[2주 과제]

**[알고리즘}**

363~365 p 의 UsefulHeap.c 소스를 타이핑한 후 한 줄 단위로 하는 일을 comment 붙여 제출하시오.

**[C++]**

아래 소스를 C++ 스타일로 다음과 같이 수정하시오.

1. Heap 타입의 heapArr 멤버는 배열 대신 메모리 할당 받아 사용하고자 한다. HeapInit 함수에서 메모리 할당 받고 main 함수에서 반환 직전에 메모리 해제하시오.
2. HInsert, HDelete 함수의 while문 안에서 적절히 reference 타입의 변수를 사용하여 가독성을 높이시오
3. 각 함수들이 Heap \* 를 인자로 받아 코드가 복잡해 보인다. Heap & 타입을 인자로 받도록 수정하시오.
4. HIsEmpty 함수의 반환 타입은 bool 로 수정하시오.

#include <stdio.h>

#define TRUE 1

#define FALSE 0

#define HEAP\_LEN 100

typedef char HData;

typedef struct {

 int pri;

 HData data;

}HeapElem;

typedef struct {

 int numOfData;

 HeapElem heapArr[HEAP\_LEN];

}Heap;

void HeapInit(Heap\* ph) {

 ph->numOfData = 0;

}

void HInsert(Heap\* ph, HData data, int pri) {

 HeapElem tmp;

 int idx, idxParent;

 ph->heapArr[ph->numOfData + 1].data = data;

 ph->heapArr[ph->numOfData + 1].pri = pri;

 ph->numOfData++;

 idx = ph->numOfData;

 while (idx > 1) {

 idxParent = idx / 2;

 if (ph->heapArr[idx].pri < ph->heapArr[idxParent].pri)

 {

 tmp = ph->heapArr[idx];

 ph->heapArr[idx] = ph->heapArr[idxParent];

 ph->heapArr[idxParent] = tmp;

 idx = idxParent;

 }

 else break;

 }

}

int HIsEmpty(Heap\* ph) {

 if (ph->numOfData == 0)

 return TRUE;

 else

 return FALSE;

}

int GetPriChildIDX(Heap\* ph, int idx) {

 int leftIndex = idx \* 2;

 int rightIndex = idx \* 2 + 1;

 if (leftIndex > ph->numOfData) //자식없음

 return 0;

 if (rightIndex > ph->numOfData) //왼쪽만 있음

 return leftIndex;

 if (ph->heapArr[leftIndex].pri < ph->heapArr[rightIndex].pri)

 return leftIndex;

 else

 return rightIndex;

}

HData HDelete(Heap\* ph) {

 int idx, idxChild;

 HeapElem tmp;

 HData ret = ph->heapArr[1].data;

 ph->heapArr[1] = ph->heapArr[ph->numOfData];

 ph->numOfData--;

 idx = 1;

 idxChild = GetPriChildIDX(ph, idx);

 while (idxChild != 0) {

 if (ph->heapArr[idx].pri > ph->heapArr[idxChild].pri) //idx번째의 pr < idxChild번째의 pr

 {

 tmp = ph->heapArr[idx];

 ph->heapArr[idx] = ph->heapArr[idxChild];

 ph->heapArr[idxChild] = tmp;

 idx = idxChild;

 idxChild = GetPriChildIDX(ph, idx);

 }

 else

 break;

 }

 return ret;

}

int main() {

 Heap heap;

 HeapInit(&heap);

 HInsert(&heap, 'A', 1);

 HInsert(&heap, 'B', 2);

 HInsert(&heap, 'C', 3);

 printf("%c \n", HDelete(&heap));

 HInsert(&heap, 'A', 1);

 HInsert(&heap, 'B', 2);

 HInsert(&heap, 'C', 3);

 printf("%c \n", HDelete(&heap));

 while (!HIsEmpty(&heap))

 {

 printf("%c \n", HDelete(&heap));

 }

 return 0;

}