알고리즘 과제 02

2022110151 데이터사이언스전공 이주연

----------문제 1. 버블정렬 Bubble Sort----------

1. **비순환적** 버블 정렬 구현

- **유사코드**

```
함수 BubbleSort( A[], n )
    Sorted ← False  (0)
    round ← 0 (초기 상태부터 시작해 단계별로 정렬되는 모습을 확인)
    while ( Sorted == False )
            Sorted ← True  (1)
            배열 A[]를 출력하기 (현재 상태)
            for i ← 1 to n-1 do
                    if  A[i-1] > A[i]  then (앞 요소가 뒤보다 큰 경우)
                        A[i-1]과 A[i]를 바꾸기(swap)
                        Sorted ← False (0)
            round ← round + 1  (단계 하나 증가)
      정렬된 배열 A[] 출력하기
```

- **프로그램 작성 & 결과 출력**

```
Round 0: 30 20 40 10 5 10 30 15
Round 1: 20 30 10 5 10 30 15 40
Round 2: 20 10 5 10 30 15 30 40
Round 3: 10 5 10 20 15 30 30 40
Round 4: 5 10 10 15 20 30 30 40
Sorted Array: 5 10 10 15 20 30 30 40
```

단계가 변화함에 따라 배열 a[]이 오른쪽부터 정렬되고 있음을 알 수 있다.

Round 1에서 정렬이 한 번 완료되어 정렬된 수 40이 맨 오른쪽에 위치하였고, 그 다음단계에서는 30 40, 그 후로는 30 30 40 과 같이 정렬되어 5 10 10 15 20 30 30 40으로 최종 정렬된 모습을 알 수 있다.

2. **순환적** 버블정렬로 구현

- **유사코드**

```
함수 bubbleSortRecursive(A[], n, totalSize)
    만약 n == 1 이면:
        종료하기   (배열이 이미 정렬되었으므로 종료)

    Sorted ← True  (1)  (배열이 정렬되었는지 확인)
    배열 A[] 출력하기   (현재 배열 상태 출력하기)

    for i ← 1 to n-1 do   (배열의 1번부터 n-1번까지 반복하기)
        if A[i-1] > A[i] then  (만약 앞의 요소가 뒤의 요소보다 크면)
            A[i-1] ↔ A[i]   (두 요소를 교환하기)
            Sorted ← False  (0)  (배열이 정렬되지 않았으니 다시 정렬해야 한다는 의미)

    만약 Sorted == True이면:
        종료하기 (배열이 정렬되었으므로 종료)

    bubbleSortRecursive(A, n-1, totalSize)  (배열 크기를 하나 줄여서 재귀 호출하기)
```

- **프로그램 작성 & 결과 출력**

```
Recursive ver. - Bubble Sort
Checking round: 30 20 40 10 5 10 30 15
Checking round: 20 30 10 5 10 30 15 40
Checking round: 20 10 5 10 30 15 30 40
Checking round: 10 5 10 20 15 30 30 40
Checking round: 5 10 10 15 20 30 30 40
Sorted Array: 5 10 10 15 20 30 30 40
```

순환적 버블정렬의 경우에는 main함수에서 bubbleSortRecursive함수 호출 시 bubbleSortRecursive(A, n - 1, totalSize);의 재귀적 호출로 인해 정렬이 계속 이루어지도록 하였다. A[i-1]값과 A[i]값을 비교해 앞의 요소가 크면 값이 바뀌도록 설정하였고, 그 결과 한 round가 끝나면 맨 오른쪽 수부터 시작해 차례대로 정렬되며 재귀적 호출을 통해 round가 변해감에 따라 정렬됨을 알 수 있다.(40 -> 30 40 -> 30 30 40 -> ...)

----------문제 2. 퀵 정렬 Quick Sort----------

- **순환적 퀵 정렬 구현 (코드 및 결과 출력)**

정렬 결과

```
Quick Sort - 1. Recursive(재귀) ver.
정렬 전 배열: 24755 4206 27412 13393 18686 1527 14717 3404 32171 1113
[STEP 1] Pivot:24755 → 27412 32171 24755 13393 18686 1527 14717 3404 4206 1113
[STEP 2] Pivot:27412 → 32171 27412 24755 13393 18686 1527 14717 3404 4206 1113
[STEP 3] Pivot:13393 → 32171 27412 24755 14717 18686 13393 1527 3404 4206 1113
[STEP 4] Pivot:14717 → 32171 27412 24755 18686 14717 13393 1527 3404 4206 1113
[STEP 5] Pivot:1527 → 32171 27412 24755 18686 14717 13393 4206 3404 1527 1113
[STEP 6] Pivot:4206 → 32171 27412 24755 18686 14717 13393 4206 3404 1527 1113
정렬 후 배열 (내림차순): 32171 27412 24755 18686 14717 13393 4206 3404 1527 1113
```

기본적으로 partition 함수는 배열을 pivot을 기준으로 분할하는 역할을 할 수 있도록 코드를 작성했다. 배열에서 pivot보다 "크거나 같은 값들"은 왼쪽으로 가고, pivot보다 "작은 값들"은 오른쪽으로 갈 수 있도록 이동시켜, 최종적으로 내림차순 정렬이 이루어지도록 하였다.

- **비순환적 퀵 정렬 구현 (코드 및 결과 출력)**

정렬 결과

```
Quick Sort - 2. Not Recursive(stack 활용) ver.
정렬 전 배열: 25049 21291 29555 16403 29494 27253 32693 2094 10886 4843
[STEP 1] Pivot:4843 → 25049 21291 29555 16403 29494 27253 32693 10886 4843 2094
[STEP 2] Pivot:10886 → 25049 21291 29555 16403 29494 27253 32693 10886 4843 2094
[STEP 3] Pivot:32693 → 32693 21291 29555 16403 29494 27253 25049 10886 4843 2094
[STEP 4] Pivot:25049 → 32693 29555 29494 27253 25049 16403 21291 10886 4843 2094
[STEP 5] Pivot:27253 → 32693 29555 29494 27253 25049 16403 21291 10886 4843 2094
[STEP 6] Pivot:29494 → 32693 29555 29494 27253 25049 16403 21291 10886 4843 2094
[STEP 7] Pivot:21291 → 32693 29555 29494 27253 25049 21291 16403 10886 4843 2094
정렬 후 배열 (내림차순): 32693 29555 29494 27253 25049 21291 16403 10886 4843 2094
```

순환적 퀵 정렬과는 달리 quick_sort_iterative함수에서는 "스택"을 활용해 비순환적인 퀵 정렬을 구현했다. 왼쪽 범위와 오른쪽 범위를 스택에 넣고, 스택에서 꺼내 해당 범위에 대해 partition을 반복하도록 코드를 작성했다.