VAKKER PROGRAMVAREKODE

Å skrive programvarekode handler ikke bare om at koden skal gjøre det den er ment å gjøre. Mange mener at koden også skal være vakker og i alle fall ikke stygg. En studie utgitt i 2011 (The aesthethic of software code: a quantitative exploration, Journal of Pscyhology of Aesthetic, Creativity and the Arts) viser at det er svært vanlig å gjøre vurderinger av hvor vakker programvarekoden er og at erfarne programmere er de mest estetikk-orienterte. Studien fant også at vurderinger av estetikk blir gjort før vurderinger av korrekthet og at mye tyder på at identifikasjon av stygg kode brukes som indikator på problematisk kode.

Studien inngår i en mer enn 50-årig debatt om i hvilken grad programmering er og bør være en kunstnerisk aktivitet, og i hvilken grad den bør sees på som et håndverk eller en ingeniørvirksomhet. Mange av de beste programmererne er tydelige på hva *de* mener om dette gjennom boktitler som ”Beautiful code” (Oram & Wilson) og ikke minst klassikeren ”The Art of Computer Programming” (Donald Knuth). Det kan ofte være et innslag av praktisk nytte i argumentasjonen for vakker kode, som når Yukihiro Matsumoto (skaperen av programmeringsspråket Ruby) i boka ”Beautiful code” skriver: ”Dersom et program er vanskelig å forstå, er det ikke vakkert.” Donald Knuth skriver i sin ACM Turing Award Lecture (1974) at: ”En programmerer som ubevisst ser seg selv som en kunster vil trives mer med det han gjør og gjøre det bedre.” Av og til så er estetikken nok i seg selv. Jon Bentley (forfatteren av boka Programming Pearls) mener at det vakreste programmet han har skrevet er den følgende sortering av en tallrekke x[i], der swap(i,j) bytter om innholdet i x[i] og x[j]:

```
void   quicksort(int t, int u)
{      int i, m;
       if (t  >= u) return;
       swap(t, randint(t, u));
       m = t;
       for (i = t + 1; i <= u; i++)
            if (x[i] < x[t])
                swap(++m, i);
       swap(t, m)
       quicksort(t, m-1);
       quicksort(m+1, u);
}
```

For mange vil det være uforståelig og kanskje også irrelevant at denne koden er vakker. For Jon Bentley er imidlertid denne koden så enkel, elegant og konsis at den gir store estetiske opplevelser. I studien ”Experience of programming beauty: some patterns of programming aesthetic”, International Journal of Man-Machine studies, 1988, finner forfatterne at det er store likheter mellom hva som oppfattes som estetisk innen programmering, som

opplevelsen til Jon Bentley, og de estetiske opplevelser innen andre felt. De finner at viktige kjennetegn på vakker kode er at den er gjenkjennbar (”familiar”), godt strukturert og innovativ. I tillegg opererer de med en ”glimpse” faktor, dvs at kode oppfattes som vakker dersom den inneholder hint om noe større, f eks hint om hvordan programmet inngår i et større system. Det finnes også røster som hevder at programmering ikke har noe med estetikk å gjøre. Rosenberger (Elegance and entropy, 1997) intervjuet en del programmerere og fikk også utsagn i retning av at et dataprograms mening er kun at det gjør det som det er ment å gjøre. Som i kunsten for øvrig, så er det ulike oppfatninger om hva som er vakkert. Lammers (Programmers at work, 1986) sammenligner denne uenigheten med at noen liker moderne poesi mens andre liker bedre den klassiske varianten.

Det er sparsomt med empirisk forskning på effekten av å være opptatt av estetisk kode. Det som imidlertid er rimelig klart er at vakker og stygg kode betyr noe for mange utviklere. I tillegg synes det å være et mønster at økt kompetanse i programmering henger sammen med økt hyppighet av estetiske opplevelser og økt fokus på estetikk. Andre gode argumenter for vakker kode er at de samme egenskapene som gjør programvarekode vakker er også egenskaper vi gjerne forbinder med kode som er lett å forstå og med lite feil, samt at det er en trivselsfaktor å omgi seg med og lage vakre ting. Alt dette er gode grunner for å etterstrebe vakker og unngå stygg kode. En viss varsomhet må likevel utvises. Det finnes tilfeller av vakker kode med lav kvalitet og ikke minst programmerere som bruker alt for lang tid på lage vakker kode i stedet for å være fornøyd med kode som er bra nok.

Det er kanskje ikke så viktig å avgjøre i hvilken grad programmere er kunstnere, håndverkere og/eller ingeniører. Mer viktig er det å ha forståelse for at estetiske opplevelser av programvarekode spiller en rolle og trolig bør fortsette å gjøre det.