버블 정렬(bubble sort)

0. 알고리즘 연습 방법

- 알고리즘을 잘 작성하기 위해서는 잘 작성된 알고리즘을 이해하고, 스스로 만들어봐야 함

알고리즘 연습 방법

1. 연습장과 펜을 준비하자.

2. 알고리즘 문제를 읽고 분석한 후에,

3. 간단하게 테스트용으로 매우 간단한 경우부터 복잡한 경우 순서대로 생각해보면서, 연습장과 펜을 이용하여 알고리즘을 생각해본다.

4. 가능한 알고리즘이 보인다면, 구현할 알고리즘을 세부 항목으로 나누고, 문장으로 세부 항목을 나누어서 적어본다.

5. 코드화하기 위해, 데이터 구조 또는 사용할 변수를 정리하고,

6. 각 문장을 코드 레벨로 적는다.

7. 데이터 구조 또는 사용할 변수가 코드에 따라 어떻게 변하는지를 손으로 적으면서, 임의 데이터로 코드가 정상 동작하는지를 연습장과 펜으로 검증한다.

---

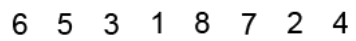
1. 정렬(sorting)이란?

- 정렬(sorting) : 어떤 데이터들이 주어졌을 때 이를 정해진 순서대로 나열하는 것
- 정렬은 프로그램 작성시 빈번하게 필요로 함
- 다양한 알고리즘이 고안되었으며, 알고리즘 학습의 필수

---

2. 버블 정렬(bubble sort)란?

- **두 인접한 데이터를 비교해서, 앞에 있는 데이터가 뒤에 있는 데이터보다 크면, 자리를 바꾸는 정렬 알고리즘**



---

3. 어떻게 코드로 만들까?

단계적으로 생각해보자

- 데이터가 네 개 일때 (데이터 갯수에 따라 복잡도가 떨어지는 것은 아니므로, 네 개로 바로 로직을 이해해보자.)
  - 예: data_list = [1, 9, 3, 2]
    - 1차 로직 적용
      - 1 와 9 비교, 자리바꿈없음 [1, 9, 3, 2]
      - 9 와 3 비교, 자리바꿈 [1, 3, 9, 2]
      - 9 와 2 비교, 자리바꿈 [1, 3, 2, 9]
    - 2차 로직 적용
      - 1 와 3 비교, 자리바꿈없음 [1, 3, 2, 9]
      - 3 과 2 비교, 자리바꿈 [1, 2, 3, 9]
      - 3 와 9 비교, 자리바꿈없음 [1, 2, 3, 9]
    - 3차 로직 적용

- 1 과 2 비교, 자리바꿈없음 [1, 2, 3, 9]
- 2 과 3 비교, 자리바꿈없음 [1, 2, 3, 9]
- 3 과 9 비교, 자리바꿈없음 [1, 2, 3, 9]

---

4. 알고리즘 구현

- 특이점 찾아보기
  - n개의 리스트가 있는 경우 최대 n-1번의 로직을 적용한다.
  - 로직을 1번 적용할 때마다 가장 큰 숫자가 뒤에서부터 1개씩 결정된다.
  - 로직이 경우에 따라 일찍 끝날 수도 있다. 따라서 로직을 적용할 때 한 번도 데이터가 교환된 적이 없다면 이미 정렬된 상태이므로 더 이상 로직을 반복 적용할 필요가 없다.

리스트 데이터	1회 로직 적용	2회 로직 적용	3회 로직 적용	4회 로직 적용
9, 1, 7	1, 7, 9	–		
9, 7, 1	7, 1, 9	1, 7, 9		
1, 9, 3, 2	1, 3, 2, 9	1, 2, 3, 9	–	
9, 3, 2, 1	3, 2, 1, 9	2, 1, 3, 9	1, 2, 3, 9	
9, 7, 5, 3, 1	7, 5, 3, 1, 9	5, 3, 1, 7, 9	3, 1, 5, 7, 9	1, 3, 5, 7, 9
5, 7, 3, 9, 4	5, 3, 7, 4, 9	3, 5, 4, 7, 9	3, 4, 5, 7, 9	–
1, 9, 3, 2, 7	1, 3, 2, 7, 9	1, 2, 3, 7, 9	–	–

1. for num in range(len(data_list))반복
2. swap=0(교환이 되었는지를 확인하는 변수를 두자)
3. 반복문 안에서, for index in range(len(data_list)-num-1) n-1번 반복해야 하므로
4. 반복문안의 반복문 안에서, if data_list[index]>data_list[index+1]이면
5. swap
6. swat+=1
7. 반복문 안에서, if swap==0이면, beak

```
def bubblesort(data):
  for index in range(len(data)-1):
    swap = False
    for index2 in range(len(data)-index-1):
      if data[index2]>data[index2+1]:
        data[index2],data[index2+1] = data[index2+1],data[index2]
        swap=True
    if swap == False:
      break
  return data
```

```
import random

data_list = random.sample(range(100),50)
print(bubblesort(data_list))
>>[1, 6, 8, 15, 16, 20, 21, 22, 24, 25, 27, 28, 29, 30, 31, 33, 34, 36, 39, 40, 42, 43, 45, 53, 56, 57, 58, 61, 62, 64, 67, 68, 71, 72
```

---

5. 알고리즘 분석

- 반복문이 두 개 O(n^2)
- 완전 정렬이 되어 있는 상태라면 O(n)