Homework #5

2020.09.29.

1. 합병 정렬 알고리즘 mergeSort()에서 합병 단계는 이미 정렬이 된 부분 배열들을 반복해서 합병한다. 합병 정렬의 또 다른 하향식(top-down) 버전은 주어진 배열 속에 이미 정렬되어 있는 부분 배열, 즉 런(run)을 합병함으로써 이 문제를 해결한다. 배열에서 런이 결정되면, 차례대로 두 개의 런에 대한 합병이 수행된다. 예를 들어 배열 [6, 7, 8, 3, 4, 1, 5, 9, 10, 2]에서 런을 구하면 다음과 같다.

   [6, 7, 8], [3, 4], [1, 5, 9, 10], [2]

   먼저 런 [6, 7, 8]과 런 [3, 4]를 합병하면 [3, 4, 6, 7, 8]이 되고, 다음으로 런 [1, 5, 9, 10]과 런 [2]을 합병하면 [1, 2, 5, 9, 10]이 된다. 마지막으로 런 [3, 4, 6, 7, 8]과 [1, 2, 5, 9, 10]을 합병하면 [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]이 된다. 주어진 배열의 부분 정렬(즉 런)을 이용하는 합병 정렬을 **자연 합병 정렬 (natural merge sort)**이라 한다.

   (1) 최초 입력 배열에서 런을 구하는 과정을 ADL로 작성하라.
   (2) 합병(merge) 알고리즘을 사용하여 런을 2개씩 합병하여 자연 합병 정렬을 수행하는 파이썬 프로그램을 작성하라.
   (3) 프로그램의 실행 시간을 합병 정렬과 비교해 보라.

2. **토너먼트 정렬(tournament sort)**은 선택 정렬과 히프 정렬의 아이디어를 사용한다. 먼저 배열 b[]를 사용하여 완전 이진 트리를 만들고, 서로 다른 키 값을 가지는 배열 a[]로부터 n개의 원소를 이 트리의 리프로 복사한다. 다음으로 가장 큰 원소가 루트에 복사될 때까지 형제 원소의 쌍 중에서 큰 것을 부모 노드로 복사하는 스포츠 토너먼트와 같이 진행한다. 토너먼트를 통해 가장 큰 원소가 루트에 복사되면 가장 큰 원소를 가지고 있는 b[1]을 a[n]으로 복사한다. 그런 다음 트리를 따라 내려가면서 토너먼트에 참여한 노드 중에서 가장 큰 원소를 가지고 있는 노드의 키 값을 0으로 변경한다. 트리의 리프에 도달하면 다시 트리를 따라 올라가면서 이전과 동일한 과정을 반복하면 정렬된 배열을 얻을 수 있다. 첫 번째 단계로 7개의 키 값 [4, 6, 7, 3, 5, 1, 2]를 완전 이진 트리를 나타내는 배열 b[]에 복사한 결과는 다음과 같다.

   [0, 0, 0, 0, 0, 0, 0, 0, 4, 6, 7, 3, 5, 1, 2, 0]

   두 번째 단계로 토너먼트를 거쳐 가장 큰 원소를 b[1]에 저장한 결과는 다음과 같다.

   [0, 7, 7, 5, 6, 7, 5, 2, 4, 6, 7, 3, 5, 1, 2, 0]

   이후 반복되는 과정은 다음과 같다.

   [0, 6, 6, 5, 6, 3, 5, 2, 4, 6, 0, 3, 5, 1, 2, 0]
   [0, 5, 4, 5, 4, 3, 5, 2, 4, 0, 0, 3, 5, 1, 2, 0]
   [0, 4, 4, 2, 4, 3, 1, 2, 4, 0, 0, 3, 0, 1, 2, 0]
   [0, 3, 3, 2, 0, 3, 1, 2, 0, 0, 0, 3, 0, 1, 2, 0]
   [0, 2, 0, 2, 0, 0, 1, 2, 0, 0, 0, 0, 0, 1, 2, 0]
   [0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0]

   (1) 최초 입력 배열이 서로 다른 키 값을 가지는 경우 토너먼트 정렬 알고리즘을 ADL로 작성하라.
   (2) 토너먼트 정렬 알고리즘을 파이썬으로 구현하고, 실행시간을 측정해 보라.