Quicksort ABC

Sorteringsrutin för ABC 80

Quicksort ABC

Sorteringsrutin för ABC 80

I många sammanhang är det väsentligt med en snabb rutin för att sortera textsträngar. Om antalet strängar är litet (några tiotal) kan problemet lösas med s k bubbelsortering, d v s man jämför (och byter) strängarna parvis till dess att allt är sorterat. Denna metod är enkel att programmera men mycket ineffektiv.

Föreliggande sorteringsrutin bygger på "quicksort"-metoden och är skriven i Z80 assembler. Den har många fördelar:

- Extremt snabb, troligtvis den snabbaste man kan få på Z80. Att sortera 1 000 strängar tar högst några sekunder.
- Korrekt sorteringsordning för svenska tecken, d v s é efter e, ü efter y samt åäö. Litet a följer direkt efter stort A etc. Sorteringsordningen bestäms av en tabell och kan förändras.
- Sorteringen är stabil, vilket innebär att om två strängar är lika, så bibehålles den ursprungliga ordningen mellan dem.
- Enkelt anrop av subrutinen där man inte är låst vid fasta variabelnamn utan kan sortera flera olika strängvektorer i samma program.
- Möjlighet att samtidigt sortera en heltalsvektor i samma ordning som strängvektorn. Detta är nödvändigt om man måste hålla reda på t ex skivminnesadresser eller om poster skall sorteras på ett nyckelbegrepp som ej ligger först i posten.
- Sorteringsrutinen upptar 1K byte.

Läs noga igenom försäljningsvillkoren. Dessa är placerade längst bak i bruksanvisningen.

1. # QSORT – SORTERINGSRUTIN FÖR ABC80

QSORT är en sorteringsrutin som skrivits i assembler och är mycket snabb.

Assemblerkoden placeras strax under DOSbuffertarna i ABC80 minnet och den kräver 1K byte. Inladdningen sker med hjälprutinen SORTINIT som sänker "taket", så att QSORT får plats, samt länkar till denna.

När QSORT laddats in kan man köra program som använder sorteringen. För att ta bort assemblerkoden trycker man på resetknappen till vänster på tangentbordets baksida.

Att anropa sorteringsrutinen från ett BASIC program är mycket enkelt:

```
K = CALL (-3703%)
```

Före detta anrop måste man emellertid på något sätt beskriva vilken strängvektor som ska sorteras (man vill eventuellt sortera flera olika strängvektorer i samma program). Antag att vi har deklarerat

```
DIM B¤ (200) = 25 , C¤ (20)
```

Sorteringsrutinen använder sig av en enda variabel som skall heta P9¤. Denna enkla sträng får beskriva vilken strängvektor som ska sorteras. Antag att vi vill sortera först B¤ och sedan C¤ enligt DIM-satsen ovan. Vi matar då in följande satser:

```
P9¤ = "B¤ " : K = CALL (-3703%)
P9¤ = "C¤ " : K = CALL (-3703%)
```

P9¤ ändras inte vid sorteringen, och behöver alltså bara sättas vid ett tillfälle om man sorterar samma strängvektor flera gånger.

Tillsammans med sorteringsrutinen levereras ett testprogram SORTDEMO som visar denna enkla tillämpning.

Funktionsvärdet (K) blir 0 om sorteringen gick bra och negativt annars. Följande fel kan uppstå:

1. Det finns ingen variabel P9¤
2. P9¤ innehåller ingen beskrivning, d v s P9¤ har ej tilldelats någon strängvektor
3. P9¤ beskriver ingen strängvektor, d v s tilldelningen är felaktig

I avancerade tillämpningar dessutom:

4. Försök att sortera fler strängar än vad som finns i vektorn
5. Heltalsvektorn är för liten
6. P9¤ innehåller en felaktig beskrivning efter strängvektorn
7. Tabellen för ny sorteringsordning är ej 64 tecken lång

2. AVANCERADE TILLÄMPNINGAR

2.1 Fullständigt anropsformat

Sorteringsrutinen kan egentligen anropas med fyra olika parametrar enligt följande format:

P9¤ = "B¤ , N%, X%, C¤ " :k = CALL (– 3703 %)

B¤ = En strängvektor, som skall sorteras, d v s nycklarna.

N% = En heltalsvariabel, som anger antalet använda strängar i strängvektorn B¤

X% = En HELTALSVEKTOR, som efter sorteringen anger ordningsrummet för de olika strängarna i B¤

C¤ = En STRÄNGVARIABEL, som definierar den sorteringsordning (kollationssekvens) som skall gälla.

Variablerna N%, X% och C¤ behöver ej anges, jfr. avsnitt 1. Det är även möjligt att endast ange en av variablerna vid anropet.

Variabelnamnen kan väljas fritt.

2.2 Längdangivelse av antal element

Många gånger dimensioneras en strängvektor för att rymma fler strängar än vad som normalt används. Man vill då kunna begränsa sorteringen till färre antal strängar i vektorn. Detta antal läggs i en heltalsvariabel och variabeln namnges i P9¤ . Antag att N% innehåller strängvektorns B¤ aktuella antal strängar i vektorn:

P9¤ = "B¤ , N%" : K = CALL (– 3703 %)

Här sorteras endast de N% första strängarna i vektorn.

2.3 Nyckeltabell

Om mån vill sortera ett antal poster med hjälp av en nyckel, d v s om man vill sortera en del av posten, skall naturligtvis endast nyckeln ingå i sorteringen. Efter sorteringen av dessa nycklar, måste man då veta de till nycklarna motsvarande posternas relativa ordningsnummer. Detta är möjligt genom att samtidigt sortera en heltalsvektor i samma ordning som strängvektorn med nycklarna blir sorterad.

Denna heltalsvektor ger på detta sätt en referens till posternas adresser. Man kan således låta posterna ligga oförändrade och endast använda heltalsvektorn för att ange deras inbördes ordning.

Denna metod är användbar oavsett om posterna är lagrade i primärminnet eller på skivminnet. I det senare fallet anger heltalsvektorn relativa postnummer.

Antag att vi har skrivit:

```
DIM B¤ (500) = 10, X % (500)
```

Sedan vi överfört 500 nycklar till B¤ skriver vi till exempel:

```
FOR K = 1 TO 500 : X % (K) = K : NEXT K
```

Vi har nu tilldelat heltalsvektorn X% värdena 1, 2, 3 . . . . . .

Därefter sorterar vi B¤ och X% med:

```
P9¤ = "B¤ , X%" : K = CALL (– 3703 %)
```

Efter sorteringen anger då första talet i X%, vilken sträng som skall komma först. Det andra talet i X% anger vilken sträng som skall komma som nummer två, och så vidare.

2.4 Sorteringsordning

Sorteringsordningen är densamma som för tecken enligt ASCII-kod utom för intervallet 64–127 (stora och små bokstäver). Där gäller följande ordning:

A a B b C c D d E e É é F f G g H h I i J j
K k L l M m N n O o P p Q q R r S s T t U u
V v W w X x Y y Ü ü Z z Å å Ä ä Ö ö _ ■

Om två strängar börjar likadant men är av olika längd, sorteras den kortaste först. Om två strängar är helt lika bibehålles den ursprungliga ordningen mellan dem (sorteringen är stabil).

2.5 Ändring av sorteringsordning

Sorteringsordningen kan, om så önskas, modifieras. Man måste då specificera en ny tabell för intervallet 64–127. För detta ändamål används en strängvariabel.

Antag att C¤ innehåller denna tabell. Sorteringsprogrammet kontrollerar att tabellen är 64 tecken lång. Med basicprogrammet NEWSEQ kan du kontrollera att alla tecken finns med i tabellen och att inga tecken förekommer två gånger. På detta sätt är det till exempel möjligt att definiera sortering i fallande ordning.

2.6 Demoprogrammet QUICKSORT

Sätt i programskivan i DRO, och skriv: RUN START och tryck RETURN.

Sorteringsrutinen laddas då in och ABC80 skrivs ut på skärmen. Kör Demoprogrammet genom att skriva RUN SORTDEMO och trycka RETURN.

Programmet genererar 200 slumpmässiga strängar med längden 3 tecken. Strängarna skrivs ut på skärmen.

Starta därefter sorteringen genom att trycka RETURN. Programmet exekveras och de sorterade strängarna samt tid för sorteringen skrivs ut på skärmen.

Du kan lätt ändra antal strängar för att köra andra exempel än det som finns lagrat på Din programskiva.

Antal strängar ändras på rad 10. N% sätts då till önskat antal strängar.