David Taub

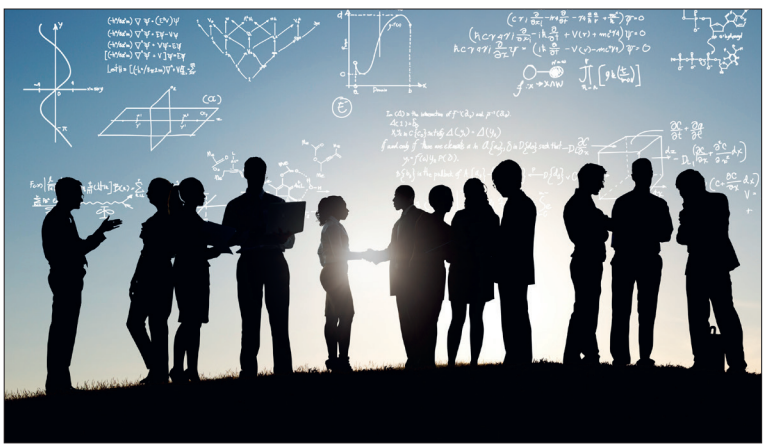
Programmering och skolmatematik

Programmeringens plats i skolans matematikundervisning är något som allt oftare ifrågasätts. Som ett inlägg i debatten vill författaren beskriva preliminära resultat från sin forskning kring programmering i matematik på gymnasiet.

Programmering i skolmatematiken är en aktuell fråga. I Nämnaren 2023:1 fanns en artikel av Bråting, Kilhamn och Lennartsson där resultat från ett flerårigt forskningsprojekt kring införandet av programmering i matematikundervisning i svenska skolor redovisades. Deras projekt fokuserade på läromedel och lärare, och resultaten var inte särskilt uppmuntrande. De, precis som flera andra med liknande forskning, ställer sig kritiska till de ändringar som gjorts i läroplanen för att inkludera programmering i matematikundervisningen.

Det kan då vara intressant att beskriva de preliminära resultaten från min egen forskning där jag har undersökt möjligheter att undervisa matematik genom programmering. Jag har angripit problemet från ett annat perspektiv och fokuserat på eleverna. Jag har samlat in data från ett flertal försök att skapa genomtänkta programmeringsaktiviteter som skulle kunna hjälpa eleverna att utveckla en djupare förståelse inom ett matematiskt område.

Även om jag är långt ifrån att vara färdig med att analysera den insamlade empirin kan jag redan nu ana vad den kommer att visa. Precis som flera andra har föreslagit, kommer mina resultat troligtvis att antyda att det är en dålig idé att undervisa programmering som en del av matematik. Det som är skillnaden är att mina resultat också antyder att när eleverna redan kan programmera finns det bra sätt att använda programmering i matematikundervisningen.



Är programmering svårt?

Man kan hitta artiklar långt tillbaka som beskriver alla möjliga problem och svårigheter elever har med att lära sig att programmera, ett sådant exempel är Benedict du Boulays arbete från 1986. Även med mer moderna programmeringsspråk och många år av undervisningserfarenhet har den allmänna uppfattningen knappt ändrats: programmering är svårt att lära sig. Eleverna har fortfarande svårt med variabler, loopar, syntax och så vidare.

Ett intressant och återkommande problem som man ser i litteraturen handlar om hur elever förhåller sig till en dator när de skriver kod. Många elever antar (troligtvis omedvetet) att datorn vet vad de menar istället för att bara göra exakt som de har skrivit. Det vill säga de förhåller sig till datorn som om den var en människa. Jag, och gissningsvis många andra lärare, känner igen detta från matematikundervisningen. Jag kan inte räkna hur många gånger jag har kommenterat en elevs otydliga redovisning och får till svar ”men du vet vad jag menar”. Man kan lätt undra om det finns en viktig koppling här värd att undersöka.

Ett exempel

```
y = 5
x = y
y = 6
print(X)
```

Det är inte ovanligt att elever tror att koden här ovanför innebär att programmet kommer att skriva ut 6 istället för 5 eftersom de antar att datorn ”vet” att de vill att x och y ska fortsätta att vara lika med varandra. Det är kanske värt att nämna att detta ”missförstånd” faktiskt kan vara sant i mer avancerade situationer där man programmerar med ”pointers”, men inte av den anledning som nybörjare tror när de gör detta misstag.

Det känns viktigt att poängtera att den största delen av moderna artiklar om svårigheter med att lära sig programmering handlar om universitetsstudenter som läser en programmeringskurs. Tänk på detta en stund – många universitetsstudenter som har valt att läsa en introduktionskurs till programmering, världen över, har svårigheter med nästan alla grundläggande begrepp inom ämnet. Jämför detta nu med idén att undervisa lite programmering ”vid sidan av” matematik till yngre elever.

Men är det verkligen svårt?

Det är värt att påpeka att man kan hitta andra åsikter i litteraturen. Till exempel påstår Andrew Luxton-Reilly att det inte alls är svårt att lära sig att programmera. Han menar att påståendet inte bara är fel utan rent av skadligt. Problemet enligt honom är istället att lärare har orimliga förväntningar på elever. Han gör skillnad mellan att säga att programmering är svårt i sig jämfört med att erkänna att det tar mer tid att lära sig det. De olika synsätten kan påverka hur människor upplever tillgänglighet till programmering: Är det bara de ”smarta” som kan lära det? Eller kan alla lära det? Då kan det bli en fråga om jämlikhet.

Min egen forskning

I min egen forskning har jag undersökt om programmering i sig, som en aktivitet, skulle kunna hjälpa elever att få en djupare förståelse för matematik. Här är själva kodskapandet i fokus, och inte det färdiga programmet. Syftet var att se om elever genom att jobba med olika programmeringsaktiviteter skulle kunna lära sig att lösa andragradsekvationer med hjälp av nollproduktmetoden.

Som jag redan nämnt är jag inte färdig med min analys, men redan nu kan jag se vartåt det lutar. Jag tror att jag kommer att hitta stöd för att elever som redan kan programmera skulle kunna använda aktiviteter liknande dem jag har konstruerat för att utveckla en djupare förståelse för matematik. Men för att vara effektivt skulle det kräva att eleverna är ganska bekväma med programmering innan de börjar med aktiviteterna. Det vill säga det blir ingen bra kombination att försöka lära sig ny matematik och att programmera samtidigt. Vid de tillfällen där det fungerade bra, och som jag hade hoppats, var eleverna tvungna att synliggöra sina egna resonemang för att klara av programmeringsaktiviteterna. Jag kan gå genom ett enkelt exempel som var en liten del av projektet.

Programmera faktorisering

I en aktivitet behövde eleverna skriva ett program som skulle fråga en användare efter ett heltal (positivt eller negativt). Programmet skulle därefter skriva ut alla faktorpar, positiva och negativa, till det angivna talet, exempelvis (2,3) som ett faktorpar till talet 6.

Eleverna började med att skriva upp all faktorpar till talet 6 med penna och papper och det hade de inget problem med att göra. Men när de skulle beskriva processen blev det plötsligt mer komplicerat. Här kunde man se att ett inadekvat matematiskt språkbruk var en del av problemet, men också att de hade implicit kunskap om processen som var svårt att göra explicit. De fick sedan försöka skriva ett program som skulle göra det med ett godtyckligt heltal. En enkel lösning i Python skulle kunna se ut så här:

```
tal = int(input("Ange ett heltal: "))
for delare in range(1,abs(tal)+1):
    if tal%delare == 0:
       delare2 = tal//delare
       print(f"({delare}, {delare2})")
       print(f"({-delare}, {-delare2})")
```

Här blev det fullständigt stopp för de flesta. Inte på grund av brist på programmeringskunskap, men på grund av att deras kunskap om faktorisering inte var fullständig. De fick därefter försöka att göra samma sak med penna och papper med talet 24 och uppmanades att verkligen tänka genom hur de gjorde. Och sedan tillbaka till programmering.

Vad jag såg var att de behövde omvandla sin implicita kunskap till explicit kunskap för att kunna skapa programmet. De behövde kunna beskriva hela processen på ett tillräckligt bra och korrekt matematiskt sätt för att kunna systematisera den i ett program. Programmets funktionalitet var inte i sig intressant, bara att det fungerade. Istället var det processen att skriva programmet som tvingade dem att utveckla sin kunskap.

Tyvärr blev brist på programmeringserfarenhet ändå ett stort hinder för några elever. Även med ett så enkelt program fastnade de i detaljer kring loopar, variabler, strängar, format på utskrift och så vidare. Det fanns inte mycket kapacitet eller tid kvar för att fokusera på det matematiska innehållet. Det vara bara med de elever som hade lite mer programmering med sig som det fungerade som det var tänkt.

Programmering ska vara ett eget ämne

Det tar tid och fokus att lära sig att programmera. Det är inte någonting man bara kan göra lite ”här och där” eller vid sidan av något annat. Inte om man förväntar sig att det ska bli användbart på något sätt. Det kräver dessutom sin egen didaktik. Detta är uppenbart när man tittar genom litteraturen kring svårigheter med att lära sig att programmera.

Liksom på många andra håll finns det även i Sverige programmeringskurser med utbildade programmeringslärare. Varför skulle man tro att lärare som inte är utbildat i vare sig programmering eller programmeringsdidaktik skulle kunna undervisa programmering som en ”side dish” i ett annat ämne? Jag läste nyligen ett 2014 citat från Stephen Bloch, en universitetslärare i programmering. Han sade detta till sina studenter första dagen i en introduktion till sin programmeringskurs:

> Suppose I assigned you to write a fifteen-page paper, due two months from now, on Napoleon’s invasion of Russia. [. . .] Now suppose I told you that the paper must be written in Swedish, using a quill pen. Now what would you need to know? [. . .] And if I expected you to learn all those things, from scratch, in one semester, you’d think I was nuts. That’s what we’ll be doing in CS1, trying to learn half a dozen different levels of knowledge at once.

Om det verkligen är viktigt att våra elever ska lära sig att programmera behöver det vara ett eget ämne, så som man har gör i några andra länder, och inte bara vara en liten del av matematikundervisningen.

LITTERATUR

Bosse, Y., & Gerosa, M. A. (2017). Why Is Programming So Difficult to Learn? Patterns of Difficulties Related to Programming Learning Mid-Stage. *ACM SIGSOFT Software Engineering Notes*, *41*(6), 1–6.

Bråting, K., Kilhamn, C. & Rolandsson L. (2023). *Vad hände med programmeringen?* Nämnaren 2023:1.

du Boulay, B. (1986). Some Difficulties of Learning to Program. *Journal of Educational Computing Research*, *2*(1), 57–73.

Hector, E. & Thelander, L. (2022). *Lärartankar – kan programmering bidra till lärande i matematik?* Nämnaren 2022:1.

Luxton-Reilly, A. (2016). Learning to Program Is Easy. *Proceedings of the 2016 Acm Conference on Innovation and Technology in Computer Science Education*.