(length)
    t.right(turn_angle)
    count = count + 1

t.up(); t.goto(0, 0); t.down()
```

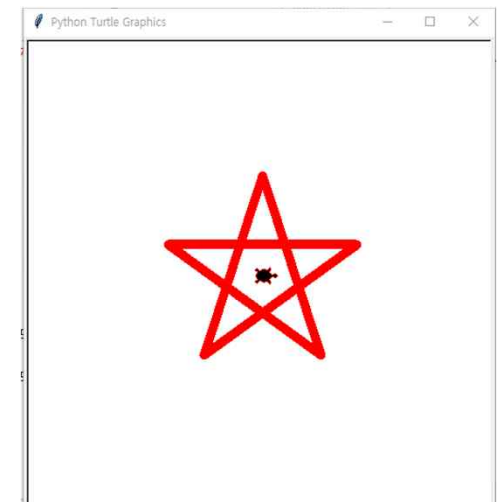
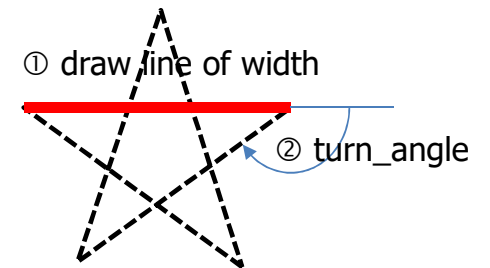
Draw a Star with Python turtle graphic
import turtle

```
turtle.setup(500, 500) #set width and height of canvas
t = turtle.Turtle()
t.shape('turtle')
t.pencolor('red')
t.width(10)
length = 200
```

```
t.penup(); t.goto(-length/2, length/6); t.pendown()
```

```
count = 0
turn_angle = 360 * 2 / 5
while count < 5:
    t.forward(length)
    t.right(turn_angle)
    count = count + 1
```

```
t.penup(); t.goto(0, 0); t.pendown()
```



References

- [1] 김영탁, **컴퓨팅 사고와 파이썬 프로그래밍**, 홍릉과학출판사, 2022. 1.
- [2] 김영탁, **자료구조와 알고리즘을 함께 배우는 파이썬 프로그래밍**, 홍릉과학출판사, 2021. 8.
- [3] Michael T. Goodrich, Roberto Tamassia, Michael H. Goldwasser, Data Structures and Algorithms in Python, Wiley, 2013.
- [4] Kent D. Lee, Steve Hubbard, Data Structures and Algorithms with Python, Springer, 2015.
- [5] Nikhil Ketakar, Deep Learning with Python, Apress, 2017.
- [6] Antonio Gulli, Amita Kappor, Sujit Pal, Deep Learning with TensorFlow 2 and Keras, Packt, 2019.
- [7] 김동근, 쉽게 배우는 파이썬 프로그래밍, 가메출판사, 2016.
- [8] 천인국, 파워유저를 위한 파이썬 Express, 생능출판사, 2020. 11.
- [9] **Python Software Foundation**, <https://www.python.org/>.
- [10] Python Tutorial, <https://docs.python.org/3/tutorial/>.
- [11] Summary **of Turtle Methods**, <http://interactivepython.org/runestone/static/IntroPythonTurtles/Summary/summary.html>.
- [12] 김영탁, 자료구조와 알고리즘을 함께 배우는 C 프로그래밍, 배움터, 2020. 2.
- [13] 김영탁, 자료구조와 알고리즘을 함께 배우는 C++ 프로그래밍, 배움터, 2020. 8.



Homework 2

Homework 2.1, 2.2

(주: 모든 프로그램 소스코드에 주석문 (comment)를 반드시 포함시킬 것 !!)

2.1 표준입력장치 (키보드)로부터 원의 반지름(radius)을 입력 받고, 그 원의 넓이(area)와 원둘레 (circumference)를 출력하는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.
(실행 예제)

```
radius = 10
Circle of radius (10) : area (314.1592), circumference(62.83184)
```

2.2 직사각형의 가로 (width)와 세로 (length)를 입력 받아 넓이(area)와 둘레(perimeter)를 계산하여 출력 하는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.
(실행 예제)

```
width, length = 100 50
Rectangle of width(100) and length(50) : area (5000), perimeter(300.0)
```



Homework 2.3

(주: 모든 프로그램 소스코드에 주석문 (comment)를 반드시 포함시킬 것 !!)

2.3 2개의 정수 a와 b를 입력받아 정수연산 $a+b$, $a-b$, $a*b$, a/b , $a//b$, $a\%b$ 를 각각 계산하여 출력하는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.
(실행 예제)

```
input a and b : 256 125
a(256) + b(125) = 381
a(256) - b(125) = 131
a(256) * b(125) = 32000
a(256) / b(125) = 2.048000
a(256) // b(125) = 2.000000
a(256) % b(125) = 6.000000
```

```
input a and b : 10 3
a( 10) + b( 3) = 13
a( 10) - b( 3) = 7
a( 10) * b( 3) = 30
a( 10) / b( 3) = 3.333333
a( 10) // b( 3) = 3.000000
a( 10) % b( 3) = 1.000000
```



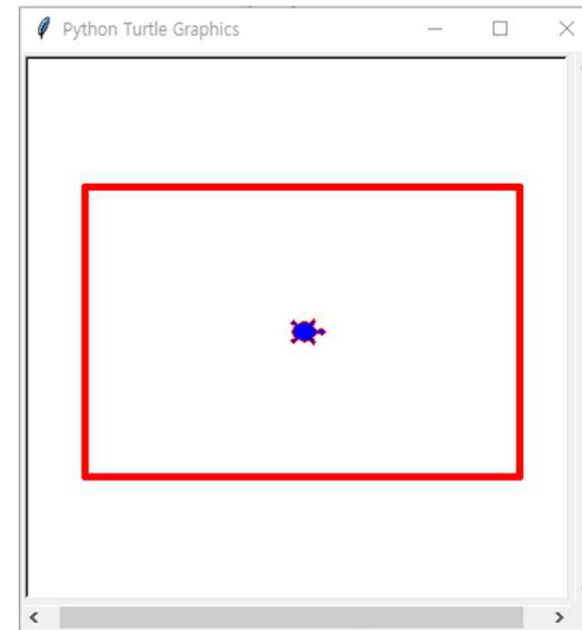
Homework 2.4

(주: 모든 프로그램 소스코드에 주석문 (comment)를 반드시 포함시킬 것 !!)

2.4 표준입력장치 (키보드)로부터 직사각형의 가로 (width) 및 세로 (length) 크기를 각각 입력 받고, 터틀 그래픽을 사용하여 지정된 크기의 사각형을 (0, 0) 좌표가 중심이 되도록 그리는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.

(실행 예제)

```
width = 300
length = 200
Drawing a rectangle of width(300), length(200) ....
```



Homework 2.5

(주: 모든 프로그램 소스코드에 주석문 (comment)를 반드시 포함시킬 것 !!)

2.5 10개의 실수 (float) 데이터를 차례로 입력받고, 입력된 데이터의 최댓값, 최솟값, 평균값을 계산하여 출력하는 파이썬 프로그램을 작성하고, 실행 결과를 제출하라.

(실행 예제)

```
input 0-th float data = 1.1
input 1-th float data = 2.2
input 2-th float data = 3.3
input 3-th float data = 4.4
input 4-th float data = 5.5
input 5-th float data = 6.6
input 6-th float data = 7.7
input 7-th float data = 8.8
input 8-th float data = 9.9
input 9-th float data = 10.1
Input data = [1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9, 10.1]
min = 1.1, max = 10.1, avg = 5.96
```



실습 2

실습 2.1 - 직각 삼각형의 면적 및 둘레 계산

```
# Lab02_1 - Calculate area and perimeter of right triangle
```

```
import math
```

```
width = int(input("width of right triangle = "))  
height = int(input("height of right triangle = "))
```

```
diagonal = math.sqrt(width*width + height*height)  
area = width * height / 2  
perimeter = width + height + diagonal
```

```
print("Right triangle of width ({} ) and height({}) => diagonal({})".format(width, height, diagonal))  
print("  area({}), perimeter({})".format(area, perimeter))
```

```
width of right triangle = 100  
height of right triangle = 100  
Right triangle of width (100) and height(100) => diagonal(141.4213562373095)  
  area(5000.0), perimeter(341.4213562373095)
```

```
width of right triangle = 200  
height of right triangle = 100  
Right triangle of width (200) and height(100) => diagonal(223.60679774997897)  
  area(10000.0), perimeter(523.606797749979)
```



실습 2.2 - 입력데이터의 통계 분석 (평균, 분산, 표준편차)

```
# while-loop, for-loop, find min and max of input data list
import math
```

```
TARGET_NUM_DATA = 10
num_data = 0
L_data = [] # empty list
L_sum = 0
print("Input {} integer data."\
      .format(TARGET_NUM_DATA))
while num_data < TARGET_NUM_DATA:
    data = int(input("data = "))
    num_data = num_data + 1
    L_data.append(data)
    L_sum = L_sum + data

avg = L_sum / num_data
diff_sq_sum = 0
for i in range(num_data):
    diff = avg - L_data[i] # difference from avg
    diff_sq_sum = diff_sq_sum + diff * diff
var = diff_sq_sum / num_data
std = math.sqrt(var)

print("Input data list = ", L_data)
print("avg = {}, var = {}, std = {}".\
      .format(avg, var, std))
```

```
Input 10 integer data.
```

```
data = 1
data = 2
data = 3
data = 4
data = 5
data = 6
data = 7
data = 8
data = 9
data = 10
```

```
Input data list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
avg = 5.5, var = 8.25, std = 2.8722813232690143
```

```
Input 10 integer data.
```

```
data = -5
data = -4
data = -3
data = -2
data = -1
data = 0
data = 1
data = 2
data = 3
data = 4
```

```
Input data list = [-5, -4, -3, -2, -1, 0, 1, 2, 3, 4]
avg = -0.5, var = 8.25, std = 2.8722813232690143
```



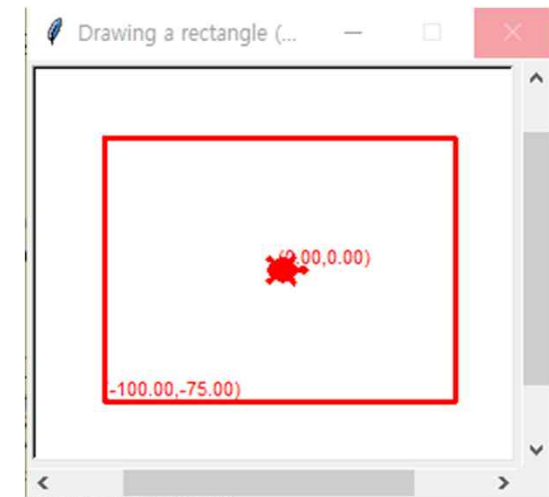
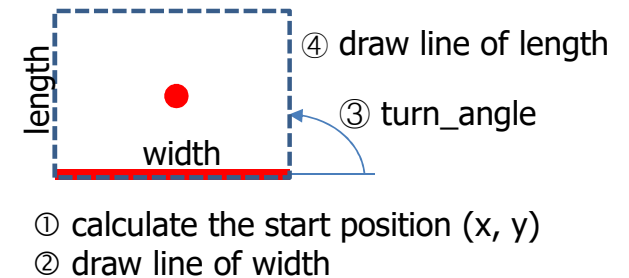
실습 2.3 - 직 사각형 그리기

```
# draw a rectangle with Turtle graphic
import turtle

turtle.title("Drawing a rectangle")
turtle.setup(300, 250)
t = turtle.Turtle()
t.shape("turtle")
t.color("red")
t.width(3)

width, length = 200, 150
print("Drawing a rectangle(width={}, length={})".format(width, length))

turn_angle = 90
t.home(); t.write(t.position())
start_x, start_y = -width//2, -length//2
t.up(); t.goto((start_x, start_y)); t.down()
t.write(t.position())
t.forward(width); t.left(turn_angle)
t.forward(length); t.left(turn_angle)
t.forward(width); t.left(turn_angle)
t.forward(length); t.left(turn_angle)
t.up(); t.home(); t.down()
input("press any key to exit")
```



실습 2.4 - 정삼각형 그리기

```
# draw a triangle
import math
import turtle

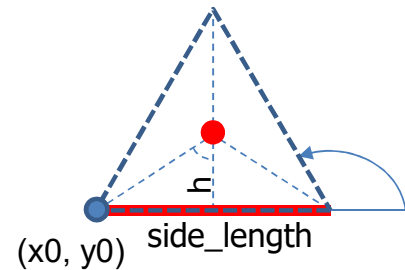
turtle.setup(400, 400) # set width and height of canvas
turtle.title("Drawing a centered triangle")
t = turtle.Turtle()
t.shape('turtle')
t.pencolor('blue')
t.width(10)

side_length = 200
turn_angle = 360 / 3
h = side_length / (2.0 * math.tan(math.pi/3))
x0, y0 = -side_length/2, -h

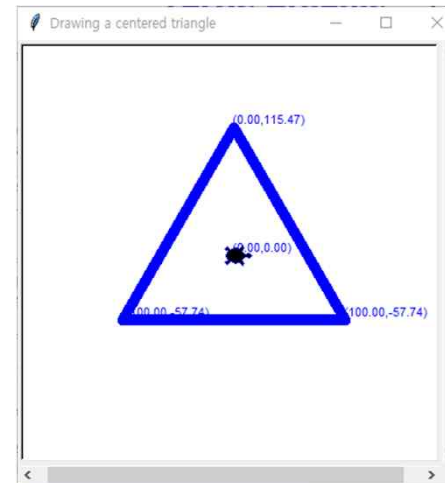
print("x0, y0 = ", x0, y0)
t.dot(5, "red")
t.write(t.pos())

t.up(); t.goto(x0, y0); t.down()
for i in range(3):
    t.write(t.pos())
    t.forward(side_length)
    t.left(turn_angle)

t.up(); t.goto(0, 0); t.down()
```



- ① calculate the start position (x0, y0)
 $\theta = 360 / 6$
 $= \pi / 3$
 $\tan(\theta) = \text{side_length} / 2 * h$
 $h = \text{side_length} / (2 * \tan(\theta))$
- ② draw line of side_length
- ③ turn left (360 / 3)
- ④ draw line of side_length



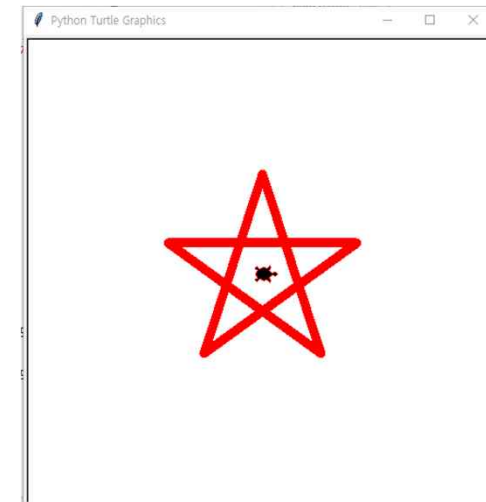
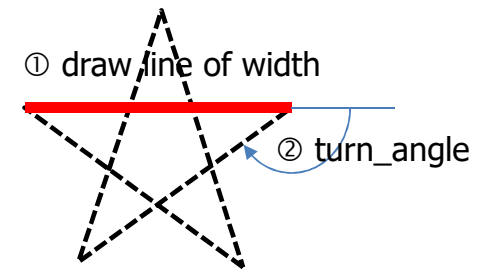
실습 2.5 - 별 그리기

```
# Draw a Star with Python turtle graphic
import turtle

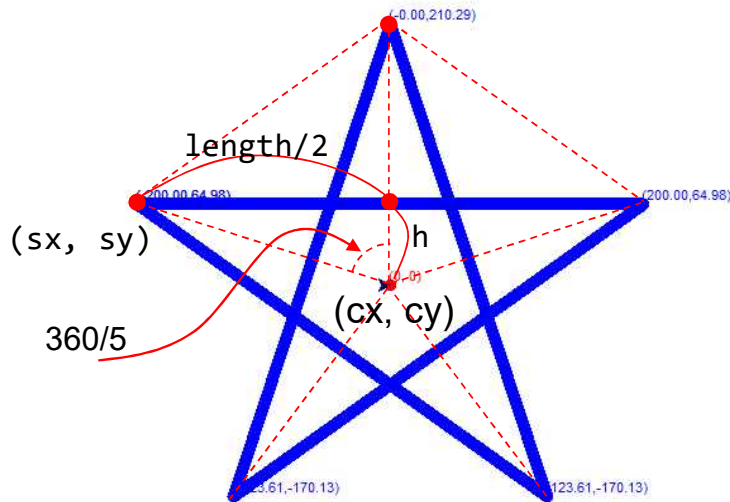
turtle.setup(500, 500) #set width and height of canvas
t = turtle.Turtle()
t.shape('turtle')
t.pencolor('red')
t.width(10)
length = 200
cx, cy = 0, 0 # center position
sx = cx - length / 2
sy = cy + length / 6 # 이 부분은 좀 더 정확한 계산을 할 것
t.penup(); t.goto((sx, sy)); t.pendown()

count = 0
turn_angle = 360 * 2 / 5
while count < 5:
    t.forward(length)
    t.right(turn_angle)
    count = count + 1

t.penup(); t.goto(0, 0); t.pendown()
```



별 그리기에서 시작점 좌표 계산



cx, cy : center position coordinate x, y
 sx, sy : start position coordinate
 $sx = cx - \text{length}/2$
 $\text{theta_rad} = \text{math.radians}(360 / 5) \# = \text{math.pi} / 5$
 $\tan(\text{theta_rad}) = (\text{length}/2) / h$
 $h = \text{length} / (2 * \tan(\text{theta})) = \text{length} / (2 * \tan(\text{math.pi}/5))$
 $sy = cy + h$

calculation of start position of star

import math

line_length = 100

cx, cy = 0, 0

sx = cx - line_length / 2

theta_rad = math.radians(360 / 5)

sy = cy + line_length / (2 * math.tan(theta_rad))

print("cx ({}), cy ({}), line_length ({})=> sx ({}), sy({})".format(cx, cy, line_length, sx, sy))

print("2 * math.tan(theta_rad) = {}".format(2 * math.tan(theta_rad)))

cx (0), cy (0), line_length (100) => sx (-50.0), sy(16.24598481164532)
 2 * math.tan(theta_rad) = 6.155367074350505

여러 개의 별 그리기

Simple Python Program to Draw a Star with center positioning

```
import turtle
import math
```

```
turtle.setup(500, 500) #set width and height of canvas
turtle.title("Drawing star(s) at given position")
t = turtle.Turtle()
t.shape('classic')
```

```
num_vertices = 5 # 5 vertices in star
```

```
turn_angle = 2*360/num_vertices
```

```
t.up(); t.home(); t.down()
```

```
t.dot(10, "red"); t.write(t.position())
```

```
while True:
```

```
    cx, cy = map(int, input("input center_x and center_y : ").split(' '))
```

```
    center = (cx, cy)
```

```
    pen_thickness, line_length = map(int, input("input pen_thickness and line_length of star : ").split(' '))
```

```
    line_color = input("line_color of star (e.g., red, blue) = ")
```

```
    t.pencolor(line_color)
```

```
    t.up(); t.goto(center); t.down()
```

```
    t.dot(10, "blue"); t.write(center)
```

```
    sx = cx - line_length/2
```

```
    theta = math.radians(360 / num_vertices) # convert angle in degree into angle in radian
```

```
    h = line_length / (2 * math.tan(theta))
```

```
    sy = cy + h
```

```
    t.width(pen_thickness)
```

```
    t.penup(); t.goto(sx, sy); t.dot(10, "blue")
```

```
    t.write(t.position()); t.pendown() #pen down to draw
```

```
    for i in range(num_vertices):
```

```
        t.forward(line_length)
```

```
        t.right(turn_angle)
```

```
    t.up(); t.home(); t.down()
```

```
input center_x and center_y : 0 0
input pen_thickness and line_length of star : 5 100
line_color of star (e.g., red, blue) = red
input center_x and center_y : -100 -100
input pen_thickness and line_length of star : 5 100
line_color of star (e.g., red, blue) = blue
input center_x and center_y : 100 100
input pen_thickness and line_length of star : 5 100
line_color of star (e.g., red, blue) = blue
input center_x and center_y :
```

