2021.Algorithm – HW1

20172674
신동녁

1.

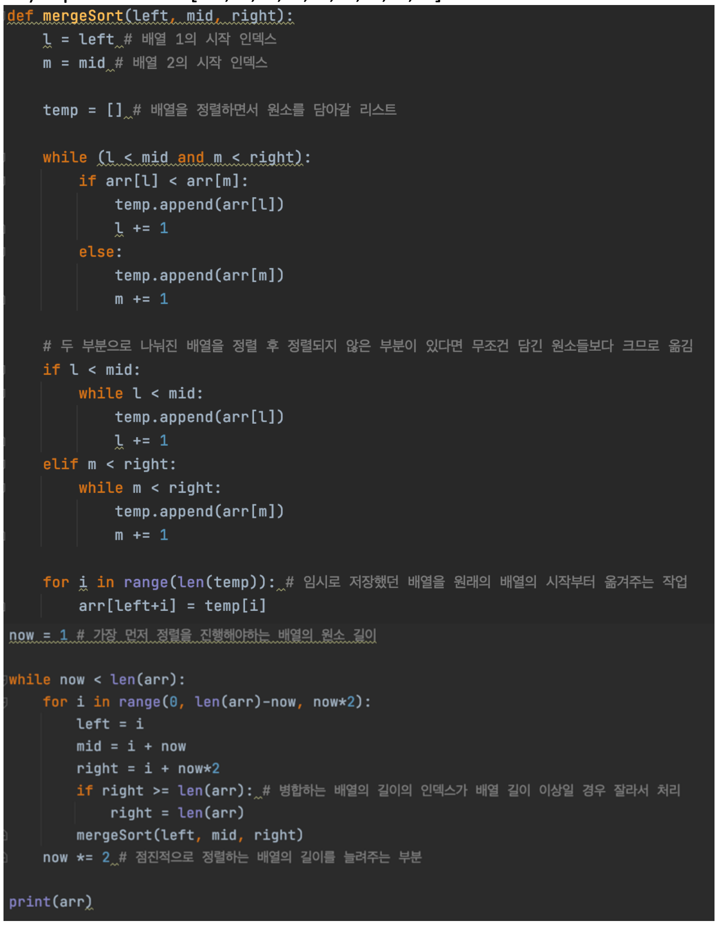
In merge sort, there are 6-array access in total for the following reasons.

At first, we assume that there are N of data in the array, and we've got a temporary array in our function. This is how it works, at first, we should copy all of data from original array to temporary array while comparing them. If the comparing is end, we move the data existed in the temporary array to where it used to be. In this process, we access to the array for **"reading"** by copying in each steps(origin->temp, temp->origin), which means **2N** of access is executed. After copying data, we access to the array for **"writing"** by moving in each steps, this also gives **2N** of access. Finally, when comparing two components from divided arrays, **2N** of **"reading"** access is added.

For these reasons, we have **6N of access**, which include 4 reading access and 2 writing access, in top-down merge sorting.

2.

Ex) input -> arr = [10, 5, 4, 2, 6, 1, 7, 9, 3]



```python
def mergeSort(left, mid, right):
    l = left # 배열 1의 시작 인덱스
    m = mid # 배열 2의 시작 인덱스

    temp = [] # 배열을 정렬하면서 원소를 담아갈 리스트

    while (l < mid and m < right):
        if arr[l] < arr[m]:
            temp.append(arr[l])
            l += 1
        else:
            temp.append(arr[m])
            m += 1

    # 두 부분으로 나눠진 배열을 정렬 후 정렬되지 않은 부분이 있다면 무조건 담긴 원소들보다 크므로 옮김
    if l < mid:
        while l < mid:
            temp.append(arr[l])
            l += 1
    elif m < right:
        while m < right:
            temp.append(arr[m])
            m += 1

    for i in range(len(temp)): # 임시로 저장했던 배열을 원래의 배열의 시작부터 옮겨주는 작업
        arr[left+i] = temp[i]
now = 1 # 가장 먼저 정렬을 진행해야하는 배열의 원소 길이

while now < len(arr):
    for i in range(0, len(arr)-now, now*2):
        left = i
        mid = i + now
        right = i + now*2
        if right >= len(arr): # 병합하는 배열의 길이의 인덱스가 배열 길이 이상일 경우 잘라서 처리
            right = len(arr)
        mergeSort(left, mid, right)
    now *= 2 # 점진적으로 정렬하는 배열의 길이를 늘려주는 부분

print(arr)
```

By increasing a length that we compare, cut the input array for the length we decide and put it into the merge function continuously. Precise explanation is in the comment written in the attached code file.