DM507 Algoritmer og datastrukturer

Forår 2017

Projekt, del I

Institut for matematik og datalogi
Syddansk Universitet

23. februar, 2017

Dette projekt udleveres i tre dele. Hver del har sin deadline, således at arbejdet strækkes over hele semesteret. Deadline for del I er tirsdag den 14. marts. De tre dele I/II/III er ikke lige store, men har omfang omtrent fordelt i forholdet 15/35/50. Projektet skal besvares i grupper af størrelse to (evt. tre, efter aftale med underviser).

Mål

Målet for del I af projektet er at implementere datastrukturen *prioritetskø*, og derefter testen den ved at bruge den til at sortere tal. Prioritetskøen vil blive brugt igen senere i del III af projektet.

Det overordnede mål for hele projektet er træning i at overføre kursets viden om algoritmer og datastrukturer til konkret programmering. Projektet og den skriftlige eksamen komplementerer derfor hinanden, og projektet er ikke alene ment som en forberedelse til den skriftlige eksamen.

Opgave

Du skal i Java implementere en datastruktur, som tilbyder (`implements` i Java termininologi) følgende interface (som udleveres):

```
public interface PQ {
    public Element extractMin();
    public void insert(Element e);
}
```

Metoden `extractMin()` skal returnere det element i prioritetskøen som har mindst prioritet. Det må gerne antages at metoden kun kaldes på en ikke-tom

prioritetskø. Metoden `insert(e)` skal indsætte elementet `e` i prioritetskøen. Det må gerne antages at metoden kun kaldes på en prioritetskø med plads til endnu et element. Det overlades til brugeren af prioritetskøen at sikre at ovenstående antagelser er opfyldt under brug, f.eks. ved at holde styr på antal elementer i prioritetskøen.

Implementationen skal være i form af en Java-klasse, som kan bruges af andre programmer. Klassen skal hedde `PQHeap`, og skal implementere ovenstående interface `PQ`. Klassen skal have én constructor-metode `PQHeap(int maxElms)`, som returnerer en ny, tom prioritetskø. Argumentet `maxElms` angiver det maksimale antal elementer i køen. Der vil være behov for at implementere metoder udover dem i interfacet, til internt brug i klassen.

> Implementationen skal laves ved hjælp af strukturen *Heap* i et array (eller evt. en `ArrayList`) af `Element`s, og *skal* basere sig på pseudo-koden i Cormen et al. kapitel 6 på siderne 163, 154, 152 og 164 (dog med de simplificeringer som er angivet nedenfor).

Ovenfor er `Element` en type, der implementerer et (nøgle,data)-par. Denne type er defineret ved følgende klasse (som udleveres):

```
public class Element {
    public int key;
    public Object data;
    public Element(int i, Object o){
        this.key = i;
        this.data = o;
    }
}
```

De to dele af et objekt `e` af typen `Element` skal blot tilgås som `e.key` og `e.data`. Vi vil omtale `e.key` som elementets prioritet. Elementers prioriteter er altså af typen `int`, og deres associerede data er af typen `Object`.

Bemærk følgende detaljer:

1. Du skal i dette projekt lave en *min*-heap struktur, mens bogen formulerer sin pseudo-kode for en max-heap struktur. Pseudo-koden skal derfor have alle uligheder vendt.

2. Bogens pseudo-kode indekserer arrays startende med 1, mens Java starter med 0. En simpel måde at anvende bogens pseudo-kode på i Java, er at lægge én til den ønskede længde på array'et, og så ikke at bruge pladsen med index 0 til noget.

3. Følgende simplificering af bogens pseudo-kode kan laves: på side 163 kan første **if** udelades pga. antagelsen om at prioritetskøen ikke er tom, og på side 164 kan de to stykker pseudo-kode på siden bygges sammen til én (pga. at vi ikke skal lave en eksplicit `increaseKey()`/`decreaseKey()`), hvorved man kan spare brugen af ∞ samt **if** i det første stykke pseudo-kode. Det anbefales at lave denne simplificering.

4. Parametrene i metoderne i interfacet `PQ` er ikke præcis de samme som i bogens pseudo-kode. Dette skyldes at i objektorienteret programmering kaldes metoder på et objekt `Q` via syntaksen `Q.metode()` fremfor `metode(Q)`, samt at bogen kun opererer med prioriteter og ikke elementer med ekstra data. Derudover er `A` i bogens pseudo-kode et array indeholdende heapen, som *ikke* bør kunne tilgås direkte af brugere af et prioritetskø-objekt `Q` på anden måde end gennem metoderne fra interfacet.

Test

Du skal naturligvis afprøve din klasse grundigt. Herunder skal du teste den med det udleverede program `Heapsort`, der sorterer ved at lave gentagne `insert`'s i en prioritetskø, efterfulgt af gentagne `extractMin`'s.[1] Du skal teste sortering af alle de udleverede datafiler.

Bemærk at programmet `Heapsort` læser fra standard input (der som default er tastaturet), og skriver til standard output (der som default er skærmen). Ved hjælp af redirection[2] af standard input og output kan man i en kommandoprompt anvende programmet (uden at ændre i det) på filer således:

```
java Heapsort  < inputfile  > outputfile
```

Det er vigtigt at I afprøver ovenstående i en kommandoprompt, da programmerne skal kunne testes automatisk efter aflevering. Man må af samme grund heller ikke i sin kildekode have `package` statements, eller organisere sin kode i en folderstruktur[3].

[1] Da programmet kun sorterer tal, sætter det data-delen af elementerne til at være `null`. Senere i del III af projektet skal data-delen faktisk bruges til noget.

[2] Læs evt. om redirection på Unix Power Tools eller Wikipedia.

[3] NB: Dette sker ofte automatisk hvis man bruger en IDE som Eclipse eller NetBeans under udvikling af koden. I så fald må man fjerne `package` statements og folderstruktur inden aflevering, (og derefter igen teste funktionaliteten, herunder redirection.)

Formalia

Du skal kun aflevere din Java source-fil `PQHeap.java`. Denne skal indeholde grundige kommentarer. Den skal også indeholde navnene og SDU-logins på gruppens medlemmer.

Filen skal afleveres elektronisk i Blackboard med værktøjet "SDU Assignment", som findes i menuen til venstre på kursussiden i Blackboard. Der behøves kun afleveres under én persons navn i gruppen.

Filen skal *også* afleveres udprintet på papir i instruktorens boks på Imada (vælg én af instruktorerne, hvis personerne i gruppen går på forskellige hold). Spørg Imadas sekretær hvis man ikke ved, hvor instruktorboksene er. Der skal blot afleveres én kopi per gruppe.

Aflever senest:

Tirsdag den 14. marts, 2017, kl. 12:00.

Bemærk at aflevering af andres kode eller tekst, hvad enten kopieret fra medstuderende, fra nettet, eller på andre måder, er eksamenssnyd, og vil blive behandlet særdeles alvorligt efter gældende SDU regler. Man lærer desuden heller ikke noget.