Introduction to Algorithms
알고리즘개론
2018 Spring Semester

Jinkyu Lee

Dept. of Computer Science and Engineering (and Software),
Sungkyunkwan University (SKKU)

Rules for all homework

- You should follow instructions.
  - Complier
    - You will get no point if your program cannot be complied with the specified complier
  - Input/output format
    - You will get no point if TA's automatic evaluation program cannot parse your input or output.
  - Permitted modification scope
    - You will get no point if you modify code outside of the permitted modification scope
  - All other rules
    - You will get severe penalty or no point if you violate the given rules.

Complier and input/output rules for all homework

- Every implementation homework will be evaluated by TA's automatic evaluation program with the following complier.
  - Complier: GCC 6.3
  - You will get no point if your program cannot be complied with GCC 6.3.
  - You can use standard library such as *stdlib.h* and *math.h*.
- Input/output format
  - You will get no point if TA's automatic evaluation program cannot parse your input or output according to the following rules.
  - Use stdin and stdout
  -
    -
  -
    -
- Recommended development environment (Windows)
  - IDE: CodeBlocks (http://www.codeblocks.org/downloads/26)
  - Compiler: MinGW (https://sourceforge.net/projects/mingw)
  - You can use the corresponding compliers for Linux and Mac.

Homework 2

- 7.5 points (7.5%)
  - 2A: 1.5 points (1.5%)
  - 2B: 3.5 points (3.5%)
  - 2C: 2.5 points (2.5%)
- Due data: 2018/4/16 Monday 23:59
  - Delay penalty: 1% per hour
  - Delay and evaluation will be applied to each file.
  - TA will only evaluate the latest version of your homework with time stamp.
  - Your time management is very important!
- Submission to iCampus
- TA: Jun Seong Lee
  - acu.pe.kr@gmail.com

Homework 2

- 2A
  - No file submission
- 2B
  - Code: Yourid_HW2B.c
    - The file type should be c, not cpp.
    - The file should be a single file.
    - Submit to “Homework 2B – Code”
  - Report: Yourid_HW2B.hwp
    - The file type can be hwp, doc(x) or pdf, not others
    - Submit to “Homework 2B – Report”
- 2C
  - Code: Yourid_HW2C.c
    - The file type should be c, not cpp.
    - The file should be a single file.
    - Submit to “Homework 2C – Code”
  - Report: Yourid_HW2C.hwp
    - The file type can be hwp, doc(x) or pdf, not others
    - Submit to “Homework 2C – Report”

Homework 2A

- 1.5 points (1.5%)

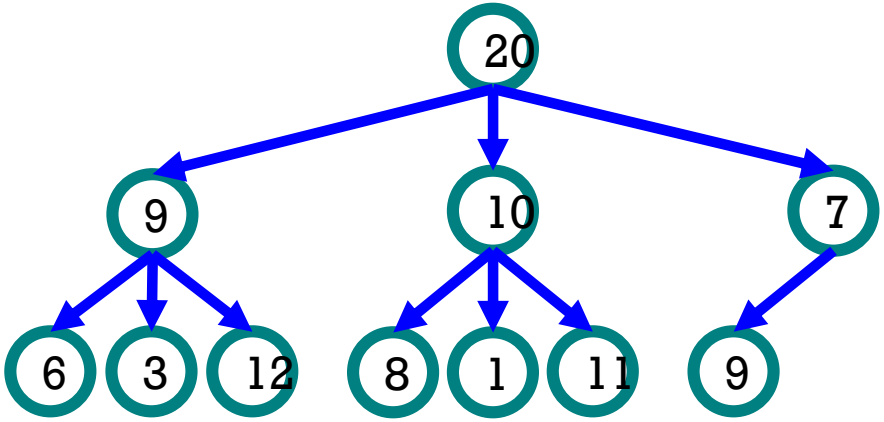
- You will have a in-class quiz in 4/4 (Wed) or 4/9 (Mon).
  - The coverage is all contents in Lecture Note 05 and 06.
  - If you have any reasonable possibility to be absent in those days, please tell me as soon as possible.
  - You will get no point if you miss the quiz.

Homework 2B

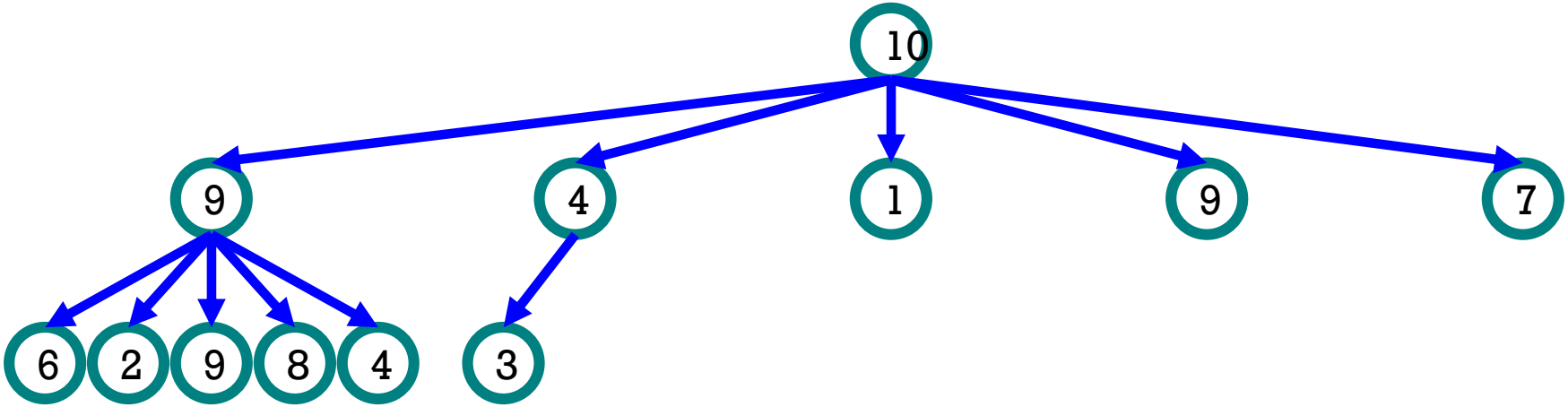
- Implement a heap sort with some modification as follows:
  - Each node's value is a natural number between 1 and 10000.
  - A complete X-nary tree is used.
    - For a complete X-nary tree with height h, all nodes at levels h-2 or less have X children nodes, and all leaves on level h are as far left as possible.
    - X can be 2, 3, 4 or 5. (i.e., binary, ternary, tetranary, or quinary)
  - Value at each node is smaller than or equal to values in subtrees.
  - Follow (but modify) the Max-Heapify, Build-Max-Heap and HeapSort algorithms in Lecture Note 05.

Homework 2B

- Example 1: a complete 3-nary tree (a complete ternary tree)



- Example 2: a complete 5-nary tree (a complete quinary tree)



Homework 2B

- Input
  - In the first line, there is only one number, which is X (2, 3, 4 or 5).
  - In the second line, each node's value is written, from the root to the leaves, and from the leftmost to the rightmost.
    - Each node's value is separated by a space.
  - Each line is separated by a new-line character.
  - Max input length: 200 bytes
  - Follow the sample input

Homework 2B

- Output
  - Print a min-heap with a complete X-nary tree, from the original one in one line.
  - And then,
  - Print every internal result in one line, after deleting the root node (which is equivalent to heap sort).
    - Each line is separated by a new-line character.
    - Every internal result should be the min-heap with a complete X-nary tree.
  - (So, the number of lines to be printed is equal to the number of nodes.)
  - Follow the sample output

Homework 2B

- Output
  - Sample input for Example 1

```
3
20 9 10 7 6 3 12 8 1 11 9
```

  - Sample output for Example 1

```
1 3 8 7 6 9 12 20 10 11 9
3 6 8 7 9 9 12 20 10 11 1
6 9 8 7 11 9 12 20 10 3 1
7 9 8 10 11 9 12 20 6 3 1
8 9 20 10 11 9 12 7 6 3 1
9 9 20 10 11 12 8 7 6 3 1
9 11 20 10 12 9 8 7 6 3 1
10 11 20 12 9 9 8 7 6 3 1
11 12 20 10 9 9 8 7 6 3 1
12 20 11 10 9 9 8 7 6 3 1
20 12 11 10 9 9 8 7 6 3 1
```

Homework 2B

- Output
  - Sample input for Example 2

```
5
10 9 4 1 9 7 6 2 9 8 4 3
```

  - Sample output for Example 2

```
1 2 3 10 9 7 6 9 9 8 4 4
2 4 3 10 9 7 6 9 9 8 4 1
3 4 4 10 9 7 6 9 9 8 2 1
4 6 4 10 9 7 8 9 9 3 2 1
4 6 9 10 9 7 8 9 4 3 2 1
6 8 9 10 9 7 9 4 4 3 2 1
7 8 9 10 9 9 6 4 4 3 2 1
8 9 9 10 9 7 6 4 4 3 2 1
9 9 9 10 8 7 6 4 4 3 2 1
9 10 9 9 8 7 6 4 4 3 2 1
9 10 9 9 8 7 6 4 4 3 2 1
10 9 9 9 8 7 6 4 4 3 2 1
```

Homework 2B

- Total score: 3.5 points (3.5%)
- Performance evaluation (3.0 points)
  - TA will test several cases.
  - For each case, the result should be printed within 10 seconds.
  - Your C code is tested with the following complier.
    - GCC 6.3
    - You will get zero point if your program cannot be complied with GCC 6.3.
  - You should follow the input and output format.
    - You will get zero point if the TA's automatic evaluation program cannot parse your input or output.

Homework 2B

- Report evaluation (0.4 points)
  - Explain your code using an example
  - No more than 2 pages
  - In English or Korean
- Code readability (and rules) evaluation (0.1 points)
  - Indent properly
  - Use meaningful names of variables
  - Write sufficient comments in English
    - Do not include any other natural language than English in you code.
  - Use correct file names

Homework 2C

- Implement a radix sort with some modification as follows:
  - All input and output numbers are ten-digit hexadecimal numbers, and all numbers are exactly ten-digit, i.e., 0000000000, 0000000001, 0000000002, 0000000003, 0000000004, 0000000005, 0000000006, 0000000007, 0000000008, 0000000009, 000000000A, 000000000B, 000000000C, 000000000D, 000000000E, 000000000F, 0000000010, …. , FFFFFFFFFF.
    - So, if a number is not four-digit or includes any other characters than 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E and F, you violate the input/output format (no point).
      - Each alphabet letter should be an upper-case letter, i.e., A, B, C, D, E and F, not a, b, c, d, e and f.

Homework 2C

- Input
  - Elements to be sorted, each separated by a space
  - Max input length: 200 bytes
  - Follow the sample input
- Output
  - You should write every the internal result in one line
    - In the n-th line, you apply radix sort in an ascending order for n digit.
    - Each line is separated by a new-line character.
    - Even though there is no need to sort, your result should be exactly ten lines.
    - The number of total lines should be 10.
  - Your radix sorting algorithm should be stable.
  - Follow the sample output

Homework 2C

- Sample input

  2004006457 C006000329 700C000657 400200083F A008000436 F005000720 D00B000355

- Sample output

  F005000720 D00B000355 A008000436 2004006457 700C000657 C006000329 400200083F
  F005000720 C006000329 A008000436 400200083F D00B000355 2004006457 700C000657
  C006000329 D00B000355 A008000436 2004006457 700C000657 F005000720 400200083F
  C006000329 D00B000355 A008000436 700C000657 F005000720 400200083F 2004006457
  C006000329 D00B000355 A008000436 700C000657 F005000720 400200083F 2004006457
  C006000329 D00B000355 A008000436 700C000657 F005000720 400200083F 2004006457
  400200083F 2004006457 F005000720 C006000329 A008000436 D00B000355 700C000657
  400200083F 2004006457 F005000720 C006000329 A008000436 D00B000355 700C000657
  400200083F 2004006457 F005000720 C006000329 A008000436 D00B000355 700C000657
  2004006457 400200083F 700C000657 A008000436 C006000329 D00B000355 F005000720

Homework 2C

- Total score: 2.5 points (2.5%)
- Performance evaluation (2.1 points)
  - TA will test several cases; TA will investigate your program to check whether insertion sort and merge sort are properly implemented.
    - If you implement other sorting algorithms than insertion sort and merge sort, the corresponding test case score will be zero.
  - For each case, the result should be printed within 10 seconds.
  - Your C code is tested with the following complier.
    - GCC 6.3
    - You will get zero point if your program cannot be complied with GCC 6.3.
  - You should follow the input and output format.
    - You will get zero point if the TA's automatic evaluation program cannot parse your input or output.

Homework 2C

- Report evaluation (0.3 points)
  - Explain your code using an example
  - No more than 2 pages
  - In English or Korean
- Code readability (and rules) evaluation (0.1 points)
  - Indent properly
  - Use meaningful names of variables
  - Write sufficient comments in English
    - Do not include any other natural language than English in you code.
  - Use correct file names