UNIMORE
UNIVERSITÀ DEGLI STUDI DI
MODENA E REGGIO EMILIA

Dispense del Corso di Laboratorio di Fondamenti di Informatica II e Lab

Federico Bolelli, Silvia Cascianelli

Esercitazione 10: Heap

Ultimo aggiornamento: 04/06/2020

Introduzione

- Siano date le seguenti definizioni:

```c
typedef int ElemType;

struct Heap{
    ElemType *data;
    size_t size;
};
typedef struct Heap Heap;
```

Introduzione

- Siano date le implementazioni delle seguenti funzioni specifiche per la creazione, eliminazione, confronto, acquisizione da file e scrittura su file di `ElemType` di tipo `int`:

```
- int ElemCompare(const ElemType *e1, const ElemType *e2);
- ElemType ElemCopy(const ElemType *e);
- void ElemDelete(ElemType *e);
- void ElemSwap(ElemType *e1, ElemType *e2);
- int ReadElem(FILE *f, ElemType *e);
- int ReadStdinElem(ElemType *e);
- void WriteElem(const ElemType *e, FILE *f);
- void WriteStdoutElem(const ElemType *e);
```

Introduzione

- Siano inoltre date le implementazioni delle seguenti funzioni primitive (e non) specifiche per *(min-)heap*:

```
- int LeftHeap(int i);
- int RightHeap(int i);
- int ParentHeap(int i);
- Heap* CreateEmptyHeap();
- void InsertNodeMinHeap(Heap *h, const ElemType *e);
- bool IsEmptyHeap(const Heap *h);
- ElemType* GetNodeValueHeap(const Heap *h, int i);
- void MoveUpMinHeap(Heap *h, int i);
- void MoveDownMinHeap(Heap *h, int i);
- void DeleteHeap(Heap *h);
- void WriteHeap(const Heap *h, FILE *f);
- void WriteStdoutHeap(const Heap *i);
```

Introduzione

- Trovate le dichiarazione e le definizione dei tipi di dato e delle funzioni descritte nelle slide precedenti nel repository GitHub al link: https://github.com/prittt/Fondamenti-II
- Leggete attentamente il README, il quale vi spiega come scaricare lo zip contenente i file `minheap_int.h` e `minheap _int.c` con le implementazione che vi serviranno per l'esercitazione, e dove trovare la documentazione delle suddette funzioni.
- Consultate la documentazione prima di utilizzare una funzione di cui non conoscete con esattezza il comportamento.
- Attenzione: non dovete implementare voi le funzioni primitive, ma dovete utilizzare quelle che vi vengono fornite.

HEAP: Heapify

- Esercizio 1 (`HeapifyMinHeap`):

Nel file `heapify.c` si implementi la definizione della seguente funzione:

```
Heap* HeapifyMinHeap(const ElemType *v, size_t v_size)
```

La funzione `HeapifyMinHeap` prende in input un vettore di `ElemType` e la sua dimensione e deve creare dinamicamente un (*min-)heap* contenente tutti gli elementi del vettore. La funzione deve quindi ritornare il puntatore al *(min-)heap* appena creato. Se il vettore di input è vuoto, la funzione dovrà ritornare uno heap vuoto.

Si testi la funzione con un opportuno `main()` di prova.

HEAP: Primitive Ricorsive

- Esercizio 2 (`MoveUpMinHeapRec`):

Nel file `move_up.c` si implementi la definizione della seguente funzione:

```
void MoveUpMinHeapRec(Heap *h, int i)
```

La funzione prende in input un *(min-)heap* e l'indice di un nodo che viola le proprietà heap. La funzione deve spostare il nodo verso l'alto, ovvero scambiarlo con il padre, fino a quando le proprietà *min-heap* non sono rispettate. A differenza della primitiva `MoveUpMinHeap`, la funzione deve essere ricorsiva.

Si testi la funzione con un opportuno `main()` di prova.

HEAP: Primitive Ricorsive

- Esercizio 3 (MoveDownMinHeapRec):

Nel file `move_down.c` si implementi la definizione della seguente funzione:

```
void MoveDownMinHeapRec(Heap *h, int i);
```

La funzione prende in input un *(min-)heap* e l'indice di un nodo che viola le proprietà heap. La funzione deve spostare il nodo verso il basso, ovvero scambiarlo con il figlio minore, fino a quando le proprietà *min-heap* non sono rispettate. A differenza della primitiva `MoveDownMinHeap`, la funzione deve essere ricorsiva.

Si testi la funzione con un opportuno `main()` di prova.

HEAP: Pop

- Esercizio 4 (Pop):

Nel file `pop.c` si implementi la definizione della seguente funzione:

```
bool PopMinHeap(Heap *h, ElemType *e);
```

La funzione `PopMinHeap` prende in input un *(min-)heap* ed estrae l'elemento minimo dallo heap, deallocando opportunamente la memoria e assicurandosi che le proprietà *(min-)heap* siano rispettate al termine dell'operazione.

La funzione deve restituire 'false' se lo heap è vuoto, 'true' altrimenti e salvare in `e` l'elemento estratto dallo heap.

Si testi la funzione con un opportuno `main()` di prova.

HEAP: Heapsort

- Esercizio 5 (Heapsort):

Nel file `heapsort.c` si implementi la definizione della seguente funzione:

```
void HeapsortMinHeap(Heap *h);
```

La funzione `HeapsortMinHeap` prende in input un *(min-)heap* e lo trasforma in modo tale che al termine dell'esecuzione l'array dei dati sia ordinato in maniera decrescente. N.B. Non bisogna usare algoritmi di ordinamento, ma sfruttare le proprietà dello heap.

Si testi la funzione con un opportuno `main()` di prova. Al termine dell'esecuzione lo heap rispetta ancora le proprietà?

HEAP: Max-Heap

- Esercizio 6 (MaxHeap):

Si modifichino le primitive fornite per *(min-)heap* di interi in modo tale che diventino primitive per *(max-)heap* di interi. Si risolvano quindi gli esercizi precedenti facendo riferimento al (max-)heap. In questo caso la funzione `Pop` dovrà estrarre l'elemento massimo dalla heap, l'`Heapify` dovrà costruire una *(max-)heap*, e la `Heapsort` dovrà ordinare in ordine crescente.

Modalità di Consegna

- Per questa esercitazione dovrete consegnare gli esercizi 1, 2, 4 e 5 utilizzando il **nuovo** sistema di sottomissione online.
- **Collegatevi al sito https://olj.ing.unimore.it/ e fate il login (in alto a destra) utilizzando le vostre credenziali shibboleth di UNIMORE.**
- Selezionate *Esercitazione Heap,* aprite il link dell'esercizio di cui volete fare la sottomissione e incollate il codice nei box relativi ai rispettivi file. **Non dovete caricare il `main()`.**
- Assicuratevi tuttavia di scrivere il `main()` in Visual Studio per verificare se quello che avete fatto funziona, prima di caricare la soluzione!
- Quando necessario, il codice che sottomettete dovrà opportunamente includere il file delle primitive (`#include "minheap_int.h"`).
- **Non caricate sul sistema l'implementazione delle primitive.**
- **Attenzione: il sistema è ancora in fase di sviluppo quindi vi chiediamo di segnalare a federico.bolelli@unimore.it eventuali errori o incongruenze riscontrate.**