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.
- 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 1

- 7.5 points (7.5%)
  - 1A: 2.5 points (2.5%)
  - 1B: 3.5 points (3.5%)
  - 1C: 1.5 points (1.5%)
- Due data: 2018/3/26 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 1

- 1A
  - No file submission
- 1B
  - Code: Yourid_HW1B.c
    - The file type should be c, not cpp.
    - The file should be a single file.
    - Submit to “Homework 1B – Code”
  - Report: Yourid_HW1B.hwp
    - The file type can be hwp, doc(x) or pdf, not others
    - Submit to “Homework 1B – Report”
- 1C
  - Code: Yourid_HW1C.c
    - The file type should be c, not cpp.
    - The file should be a single file.
    - Submit to “Homework 1C – Code”
  - Report: Yourid_HW1C.hwp
    - The file type can be hwp, doc(x) or pdf, not others
    - Submit to “Homework 1C – Report”

Homework 1A

- 2.5 points (2.5%)

- You will have a in-class quiz in 3/26 (Mon), 3/28 (Wed) or 4/2 (Mon).
  - The coverage is all contents in Lecture Note 01, 02, 03 and 04.
  - 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 1B

- Implement a calculator using data structure as follows:
  - All input and output numbers are four-digit octal numbers, and all numbers are exactly four-digit, i.e., 0000, 0001, 0002, 0003, 0004, 0005, 0006, 0007, 0010, …. , 7777.
    - So, if a number is not four-digit or includes any other characters than 0, 1, 2, 3, 4, 5, 6 and 7, you violate the input/output format (no point).
  - Input of the calculator is postfix.
    - If you don't remember what postfix is, refer the supplementary file.
    - There are five operators: +, -, *, /, %, which mean the addition, subtraction, multiplication, division, modular operations, respectively.
    - For each internal calculation, the internal result has only the last four-digit numbers.
      - Example: 5000 5000 + 0002 / = (5000 5000 +) 0002 / = 2000 0002 / = 1000
    - If you apply the division operation, you apply the truncation division.
      - Example: 0013 0003 / = 0003
    - Use stack

Homework 1B

- Input
  - Each number or operator is separated by a space.
  - Max input length: 200 bytes
  - Follow the sample input.
- Output
  - A four-digit octal number
  - Follow the sample output
- Sample input & output 1

```
1002 0007 * 1000 0040 / +
7036
```

- Sample input & output 2

```
0013 0003 /
0003
```

- Sample input & output 3

```
1002 0010 * 2000 /
0000
```

Homework 1B

- 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 1B

- 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 1C

- Exercise insertion sort and merge sort (EASY)
  - All input and output numbers are four-digit hexadecimal numbers, and all numbers are exactly four-digit, i.e., 0000, 0001, 0002, 0003, 0004, 0005, 0006, 0007, 0008, 0009, 000A, 000B, 000C, 000D, 000E, 000F, 0010, …. , FFFF.
    - 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 1C

- Input
  - 1 or 2 (1 means insertion sort, and 2 means merge sort)
  - Elements to be sorted, each separated by a space
  - Max input length: 200 bytes
  - Follow the sample input
- Output
  - Elements sorted in a descending order, each separated by a space
  - Follow the sample output

- Sample input & output 1

```
1 1000 2000 3000 4000 F000
F000 4000 3000 2000 1000
```

- Sample input & output 2

```
2 1000 2000 3000 4000 F000
F000 4000 3000 2000 1000
```

Homework 1C

- Total score: 1.5 points (1.5%)
- Performance evaluation (1.2 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 1C

- Report evaluation (0.2 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