Introduction to Data Structures

Jinkyu Lee

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

Homework 4A

- 40 points for coding evaluation
  - Submission format
    - File name: yourid_HW4A.c
      - Example: 2000123456_HW4A.c
    - File type: .c (NOT .cpp)
  - Submission site: https://icampus.skku.edu
    - Week 11: [Homework] 4A (code)

- 1 point for report
  - The report is not evaluated in detail but evaluated as Pass/Fail
  - Template: Homework Report Template.docx
  - Submission format
    - File name: yourid_HW4A.pdf
      - Example: 2000123456_HW4A.pdf
    - File type: .pdf
  - Submission site: https://icampus.skku.edu
    - Week 11: [Homework] 4A (report)

- Due date
  - 11/24 23:59 (no late submission accepted)

Rules for homework

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

Complier and input/output rules for homework

- Every implementation homework will be evaluated by TA's automatic evaluation program with the following complier.
  - Complier: GCC 7.X, 8.X, 9.X or 10.X
    - https://gcc.gnu.org/
  - You will get no/less point if your program cannot be complied with GCC 7.X, 8.X, 9.X or 10.X.
    - For example, do not rely on visual studio.
  - You can use standard library such as *stdlib.h* and *math.h*.
- Input/output format
  - You will get no/less point if TA's automatic evaluation program cannot parse your input or output according to the following rules.
  - Use `stdin` and `stdout`

Problem

- Problem: Heap implementation for 3-digit hexadecimal numbers.
  - A 3-digit Hexadecimal number is expressed as 000, 001, 002, … 009, 00A, 00B, 00C, 00D, 00E, 00F, …, FF0, FF1, FF2, FF3, FF4, FF5, FF6, FF7, FF8, FF9, FFA, FFB, FFC, FFD, FFE or FFF.
    - Be careful that each character is either 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E or F. (i.e., decimal numbers and upper-case letters A-F).
  - Sort data in an ascending order using heap.
  - Construct Min Heap (NOT Max Heap)

Input/Ouput

- Input

```
5
0E0
321
EEE
CCC
3D0
```

The number of elements in the heap

- Output

```
0E0
321
3D0
CCC
EEE
```

Template (4A_template.c)

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
/* Modify from here */


/* Modify to here */

#define MAX_HEAP 100

typedef enum { false, true } bool;

typedef struct {
      char small;
      char middle;
      char large;
} Hexa_num;

typedef struct {
    Hexa_num data;
    // This is a priority as well as data
} PNode;

typedef struct {
      PNode items[MAX_HEAP + 1];
      int num;
} Heap;

void InitHeap(Heap *pheap);

bool IsEmpty(Heap *pheap);

bool IsFull(Heap *pheap);

void Insert(Heap *pheap, Hexa_num data);

Hexa_num Delete(Heap *pheap);

void HeapSort(Hexa_num a[], int n);

Hexa_num *CreateHexaNum(int n);

void GetInput();

/* Modify from here */


/* Modify to here */

int main() {

      GetInput();
      /*
      5
      0E0
      321
      EEE
      CCC
      3D0
      */

      return 0;
}
```

```
void HeapSort(Hexa_num a[], int n) {
      Heap heap;
      InitHeap(&heap);
      // Insert all elements to the h
      for (int i = 0; i < n; i++)
            Insert(&heap, a[i]);

      // Remove all elements from the
      for (int i = 0; i <= n-1; i++)
            a[i] = Delete(&heap);

      for (int i = 0; i < n; i++)
            printf("%c%c%c\n", a[i].l
a[i].small);
}

Hexa_num *CreateHexaNum(int n){
    char a[4];
      Hexa_num *res = (Hexa_num*)mall
      for (int i = 0; i < n; i++) {
        scanf("%s", &a);
        res[i].large = a[0];
        res[i].middle = a[1];
        res[i].small = a[2];
      }
      return res;
}

void GetInput() {
    int n;
    Hexa_num *data;
    scanf("%d", &n);
    data = CreateHexaNum(n);
      HeapSort(data, n);
}

/* Modify from here */


/* Modify to here */
```

Template

- You cannot modify the template except the space between /*Modify from here*/ and /*Modify to here*/
  - Do not remove /*Modify from here*/ and /*Modify to here*/
  - TA will copy the space and evaluate your code.
  - You may add user-defined functions and header files between /*Modify from here*/ and /*Modify to here*/.

  - In the space, you need to implement the following functions. (Next page)
    - void InitHeap(Heap *pheap);
    - bool IsEmpty(Heap *pheap);
    - bool IsFull(Heap *pheap);
    - void Insert(Heap *pheap, Hexa_num data);
    - Hexa_num Delete(Heap *pheap);

Evaluation

- Evaluation
  - TA will test several cases.
  - Read Pages 7~8 (regarding template) carefully.
  - For each test case,
    - If your C code results in an answer within 10 seconds on a platform with average computing power AND, you solved the problem by constructing Min Heap,
      - If your output is perfect for all numbers,
        - You get 100%.
      - Else,
        - You get 0%.
    - Else,
      - You get 0%.