Algorithms

Assignment #1

Yoon Suk Kang
School of Computer Science
Chungbuk National University

Assignment #1: Description (Cont.)

❑ Overview

- Sorting a list of unordered numbers **in ascending order** using **sorting algorithms**

❑ 3 sorting algorithms to implement

- Insertion sort
- Merge sort
- Merge-insertion sort
  - Insertion sort on small subarrays in merge sort

Assignment #1: Description (Cont.)

- **Insertion sort**

```
INSERTION-SORT(A)
1  for j = 2 to A.length
2      key = A[j]
3      // Insert A[j] into the sorted sequence A[1 .. j − 1].
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

Assignment #1: Description (Cont.)

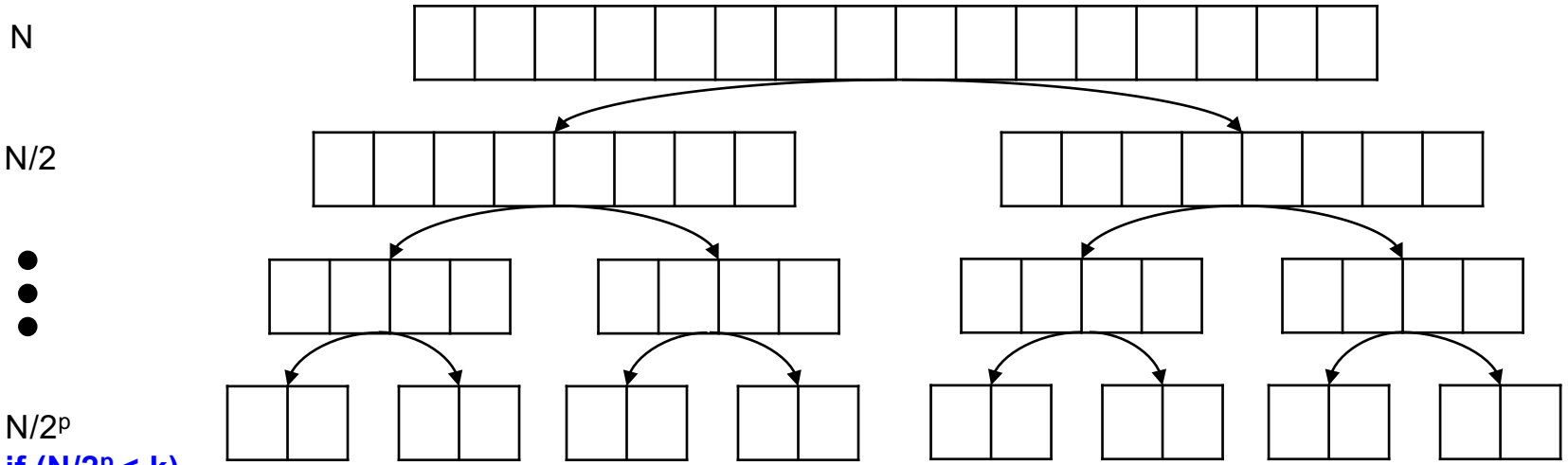
- **Merge sort**

```
MERGE-SORT(A, p, r)
1  if p < r
2      q = ⌊(p + r)/2⌋
3      MERGE-SORT(A, p, q)
4      MERGE-SORT(A, q + 1, r)
5      MERGE(A, p, q, r)
```

```
MERGE(A, p, q, r)
1   n1 = q − p + 1
2   n2 = r − q
3   let L[1 .. n1 + 1] and R[1 .. n2 + 1] be new arrays
4   for i = 1 to n1
5       L[i] = A[p + i − 1]
6   for j = 1 to n2
7       R[j] = A[q + j]
8   L[n1 + 1] = ∞
9   R[n2 + 1] = ∞
10  i = 1
11  j = 1
12  for k = p to r
13      if L[i] ≤ R[j]
14          A[k] = L[i]
15          i = i + 1
16      else A[k] = R[j]
17          j = j + 1
```

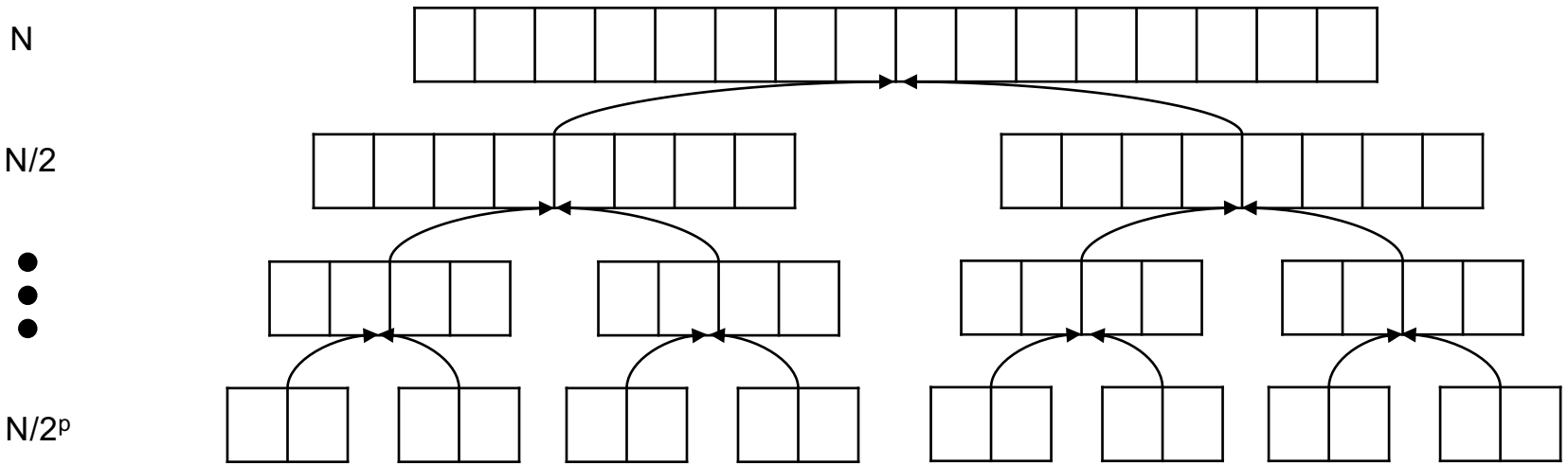
Assignment #1: Description (Cont.)

- **Merge-insertion sort (a modification of merge sort)**
  - Divide a given list **using the standard dividing mechanism**
  - Sort $n/2^p$ sublists of length k **using the insertion sort**



Assignment #1: Description (Cont.)

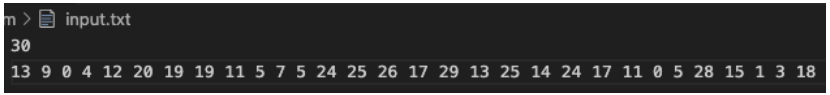
- **Merge-insertion sort (a modification of merge sort)**
  - Finally, merge the lists **using the standard merging mechanism**



Assignment #1: Description (Cont.)

❑ Input file

- The first line has an integer value which represents the length of the unordered list
- The second line has the unordered list that you will sort

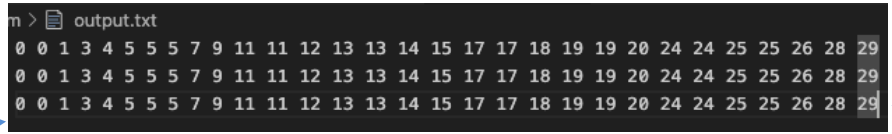


Example of Input.txt

❑ Output file

- The first line is the result of Insertion sort
- The second line is the result of Merge sort
- The last line is the result of Merge-insertion sort

No newline at the end of the output file



No space at the end of each line

Example of output.txt

Assignment #1: Makefile (C/C++)

❑ Writing a MakeFile and Test

```
all: compile run

compile: sorting.c
    gcc sorting.c -o sorting

run: sorting
    ./sorting input.txt output.txt

clean: sorting
    rm sorting
```

Makefile

'make' command performs 'compile' and 'run'

```
dyk@DavidMacBook hw % make
gcc sorting.c -o sorting
./sorting input.txt output.txt
insertion sorting done
merge sorting done
merge-insertion sorting done
dyk@DavidMacBook hw % ls
Makefile        input.txt       output.txt      sorting         sorting.c
dyk@DavidMacBook hw % cat output.txt
0 0 1 3 4 5 5 5 7 9 11 11 12 13 13 14 15 17 17 18 19 19 20 24 24 25 25 26 28 29
0 0 1 3 4 5 5 5 7 9 11 11 12 13 13 14 15 17 17 18 19 19 20 24 24 25 25 26 28 29
0 0 1 3 4 5 5 5 7 9 11 11 12 13 13 14 15 17 17 18 19 19 20 24 24 25 25 26 28 29
```

After executing your code

"output.txt" that includes the results of your code will be generated

Assignment #1: Makefile (Python)

❑ Writing a MakeFile and Test

```
run: sorting.py
    python3 sorting.py input.txt output.txt
```

Makefile

'make' command performs 'run' (python3 sorting.py input.txt output.txt)

```
dyk@DavidMacBook hypy % make
python3 sorting.py input.txt output.txt
insertion sorting done

merge sorting done

merge-insertion sorting done

dyk@DavidMacBook hypy % ls
Makefile        input.txt       output.txt      sorting.py
dyk@DavidMacBook hypy % cat output.txt
0 0 1 3 4 5 5 5 7 9 11 11 12 13 13 14 15 17 17 18 19 19 20 24 24 25 25 26 28 29
0 0 1 3 4 5 5 5 7 9 11 11 12 13 13 14 15 17 17 18 19 19 20 24 24 25 25 26 28 29
0 0 1 3 4 5 5 5 7 9 11 11 12 13 13 14 15 17 17 18 19 19 20 24 24 25 25 26 28 29
```

After executing your code

"output.txt" that includes the results of your code will be generated

Assignment #1: Submission Guideline

❑ Submission

- What: **Compressed file (Source code, makefile)**
  - Compressed filename: Assignment1_studentID
- Where: **Assignment board on eCampus**
- Deadline: **11:59 PM, October 3rd (Thursday), 2024**
  - +1 day (75%), +2 days (50%), +3 days (**Not accepted**)

❑ Cautions

- Make sure it compiles and runs properly!!
  - **The size of evaluation input file will vary,** so make sure your code can handle it
  - If it fails to compile or run during evaluation, it will be considered **as non-functional**
- Comments for all functions/algorithms that you implemented

Q&A:
Assignment #1