list

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 문제
    - 입력: nums = [2, 7, 11, 15], target = 9
    - 출력: [0, 1]
    - 설명

      nums[0] + nums[1] = 2 + 7 = 9

      따라서 0 과 1을 리턴

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 브루트 포스(Brute – Force, 무차별 대입 방식)로 계산
    - 입력: nums = [2, 7, 11, 15], target = 9

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 브루트 포스(Brute – Force, 무차별 대입 방식)로 계산

```
def bruteforce(nums, target):
    for i in range(len(nums)):
        for j in range(i + 1, len(nums)):
            if nums[i] + nums[j] == target:
                return [i, j]

nums = [2, 7, 11, 15]
print(bruteforce(nums, 9))
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 브루트 포스(Brute – Force, 무차별 대입 방식)로 계산

```
public class ArrayTest {
    public static int [] bruthForce(int []nums, int target) {
            int [] ar = new int[2];
            for(int i=0; i<nums.length-1; i++) {
                    for(int j=i + 1; j<nums.length; j++) {
                            if(nums[i] + nums[j] == target) {
                                    ar[0] = i;
                                    ar[1] = j;
                            }
                    }
            }
            return ar;
    }

    public static void main(String [] args) {
            int [] ar = {2, 7, 11, 15};
            int target = 9;
            System.out.println(Arrays.toString(bruthForce(ar, target)));
    }
}
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ in 연산자 이용

```
def inOperator(nums, target):
    for i, n in enumerate(nums):
        complement = target - n
        if complement in nums[i + 1:]:
            return [nums.index(n), nums[i + 1:].index(complement) + (i + 1)]

nums = [2, 7, 11, 15]
print(inOperator(nums, 9))
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 자바 배열의 순회

```
public class ArrayTest {
    public static int [] forin(int []nums, int target) {
        int [] ar = new int[2];
        int i=0;
        inner : for(int n : nums) {
            int complement = target - n;
            int [] imsi = Arrays.copyOfRange(nums, 1, nums.length);
            int j=0;
            for(int k : imsi) {
                if(complement == k) {
                    ar[0] = i;
                    ar[1] = j + 1;
                    break inner;
                }
                j++;
            }
            i++;
        }
        return ar;
    }
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 자바 배열의 순회

```
        public static void main(String [] args) {
                int [] ar = {2, 7, 11, 15};
                int target = 9;
                System.out.println(Arrays.toString(forin(ar, target)));
        }
}
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ Dictionary 이용

```
def dictionary(nums, target):
    nums_map = {}
    # 키와 값을 바꿔서 딕셔너리로 저장
    for i, num in enumerate(nums):
        nums_map[num] = i

    # 타겟에서 첫 번째 수를 뺀 결과를 키로 조회
    for i, num in enumerate(nums):
        if target - num in nums_map and i != nums_map[target - num]:
            return [i, nums_map[target - num]]

nums = [2, 7, 11, 15]
print(dictionary(nums, 9))
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ Map 이용

```
public class ArrayTest {
    public static int [] map(int []nums, int target) {
        int [] ar = new int[2];
        Map<Integer, Integer> map = new HashMap<>();
        int i=0;
        for(int num : nums) {
            map.put(num, i);
            i++;
        }
        Set<Integer> set = map.keySet();
        i = 0;
        for(int x : set) {
            if(set.contains(target-x) && i != map.get(target-x)) {
                ar[0] = i;
                ar[1] = map.get(target-x);
                break;
            }
            i++;
        }
        return ar;
    }
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ Map 이용

```
        public static void main(String [] args) {
                int [] ar = {2, 7, 11, 15};
                int target = 9;
                System.out.println(Arrays.toString(map(ar, target)));
        }
    }
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 조회 구조 개선

```
def searchAdvance(nums, target):
    nums_map = {}
    # 하나의 `for`문으로 통합
    for i, num in enumerate(nums):
        if target - num in nums_map:
            return [nums_map[target - num], i]
        nums_map[num] = i

nums = [2, 7, 11, 15]
print(searchAdvance(nums, 9))
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 조회 구조 개선

```
public class ArrayTest {
    public static int [] dict(int []nums, int target) {
        int [] ar = new int[2];

        Map<Integer, Integer> map = new HashMap<>();

        int i=0;
        for(int num : nums) {
            map.put(num, i);
            if(map.containsKey(target-num)) {
                ar[0] = map.get(target-num);
                ar[1] = i;
                return ar;
            }
            i++;
        }
        return ar;
    }
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 조회 구조 개선

```
        public static void main(String [] args) {
                int [] ar = {2, 7, 11, 15};
                int target = 9;
                System.out.println(Arrays.toString(dict(ar, target)));
        }
    }
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 2개의 포인터 이용

```
def twoPointer(nums, target):
    left, right = 0, len(nums) - 1
    while not left == right:
        # 합이 타겟보다 크면 오른쪽 포인터를 왼쪽으로
        if nums[left] + nums[right] < target:
            left += 1
        # 합이 타겟보다 작으면 왼쪽 포인터를 오른쪽으로
        elif nums[left] + nums[right] > target:
            right -= 1
        else:
            return [left, right]

nums = [2, 7, 11, 15]
print(twoPointer(nums, 9))
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 2개의 포인터 이용

```
public class ArrayTest {
    public static int [] twoPointer(int []nums, int target) {
            int [] ar = new int[2];

            int left = 0;
            int right = nums.length - 1;

            while(left != right) {
                    if(nums[left] + nums[right] < target)
                            left += 1;
                    else if(nums[left] + nums[right] > target)
                            right -= 1;
                    else {
                            ar[0] = left;
                            ar[1] = right;
                            break;
                    }
            }
            return ar;
    }
```

연습문제

- 덧셈하여 타겟을 만들 수 있는 배열의 두 숫자 인덱스를 리턴
  - ✓ 2개의 포인터 이용

```
        public static void main(String [] args) {
                int [] ar = {2, 7, 11, 15};
                int target = 9;
                System.out.println(Arrays.toString(twoPointer(ar, target)));
        }
}
```