Lab 2: Review on C Programming

Problem 1.

In this assignment, you will practice programming by using recursive algorithm. In addition, you need to know the fundamental file I/O, and how to use command line arguments.
Use a command line argument with designated input and output file names. Read a set of non-redundant characters that are separated by comma (,) and enclosed by parentheses ({ }) in the input file. Create a pointer of characters to save the elements given. Then print out all permutation of the given set in the output file as follows.

1. An example of command line

   >p2_1 input.txt output.txt

2. An example of input (input.txt)

   {a,b,c}

3. Corresponding output (output.txt)

   {a,b,c}
   {a,c,b}
   {b,a,c}
   {b,c,a}
   {c,b,a}
   {c,a,b}

4. Conditions :
   - The element is one character.
   - The maximum number of elements in a set is 100.
   - You should use recursive algorithm.

Problem 2.

In this problem, you will practice programming by using pointer and structure. Read the first integer from the input file to figure out the number of students. From the second line, the student name and three scores (for literature, math, and science) are provided in each line. The name and scores are separated by a space. Create an array of structures, and put the information in it. Then print the structures in the output file.

1. An example of command line
   >p2_2 input.txt output.txt

2. An example of Input (input.txt)

```
3
Sarah 96 90 80
Minsu 55 70 76
Nara 88 70 96
```

Name	Sarah	Minsu	Nara
Literature	96	55	88
Math	90	70	70
Science	80	76	96

3. Corresponding output (output.txt)

4. Data structure :

```
typedef struct student{
        char *name;
        int literature;
        int math;
        int science;
} student;
```

5. Conditions
   - The length of the student name should be up to 30 characters.
   - No blank space is allowed in the name.
   - Student name and scores are separated by a space in the input file

*** Whenever you allocate memory, you must free it afterward.

Problem 3.

In this problem, you will practice programming by using array of structure. Read an integer in the first line of the input file to figure out the number of students in the course. From the second line, the student name, assignment scores, and exam scores are provided in order in each line. Create two structures for STUDENT and COURSE. Then print the average scores for the assignments and the exams as the final score. In addition, print the average score of the assignment and the exam for the course.

1. An example of command line

```
>p2_3 input.txt output.txt
```

2. An example of Input (input.txt)

```
3
Minsu
50 43 48 20 35 39
50 39
Suji
50 49 45 50 48 50
45 45
Jisu
50 39 40 48 50 35
50 45
```

3. Corresponding output (output.txt)

```
Minsu
HwAvg 39.17
ExamAvg 44.50
Suji
HwAvg 48.67
ExamAvg 45.00
Jisu
HwAvg 43.67
ExamAvg 47.50

Course
HwAvg 43.83
ExamAvg 45.67
```

4. Data structure that you should use

```
#define NUM_HW 6
#define NUM_EXAM 2
#define MAX_ENROLL 5

    typedef struct{
        studentT students[MAX_ENROLL];
        int numEnrolled; //the number of students
    } courseT;
```

```
typedef struct {
        string name;
        int hw [NUM_HW];
        int exams [NUM_EXAM];
        double hwAvg;
        double examAvg;
} studentT;
```

5. Conditions :

- The maximum number of students is 5.
- The length of the student name should be up to 30 characters.
- No blank space is allowed in the name.
- Scores are separated by space in the input file.
- Round off the numbers to three decimal places.

** Whenever you allocate memory, you must free it afterward.