Datastrukturer og Algoritmer, dat C
Henning Christiansen
2/10-2002

Opgaver til forelæsningen 1/10-2002 om Rekursion, dynamisk programmering, m.v.

Det foreslås at prioritere 1.1, 1.2, 1.4 og 3 højest. Opgave 2 er velegnet som hjemmesyssel.

Opgave 1
Betragt programmet side 271 i bogen som handler om at give byttepenge tilbage (programtekst med hovedprogram kan findes via link på kursushjemmesiden).

1. I forelæsningen forklarede vi kun den del af algoritmen, som handlede om at bestemme det mindste antal antal mønter, som skal til for at give et vist beløb tilbage. Men algoritmen indeholder faktisk også noget ekstra, som gør det mulig at udskrive hvilke mønter der faktisk er tale om. Men det er ikke forklaret hvordan eller hvorfor det virker — og de benyttede variabelnavne er heller ikke hjælpsomme. Opgaven lyder nu: Find ud af og forklar for hinanden hvordan denne del af algoritmen virker.
2. Nu skal programmet tilpasses danske forhold, og for nemheds skyld, regn alt i ørebeløb. Indsæt data svarende til danske sedler og mønter, og tilføj princippet "rund-op-ned-til-nærmeste-25øre". Implementér og aftest programmet.
3. Algoritmen synes ikke at være specielt effektiv.
    - F.eks. udnytter den ikke — eller antager ikke, at møntenhederne optræder i voksende rækkefølge i arrayet. Er der steder i programmet, hvor man kunne optimere under den antagelse?
    - En anden mulig kilde til ineffektivitet er at algoritmen konstuerer den samme løsning på forskellig måde. F.eks. hvis der skal gives 9 kr tilbage, så konstrueres den optimale løsning 5+2+2 ved først at prøve med en 2kr-mønt og slå op i tabellen og se hvordan man giver 7kr tilbage (5+2); men den konstrueres også først at prøve med en 5kr-mønt, og derefter slå op i tabellen og se hvordan man giver 4kr tilbage. Vil det kunne betale sig at sætte noget ekstra bogholderi på til at undgå det? Eller vil det blot sløve udregningen ned alligevel? [Det er OK at argumentere intuitivt].
4. Der findes faktisk mere optimale måder at give penge tilbage på end den som vor algoritme når frem til. Tag beløbet 99kr. Vi vil forvente 50+20+20+5+2+2, dvs. 6 mønter. Men den kan klares med to enheder: "har du en krone, får du en hund af mig". Tilpas algoritmen så den tillader denne måde at give tilbage på og minimerer det totale antal mønter, som gives tilbage.
5. Bogen omtaler en "greedy" algoritme side 268, som nogen gange virker og nogen gange ikke virker. Vil den virke i Danmark? Hvilke krav skal være opfyldt af en møntsystem for at "greedy" virker?

Opgave 2

Tag udgangspunkt i programmet ”drawRuler” side 247 i bogen (programtekst med hovedprogram han findes via link på kursushjemmesiden). Det indeholder noget primitivt grafik, som viser hvordan man definere sig en tegneplade og tegne en streg. Det er sådan set det eneste vi skal bruge fra det program. Brug disse faciliteter til at løse så mange af følgende opgaver som du gider, eller hit selv på nogen, der er sjovere:

- Skriv et program som tegner en ramme – og rekursivt inde i den en ramme som er 85% mindre på begge leder — og rekursivt inde i den .... . Vær opmærksom på, du selv må hitte på et stopkriterium.
- Du har sikkert læst “ramme” som et rektangel hvis sider er parallelle med tegnefladens sider. Skriv nu et program som først tegner en ramme (i nævnte betydning). Dernæst inde i den en rombe (dvs. en firkant på højkant), hvis hjørner rører ved rammens sider. Inde i den igen en ramme hvis hjørner rører ved rombens sider — og så fremdeles. Benyt to metoder som skiftevis kalder hinanden.
- Tegn en rekursiv busk, som ca. ser ud som et “Y” med to mindre buske i grenene. I første omgang, så Y’erne alle peger opad, men prøv at dreje dem lidt hver gang, så det ser mere dekorativt ud.
- Bland Fibonacci-tal ind i historien og se om du kan få tegningen til at konvertere mod det gyldne snit. (Det gylde snit kan findes som grænseværedien for forholdet mellem to Fibonacci-tal:

  1/1, 1/2, 2/3, 3/5, 5/8, 8/13, 13/21, 21/34...

Opgave 3

Betragt de fire versioner af Fibonacci-beregnere i foreslæsningsOHerne (via kursets hjemmeside), og angive “O” udtryk for dem.

1. Angiv “O” udtryk for dem hver især, når de kaldes én enkelt gang med et eller anden n.
2. Angive “O” udtryk for et program som ikke laver andet end at kalde på Fibonacci-metoden. Det antages, at den kaldes m gange med et argument som er begrænset opadtil med n.