Algoritmer

Det minsta heltalet – hitta fel i lösningen

Ett problem att lösa: bestäm det minsta heltalet

Det finns en sekventiell samling med heltal. Bestäm det minsta heltalet i samlingen.

Lösningen

Följande lösning innehåller två fel, utplacerade i två olika rader. Det första felet gör att algoritmen blir oändlig. Det andra felet leder till ett felaktigt svar för vissa talsekvenser.

Felen i lösningen ska hittas och korrigeras. När ett fel korrigeras, ska bara en del av den motsvarande raden ändras.

En lösning till problemet – innehåller två fel

```
// min returnerar det minsta elementet i en sekventiell samling.
// Om samlingen är tom, kastas ett undantag av typen IllegalArgumentException.
public static int min (int[] element) throws IllegalArgumentException
{
    if (element.length == 0)
        throw new IllegalArgumentException ("tom samling");

    // hör ihop med spårutskriften 2:
    // int    antalVarv = 1;

    int[]    sekvens = element;
    int      antaletPar = sekvens.length / 2;
    int      antaletOparadeElement = sekvens.length % 2;
    int      antaletTankbaraElement = antaletPar + antaletOparadeElement;
    int[]    delsekvens = new int[antaletTankbaraElement];
    int      i = 0;
    int      j = 0;
    while (sekvens.length > 1)
    {
        // skilj ur en delsekvens med de tänkbara elementen
        i = 0;
        j = 0;
        while (j < antaletPar)
        {
            delsekvens[j++] = (sekvens[i] < sekvens[i + 1])? sekvens[i] : sekvens[i + 1];
            i += 2;
        }
        if (antaletOparadeElement == 1)
            delsekvens[j] = sekvens[sekvens.length - 1];

        // utgå nu ifrån delsekvensen
        sekvens = delsekvens;
        antaletPar = antaletTankbaraElement / 2;
        antaletOparadeElement = antaletTankbaraElement % 2;
        antaletTankbaraElement = antaletPar + antaletOparadeElement;

        // spårutskrift 1 – för att följa sekvensen
        // System.out.println (java.util.Arrays.toString (sekvens));

        // spårutskrift 2 - för att avsluta loopen i förväg
        // (för att kunna se vad som händer i början)
        // if (antalVarv++ == 10)
        //    System.exit (0);
    }

    // sekvens[0] är det enda återstående tänkbara elementet
    // - det är det minsta elementet
    return sekvens[0];
}
```

Uppgifter i samband med problemet och lösningen

1. . Spåra metoden `min` i fallet att sekvensen innehåller sexton element. Använd papper och penna: rita en serie bilder som visar hur talsekvensen transformeras. Analysera transformationer av talsekvensen, och hitta det första felet i metoden. Rätta detta fel.

2. För att hitta det andra felet spåra den korrigerade varianten av metoden `min`. Använd en sekvens som innehåller nitton element med det minsta elementet på den sjuttonde positionen. Rita en serie bilder som visar hur det minsta heltalet bestäms. Varför erhålls ett felaktigt svar? Hitta det andra felet i metoden och rätta det.

3. Spåra den variant av metoden `min` som innehåller två fel med datorns hjälp. Skapa ett testprogram som anropar metoden `min`. Använd en sekvens med sexton element. Aktivera först spårutskriften `1` och följ utvecklingen. Aktivera sedan även spårutskriften `2`. Analysera de uppgifter som erhålls och hitta det första felet. Korrigera detta fel.

4. Spåra med datorns hjälp den variant av metoden `min` som enbart innehåller det andra felet. Använd en sekvens som innehåller nitton element med det minsta elementet på den sjuttonde positionen. Man ska aktivera bara spårutskriften `1`. Analysera de data som erhålls och hitta felet. Korrigera felet.

5. Lös problemet på ett annat sätt: skapa en ny metod `min` som bestämmer det minsta elementet i samlingen. I denna metod ska inte urskiljningsstrategin användas, utan en minneseffektivare och enklare strategi: uppdateringsstrategin.