GÖTEBORGS UNIVERSITET

ÄMNESLÄRARPROGRAMMET

LÄRARE, PROGRAMMERADE FÖR FÖRÄNDRING?

En kvalitativ studie över lärares förhållningssätt till styrdokumentens skrivningar gällande programmering.

Liv Ejsing

Ämneslärarprogrammet med inriktning mot arbete i gymnasieskolan

Självständigt arbete (examensarbete):	15 hp
Program och kurs:	Ämneslärarprogrammet, LGMA21A
Nivå:	Avancerad nivå
Termin/år:	VT/2021
Handledare:	Johanna Pejlare
Examinator:	Laura Fainsilber
Nyckelord:	Matematik, programmering, styrdokument, undervisning

Abstract

During the last three years programming has been a part of the school curricula in Sweden and should be seen as a compulsory element in mathematics education. The following study aims to show how the implementation of programming has transformed the subject of mathematics in school settings, teachers' experiences and to explore the possibilities that programming provides to traditional mathematics. Through interviews with four working math teachers in secondary schools, an insight is given on how education in mathematics is conducted and combined with programming elements.

The study is theoretically embedded in Chevallards framework about transposition of knowledge, which can be described as a theory that shows how knowledge is transposed to fit different stages in the educational system. By interviewing teachers, it is possible to answer three questions that relates to different stages in the didactical transposition process. They are 1) *How do teachers interpret the national curriculum regarding programming in mathematics?* 2) *How do teachers conduct education with programming in mathematical problem solving?* and 3) *What effects can teachers see that programming has on the pupils learning?* The second question will be answered by examining educational materials that the teachers have used in their education. The material is analyzed with an instrumental approach to better understand how pupils can develop mathematical knowledge by using a technical object.

The qualitative data from the interviews has been analyzed with a thematic method. By doing so the study can conclude that the teachers all perceive the curricula differently and therefore the education in programming may vary and sometimes even be deprioritized, since it is not a skill that pupils are being graded on. Programming can successfully be taught as a mathematical tool, but this demands commitment from teachers who often lack didactical knowledge in programming. This is possibly the reason why the teachers in the study could not see if their pupils developed any new skills by using programming.

Förord

Jag vill inledningsvis passa på att tacka min handledare, Johanna Pejlare, som har gett mig nyttig återkoppling på arbetet under hela processen. Min examinator, Laura Fainsilber, och opponent, Felicia Olofsson, förtjänar också tack för det arbete de har lagt ner på att läsa studien och komma med goda råd. Ett tack ska även min sambo ha som, för det mesta, har låtit mig plugga ifred och alltid stöttat mig när det varit tufft.

Det största tacket vill jag dock ge mina underbara medstudenter Frida Haglund och Julien Eskola, som gett den bästa feedbacken och alltid fått mig att skratta. Vem vet vad det här arbetet hade varit utan er.

Liv Ejsing

Innehållsförteckning

Figurförteckning

1. Inledning

Att välja en karriär som lärare innebär också ett val av en arbetsmiljö som är under ständig förändring. Idag har många verksamma lärare upplevt mer än ett betygssystem och Skolverkets styrdokument revideras och omformuleras på en regelbunden basis. Det här är ingen studie som har för avsikt att rikta kritik mot detta fenomen men det väcker en fundering om det ges möjlighet för lärarna att "hänga med i svängarna".

Jag går nu min sista termin på lärarutbildningen och under min tid som student har ett nytt moment införts i båda mina undervisningsämnen, matematik och teknik. Det nya momentet är programmering, något som jag absolut inte har några förkunskaper i. En förhoppning var att utbildningen skulle förbereda mig för att undervisa även i detta, vilket tyvärr inte visat sig vara verkligheten. Visst, vi har fått diskutera de nya skrivningarna om programmering och haft programmeringsinslag i vår undervisning. Det är dock inget som jag kände var tillräckligt för att känna mig förberedd till att undervisa i det. För mig var valet av ämne till examensarbetet ett sätt att fördjupa mina kunskaper om kraven som ställs på mig som lärare inom programmering, få inspiration till lektionsupplägg och lära mig utav erfarna lärares undervisning om just detta.

Det största frågetecknet har för mig varit hur programmering är tänkt att undervisas på ett sätt som skulle gynna elevernas matematiska kunskapsutveckling. Jag har svårt att se hur ett vetenskapsfält, programmering, ska användas på ett annat vetenskapsfält, matematik, utan att först behöva lära sig att programmera. Skolverket har varit tydlig i sina avsikter, det är inte meningen att de nya skrivningarna ska leda till att eleverna blir skickliga programmerare. Samtidigt som arbetsmarknadens framtida behov av programmerare är en av de betydande anledningarna till införandet. De här två målbilderna fick jag inte att gå ihop. Hur ser aktiva, erfarna lärare på frågan nu, tre år efter att förändringen av styrdokumenten tog plats? Följande studie undersöker fyra verksamma lärares uppfattningar och erfarenheter om programmering. De får, genom studien, en chans att yttra sina åsikter om införandet och dela med sig av tankar och lärdomar som de stött på i undervisningssammanhang. Studien kan vara hjälpsam för samtliga matematiklärare som känner en osäkerhet kring programmeringsundervisningen och inte vet vart de ska vända sig.

1.1 Syfte och frågeställningar

Studien ämnar undersöka hur lärare tolkar och tillämpar programmeringstillägget i styrdokumenten. Utifrån lärarperspektivet studeras även hur eleverna upplever programmeringsmomenten och vilka kunskaper lärarna anser att eleverna får med sig. Intentionen är att se om undervisningen bedrivs i enlighet med vad Skolverkets avser och om programmering utvecklar eleverna på det sätt som Skolverket önskar. Utifrån detta formulerades följande frågeställningar:

- Hur förhåller sig lärare till styrdokumentens skrivningar om programmering i matematik?
- Hur undervisar lärare om programmering i samband med matematisk problemlösning?
- Vilka effekter uppfattar lärare att programmering har på elevers kunskapsutveckling?

1.2 Avgränsningar

Skrivningarna om programmering har placerats under problemlösning för både gymnasiet och högstadiet. Av den anledningen har studien avgränsats till att enbart betrakta skrivningarna om programmering som berör problemlösning, även om programmering också förekommer under algebra i grundskolan. Det gör att lärare på gymnasiet såväl som på högstadiet kan ingå i studien.

2. Bakgrund

Nedan följer en bakgrund som ämnar att sätta undersökningen i en teoretisk kontext. Här kommer styrdokumentens, nuvarande och framtida, skrivningar innehållandes programmering att presenteras tillsammans med Skolverkets definition av programmering och digitala verktyg. En kort redogörelse för programmeringens roll i skolan kommer att följas av en utförlig presentation av aktuell och relevant forskning om programmering som ett inslag i skolmatematiken.

2.1 Styrdokument

Sedan höstterminen 2018 ska undervisas som en del av matematikundervisningen i alla årskurser på grundskolan och i många matematikkurser på gymnasiet. Utöver det ska programmering även förekomma i teknikundervisningen i grundskolan och finnas som fristående kurs på gymnasiet.

Skolverket har en avsiktlig och tydlig progression med programmering från grundskolans tidiga årskurser till de senare (Skolverket, 2021a), för att sedan skilja sig markant åt i formuleringarna för gymnasiet. Målet är att elever ska erhålla tillräckligt med kunskap och erfarenhet om programmering i de lägre åldrarna för att de på högstadiet och gymnasiet mer ska kunna använda det som ett verktyg vid matematisk problemlösning. Däremot är programmering ännu inte explicit en del av kunskapskraven i matematik, varken i grundskolan eller på gymnasiet.

2.1.1 Grundskolan

Skolverket (2021a) ser undervisning om digitala verktyg och programmering som en förutsättning för att fostra demokratiska medborgare i ett alltmer digitaliserat samhälle. Av den anledningen är programmering numera en del av matematikämnets syfte, som gäller för alla årskurser på grundskolan. Formuleringen som gäller lyder:

> Eleverna ska ges möjligheter att utveckla kunskaper i att använda digitala verktyg och programmering för att kunna undersöka problemställningar och matematiska begrepp, göra beräkningar och för att presentera och tolka data.
>
> (Skolverket, 2018a)

I samtliga årskurser på grundskolan står programmering som en del av kunskapsområdet ”algebra” i den centrala innehållet. Målet är att eleverna ska få bekanta sig med grunderna i programmering i olika miljöer (Skolverket, 2021a). I de lägre årskurserna ska eleverna lära sig stegvisa instruktioner i visuella programmeringsmiljöer. Denna kunskap ska sedan utvecklas till att skapa och testa egna algoritmer i textbaserad programmeringsmiljö på högstadiet. I takt med att eleverna blir mer bekanta med programmering, ska de kunna använda programmering som ett verktyg vid problemlösning. Av den anledningen finns programmering även under kunskapsområdet ”problemlösning” i det centrala innehållet för årskurs 7-9.

Skrivningarna om programmering under punkten algebra i respektive årskurs är följande:

> Årskurs 1-3
> Hur entydiga stegvisa instruktioner kan konstrueras, beskrivas och följas som grund för programmering. Symbolers användning vid stegvisa instruktioner.
>
> Årskurs 4-6
> Hur algoritmer kan skapas och användas vid programmering. Programmering i visuella programmeringsmiljöer.
>
> Årskurs 7-9
> Hur algoritmer kan skapas och användas vid programmering. Programmering i olika programmeringsmiljöer.
>
> (Skolverket, 2018a)

Skrivningarna om programmering under punkten problemlösning för årskurs 7-9 är följande:

> Hur algoritmer kan skapas, testas och förbättras vid programmering för matematisk problemlösning.
>
> (Skolverket, 2018a)

2.1.2 Gymnasiet

I gymnasiets ämnesplaner för matematik har programmering en mindre roll än vad den har i grundskolan. Detta beror på att programmering ska finnas som en valbar kurs för de elever som är intresserade (Skolverket, 2021b). I matematikämnets syfte i ämnesplanerna nämns inte programmering explicit. Däremot innehåller den en skrivning om digitala verktyg, dit programmering räknas, som lyder:

> I undervisningen ska eleverna dessutom ges möjlighet att utveckla sin förmåga att använda digitala verktyg för att lösa problem, fördjupa sitt matematikkunnande och utöka de områden där matematikkunnandet kan användas.
>
> (Skolverket, 2018b)

Idag finns programmering endast under problemlösning i det centrala innehållet för matematikkurserna 1c, 2c, 3b, 3c, 4, 5 och specialisering (Skolverket 2018b). Skrivningarna om programmering är identiska för varje kurs och lyder som följer:

> Strategier för matematisk problemlösning inklusive modellering av olika situationer, såväl med som utan digitala verktyg och programmering.
>
> (Skolverket, 2018b)

Med de här skrivningarna angående programmering vill Skolverket medvetet lämna tolkningsmöjligheter till undervisande lärare, om hur programmering ska förekomma i undervisningen (Skolverket, 2017). Vilka programmeringsspråk och miljöer som förekommer är helt valfritt och även kalkylblad nämns som alternativ för att hantera stora data med hjälp av iterativa eller villkorsstyrda beräkningar. Så länge programmeringsmiljöerna är relevanta för eleverna och fördjupar deras matematiska kunnande vid problemlösning, anser Skolverket att undervisningen når sitt mål med programmering.

2.1.3 Revideringar från 2021

Från och med höstterminen 2021 kommer reviderade ämnesplaner på gymnasiet att gälla i matematik, engelska och moderna språk. Regeringen har beslutat om att skjuta på förändringar av kursplanerna i grundskolans ämnen till höstterminen 2022 i och med den påfrestning som skolorna för närvarande upplever under rådande coronapandemi (Skolverket, 2021c).

I kommentarmaterialet till den reviderade kursplanen i matematik framgår det att programmeringsmomenten kommer behålla sin nuvarande formulering (Skolverket, 2021a). Däremot har Skolverket valt att förtydliga vad de skrivningarna innebär i praktiken och ger även exempel på hur undervisningen kan gå till. Förmågor som premieras i årskurs 7-9 är att kunna felsöka, generalisera och förbättra koder genom att kritiskt granska resultaten. Det står även hur programmering med fördel kan kombineras med undervisning av andra ämnen såsom teknik, samhällskunskap och bild.

För gymnasiet däremot kommer revideringarna i ämnesplanen för matematik påverka programmeringens roll avsevärt. Under ämnets syfte skriver Skolverket (2021b) att programmering ingår i begreppet “digitala verktyg”, med samma skrivning som tidigare. Däremot har formuleringen kring programmering tonats ned under centralt innehåll. I kurserna Ma1c och Ma2c kommer eleverna inte längre behöva producera egen kod utan endast möta exempel på hur programmering kan vara användbar i matematiken. Skolverket vill se att programmering tillämpas som ett verktyg inom matematiken. Skrivningen som innefattar programmering ingår i punkten “problemlösning, verktyg och tillämpningar” i det centrala innehållet och har följande formulering:

> Exempel på hur programmering kan användas som verktyg vid problemlösning, databearbetning eller tillämpning av numeriska metoder
>
> (Skolverket, 2021b)

Kraven på programmering ökar i kurserna Ma3b, Ma3c, Ma4 och Ma5. Trots denna revidering lämnar Skolverket fortsatt mycket utrymme till läraren att själv besluta om vilket programmeringsspråk eller miljö som ska användas och inom vilket område som programmering ska tillämpas på (Skolverket, 2021b). Som tidigare kommer även denna nya formulering hamna under punkten ”problemlösning, verktyg och tillämpningar” i det centrala innehållet men har följande formulering:

> Användning av programmering som verktyg vid problemlösning, databearbetning eller tillämpning av numeriska metoder.
>
> (Skolverket, 2021b)

2.2 Digitala verktyg och programmering

I samband med införandet av programmering 2018 släppte Skolverket ett antal moduler för gymnasiet och grundskolan som skulle fungera som stöd för lärare i deras undervisning. Där har Helenius m.fl. (2018) gett sin förklaring på vad programmering egentligen är. Författarna väljer att definiera programmering som att skapa ett datorprogram och att detta är en process

som sker i flera faser, med en struktur som liknas med att följa ett matlagningsrecept. Koder är de instruktioner som ges till programmet och kan sättas ihop till en sekvens. Begreppet ”algoritm” definierar Helenius m.fl. som en uppsättning sådana sekvenser.

Hur Skolverket skiljer på digitala verktyg och programmering är inte helt tydligt. I kommentarmaterialet inför revideringen 2021 (Skolverket, 2021b) står det att digitala verktyg används som hjälpmedel för att illustrera matematiska fenomen och lösa ekvationer innan eleverna behärskar att göra detta för hand. För gymnasieskolan ska programmering ingå i begreppet digitala verktyg. Det finns alltså inget krav på att skapa program, som var Skolverkets definitions enligt Helenius m.fl. (2018), i flera av gymnasiets matematikkurser. Däremot kan programmering var ett sätt att behandla annat matematiskt innehåll.

Ur dessa skrivningar om digitala verktyg och programmering, kan slutsatsen dras att Skolverket ser digitala verktyg som ett paraplybegrepp där programmering ingår. Däremot anses det inte vara samma sak att programmera som att använda ett digitalt verktyg. Av den anledningen skulle det inte räcka att ha använt program som GeoGebra eller redskap som miniräknare i undervisningen för grundskolan om eleverna inte själva är delaktiga i skapandet av programmen, då dessa snarare faller under den bredare kategorin ”digitala verktyg”. Även forskare väl insatta i programmering i svensk matematikundervisning har valt att klassificera dessa som enbart digitala verktyg (Kilhamn m fl., 2021a). Författarna har även valt att räkna kalkylprogram till programmering, vilket Skolverket (2021b) också till viss mån gör. I de reviderade ämnesplanerna förekommer begreppen programmering och kalkylprogram i olika punkter i centralt innehåll för vissa gymnasiekurser, något som kan tolkas som att begreppen är skilda från varandra. Vid förtydligande skriver Skolverket följande:

> Även kalkylblad kan användas för iterativa eller villkorsstyrda beräkningar, vilket är det som ofta används för att kännetecknа programmering. Att låta elever använda iterativa eller villkorsstyrda beräkningar i kalkylblad för att arbeta med problemlösning, databearbetning eller tillämpning av numeriska metoder kan vara särskilt användbart om eleverna saknar relevanta kunskaper i att skriva kod. Samtidigt innehåller kalkylblad många begränsningar som konventionell programmering inte gör. I den mån eleverna behärskar att skriva kod är det därför lämpligt att de får använda denna kunskap för att fördjupa sitt matematiska kunnande.
>
> (Skolverket, 2021b)

Skolverket (2021b) förtydligar också hur skrivningarna om kalkylprogram under centralt innehåll syftar till att eleverna själva ska skriva in data och göra beräkningar och inte bara hantera färdigskrivna program. Här upptäcks en skillnad i hur kommentarmaterialet för grundskolans reviderade kursplanen talar om kalkylprogram. Under en rubrik om statistiskt material nämns kalkylprogram som ett vanligt exempel på ett digitalt verktyg som lämpar sig bra för att bearbeta stora mängder data.

2.3 Programmering i skolan

Teknik har en allt större inverkan på människors liv och påverkar stora delar av samhällets utveckling. För att kunna mätta den framtida arbetsmarknaden inom digitala yrkesgrupper och för att bättre förstå och påverka sin omvärld, har programmering i olika grad tagit sig in i många länders styrdokument de senaste fem åren (Helenius m fl., 2019). Det är lätt att tro att

programmering är ett aktuellt ämne och något nytt i skolans värld men faktum är att programmering och datorer länge har varit ett omdiskuterat ämne för politiker.

Redan på 1960-talet blev programmering en del av undervisningen i grundskolan. (Helenius m.fl., 2019). Det var datavetaren Alan Perlis som förespråkade undervisning av programmering för att fördjupa elevernas förståelse för beräkningsprocessen (Mannila, 2017). På den här tiden var användande av datorer likställt med att programmera men i vilken mån programmering lärdes ut varierade över åren. När matematikprofessorn Seymour Papert utvecklade programspråket LOGO på 80-talet, kunde de första effekterna av undervisningen av programmering studeras. Papert skapade LOGO utifrån sin vision av att lära elever abstrakt matematik genom att konkretisera det i en digital lärmiljö som engagerade eleverna (Papert refererad till i Hickmott m.fl., 2018). Målet var att eleverna skulle utveckla sitt matematiska tänkande genom att lekfullt programmera och visualisera matematik. Papert (refererad till i Misfeldt & Ejsing-Duun, 2015) beskriver hur lärarens roll i sådana undervisningssituationer är att knyta ihop elevernas arbete med traditionella, matematiska koncept och begrepp. Misfeldt och Ejsing-Duun redogör för hur Paperts syn på matematikundervisningen inte levde upp till förväntningarna, då elever lätt kunde förbise matematiken vid programmeringsmomenten och därmed uteblev ofta möjligheten att lära sig just matematik. Dessutom kunde Noss och Hoyles (refererad till i Kilhamn m fl., 2021a) se ett samband mellan förlusten på matematiska lärdomar i LOGO och bristen på didaktisk kunskap om programmering hos lärarna, samtidigt som det ansågs vara ett tidskrävande moment.

Mannila (2017) redogör för hur programmeringsliknande moment implementerades i läroplanen för grundskolan från 1980. Även då hamnade datormoment i matematikens kursplaner med syfte att eleverna skulle få kunskaper om datorer, snarare än att förmågor i att programmera. Förståelse för hur datorer fungerar framhölls som en så pass viktig kunskap att det även tillkom skrivningar om det i både naturorienterande och samhällsorienterande ämnen. I takt med att datorerna blev alltmer användarvänliga och programmeringen bakom dem syntes mindre och mindre, tappade även motivationen för att eleverna skulle erhålla kunskaper om just programmering i grundskolan och togs därför bort och ersattes med datavetenskap. År 2006 släppte Jeanette Wing en artikel där hon argumenterade för att göra *computational thinking* (CT) en lika nödvändig förmåga som att läsa, skriva och räkna (Hickmott. m fl., (2018). Begreppet CT myntades av Seymour Papert nästan 30 år innan Wings artikel skrevs och nu fick begreppet så pass stor genomslagskraft att programmering i skolan åter igen var tapeten i många länder. Begreppet skiljer sig från programmering eftersom det mer syftar till ett algoritmiskt sätt att tänka i problemlösningssammanhang och inte till att producera kod.

Även införandet i Sverige bygger på en definition av programmering som liknar begreppet computational thinking. Skolverket (2016) publicerade ett dokument som utgjorde underlaget för de förändrade kurs- och ämnesplanerna där de redogör för att programmering ska från och med juli 2018 ingå i elevernas digitala kompetensutveckling. I dokumentet ger Skolverket två möjliga tolkningar på begreppet programmering. Den första tolkningen likställs med att skriva kod, något som kanske på senare år omformulerats till att skriva program (Helenius m fl., 2019). Den andra tolkningen, vilken har varit utgångspunkten för implementeringen, innefattar såväl problemformulering som lösningsmetod och dokumentation (Skolverket, 2016). Den här tolkningen medför att kognitiva färdigheter utvecklas så som kreativitet, logiskt tänkande och generaliseringsförmåga, vilket liknar argument som Papert och Wing framförde tillsammans med begreppet CT. I samband med införandet publicerade Skolverket en sammanställning av aktuell forskning inom ämnet och exempel på undervisningsmaterial.

2.4 Aktuell forskning

Valet att inkludera programmering i just matematikämnet hör troligtvis samman med den allmänna uppfattningen om att programmering och matematik på något sätt hör ihop. I en enkätundersökning som gjordes av Misfeldt m fl. (2019), var det många matematiklärare, men inte alla, som höll med om att det finns ett tydligt samband mellan de båda ämnena, även om de inte såg det som samma ämne. Samma fenomen syntes i en studie som jämförde undervisningsmaterial för programmering från myndigheter i Sverige, Danmark och England där programmering endast är specifikt knutet till matematikämnet i Sveriges styrdokument (Misfeldt m fl., 2020). Trots denna skillnad i styrdokumenten, hade alla uppgifter lika explicita kopplingar till matematik när det gäller problemlösning och till viss del även till matematiska koncept. Detta tyder på att det finns en relation mellan de båda ämnena, i alla fall när det gäller att undervisa om programmering.

Kopplingen mellan de båda ämna kan också vara att goda matematikkunskaper hjälper nybörjare inom programmering att skriva kod, något som Veerasamy m fl. (2016) hävdar är ett vedertaget antagande. I deras studie analyserades slutproven för en programmeringskurs för nybörjare på universitetsnivå, för att identifiera vanliga misstag och fallgropar. Till forskarnas förvåning upptäcktes en kunskapslucka hos de novisa programmerarna som deltog i studien när över två tredjedelar av dem inte klarade av att besvara en matematikbaserad uppgift i provet. Slutsatsen som drogs var att även om matematiska kunskaper hjälper nybörjare att lära sig programmera, är det inte nödvändigtvis så att programmering utvecklar utövarnas matematiska kunskaper. Det är en intressant observation eftersom Skolverkets intention var att utveckla matematiska förmågor med hjälp av programmering.

Innan införandet av programmering i Sveriges skolor, fanns det lite forskning som tydde på att den dialektiken som Skolverket ville se mellan de båda ämnena faktiskt fanns. Av den anledningen har det på senare år tillkommit en hel del studier som har valt att utforska sambandet mellan programmering och matematik i skolans värld och om programmering faktiskt kan bidra till förbättrade matematikkunskaper (se till exempel: Bråting m.fl, 2021; Bråting & Kilhamn, 2021; Misfeldt & Ejsing-Duun, 2015; Misfeldt m.fl., 2019; Hickmott m.fl., 2018).

Hickmott m.fl. (2018) undersökte artiklar som innehåller *computational thinking* (CT) i skolundervisningen och som publicerades mellan 2006 och 2016. CT är inte samma sak som programmering, utan syftar mer till de tankeprocesser som sker när elever använder datorer eller algoritmer för att lösa problem men det går att likställa de två begreppen i det här avseendet (Bråting & Kilhamn, 2021). Hickmott m.fl (2018) sorterade och analyserade artiklarna som innehöll begreppet CT utifrån om de uttryckligen eller slumpartat länkat samman CT med matematik. Det visade sig att de flesta av artiklarna inte kopplar samman CT med matematiska koncept alls och bara en cirka en fjärdedel gör det uttryckligen. Här kunde forskarna identifiera behovet av ytterligare studier som länkar samman de två ämnena och främst på ett sätt som undersöker elevernas långsiktiga matematiska fördelar från CT.

Även i läroböcker har det upptäckts att programmeringsmomenten sällan explicit knyts ihop med relevanta matematiska begrepp och koncept. Detta upptäckte Bråting & Kilhamn (2021) när de utförde en kvalitativ innehållsanalys av läromedel för matematik med programmeringsmoment. Studien undersökte läroböcker för årskurserna 1-6 och analyserade programeringsmomenten utifrån Brennan och Resnicks teoretiska ramverk om CT. Att läromedlen inte explicit knyter an programmeringsmomenten till matematiska begrepp och

koncept är särskilt problematiskt eftersom läroböckerna har ett stort inflytande på lektionernas utformning, menar författarna.

För att lyckas med att utveckla elevernas matematiska kunskaper genom programmering, måste läraren aktivt göra kopplingar mellan de båda kunskapsområdena. Detta kunde Misfeldt och Ejsing-Duun (2015) konstatera i deras fallstudie där de observerade matematikundervisningen på olika skolor för elever i årskurs 5. Eleverna skulle lära sig matematiska koncept genom att programmera spel i programmet Hopscotch. Det visade sig att programmeringsuppgifterna själva inte utvecklade matematiska förmågor men att med lärarens hjälp kunde eleverna förstå hur programmering och matematik hänger ihop. För att lyckas knyta ihop de två disciplinerna måste lärare ha goda kunskaper inom både matematik och programmering men tyvärr är det många lärare som uppger sig ha för lite kunskaper om programmering för att hantera det. Detta tror Nouri m.fl. (2020) vara avgörande för resultatet i deras kvalitativa studie där drivna grundskolelärare, som har stor tilltro till att inkludera programmering i matematiken, fick beskriva vilka kunskaper eleverna utvecklade i samband med programmering. Förmågorna som lyftes i intervjuerna låg ofta utanför de som räknas till computational thinking och traditionella matematiska färdigheter. I stället var det många lärare som lyfte kognitiva färdigheter och samarbetsförmåga som exempel.

Denna brist på kunskap inom ämnet visar sig också i lärarnas tolkningar av styrdokumenten där lärare inte visste om till exempel GeoGebra eller kalkylprogram räknades till programmering eller inte, samt att lärarna kändes sig oförmögna att konstruera programmeringsuppgifter själva. (Bråting m.fl., 2020). Bråting m.fl. genomförde en kvalitativ studie med lärare som tidigt inkluderade programmering i matematikundervisningen och som visade stor entusiasm inför införandet av programmering. Lärarna arbetade i varierande årsgrupper i grundskolan och var eniga om att programmering är ett bra sätt att höja elevernas motivation för matematik. Forskarna kunde från lärarintervjuerna konstatera att lärarna oftast hittar inspiration till uppgifter på sociala medier snarare än från läroböcker. De kunde också se att undervisningen främst verkade syfta till att lära eleverna att koda, snarare än att fördjupa deras matematiska kunskaper. Detta strider mot Skolverkets avsikter med programmering, vilket är att elever ska få en bättre förståelse för matematiska koncept och begrepp och inte att de ska bli kompetenta programmerare (Helenius m.fl., 2019).

Förhoppningen från Skolverkets sida är att i takt med att elever och lärare skaffar sig erfarenhet i programmering, kommer det att kunna användas som ett verktyg inom matematiken. Kilhamn m fl (2021b) ställer frågan om det är rimligt att ta tid från matematikämnet om målet är att lära sig hantera ett nytt redskap. Författarna baserar denna åsikt på deras lärarintervjuer med lärare i olika årsgrupper i grundskolan som är drivna och engagerade i programmeringsfrågan, där många lärare uppgav icke-matematiska förmågor som motiv för införandet. Lärarna såg inte heller programmering som en matematisk aktivitet, även om kopplingar till matematiken kunde göras med hjälp av läraren vid undervisningsmomenten. Samma författare har även studerat olika lärargrupper på grundskolan som genomfört “lession studies” i programmering (Kilhamn m fl., 2021a). Lesson studies innebär att lärare grupperas ihop och tillsammans planerar, genomför och utvärderar lektioner och utifrån dessa har forskarna analyserat hur väl lektionerna knyter an till matematiken. Där skiljde sig resultatet från deras andra studie eftersom många lärargrupper hade matematiska mål med lektionerna, även om de inte explicit gjorde någon koppling till matematiken under lektionen. Författarna menar att det är möjligt att lärarna nu, några år efter införandet, börjar se programmering som en del av matematiken och därmed

legitimerar matematiklektioner som bara går ut på att lära sig programmering i stället för att lära sig programmeringens användningsområden i matematiken.

Från Kilhamn m fl. (2021a) studie uppmärksammas även hur lektioner med programmeringsinnehåll bedrivs. Det var ca 135 grundskollärare från 15 olika skolor som deltog i studien vars mål var att dels undersöka relationen mellan matematik och programmering, dels vilket matematiskt innehåll som berördes på lektionerna. Den största gruppen var lärargrupper som enbart undervisade programmering utan att relatera programmering till matematik eller knyta an till några andra matematiska områden. När väl lärargrupper valde att undervisa programmering i samband med andra traditionella matematikområden, var det vanligt att inkludera programmering tillsammans med taluppfattning och geometri. Under lektionerna användes sällan programmering som ett verktyg för att utforska matematik. Endast 4 av de 32 lärargrupperna i studien genomförde undervisningen med programmering som ett matematiskt verktyg och samtliga lärargrupper som gjorde detta undervisade på mellanstadiet.

Kilhamn m fl. (2021a) kunde även se att textbaserade programmeringsmiljöer var vanligast att undervisa i på högstadiet, även om det förekom programmering med block och i Excel även på högstadiet. Valet av programmeringsmiljö har varit föremål för undersökning tidigare. I studien som gjordes av Bråting m fl. (2021) kunde forskarna se att engagemanget hos eleverna varierade beroende på vilken miljö de programmerade i. Blockbaserade miljöer kan stimulera eleverna kreativitet och är inte så hårt bunden till en viss struktur eller syntax, vilket kan vara en fördel för elevernas motivation. Trots det föredrar många lärare att undervisa i textbaserade programmeringsmiljöer för att de lättare går att anpassa på undervisningen och eleverna inte blir lika distraherade av estetiken och val av effekter som de kan bli av blockprogrammering.

3. Teoretiskt ramverk

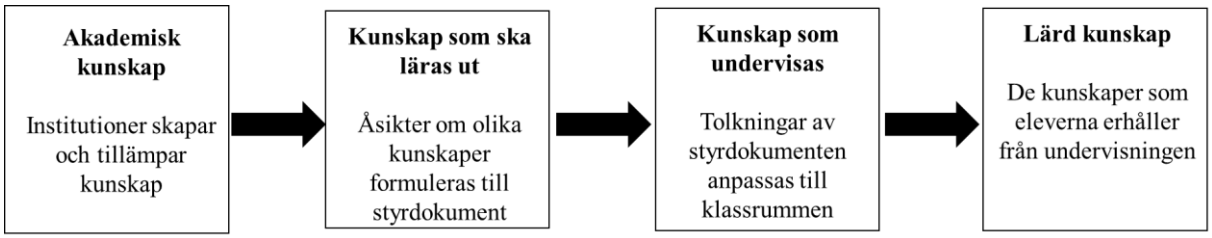
För studien används Chevallards teori om didaktisk transposition som övergripande ramverk. Teorin grundar sig i att kunskap ofrånkomligt förändras i olika steg när den går från att användas och skapas inom forskning till att läras ut till elever i skolan. Programmeringstilläggen som tillkom 2018 bör, enligt ramverket, därför studeras på olika nivåer. Genom studiens kvalitativa metod kan lärares uppfattningar om dessa transpositionsprocesser studeras närmare.

Vid djupare analyser av lärarnas undervisningsmaterial kopplat till programmeringsmomenten används en instrumentell ansats. Ansatsen har ofta använts i samband med att tekniska inslag infinner sig i matematikundervisningen och syftar till att undersöka hur dessa artefakter transformeras till ett matematiskt verktyg.

3.1 Chevallards teori om didaktisk transposition

Chevallard introducerade sin teori om didaktisk transposition på en sommarskola för matematikdidaktik i Chamrousse, 1980 (Bosch & Gascón, 2006). Vid den här tidpunkten influerades den matematikdidaktiska forskningen starkt av psykologins påverkan på inlärning och hur det praktiska arbetet gick till i klassrummen. Detta ansåg Chevallard inte vara tillräckligt för att skaffa sig en uppfattning om ett didaktiskt fenomen, utan ansåg att kunskapsstoffet som ska undervisas i skolan måste studeras från alla berörda aktörers synvinkel. Chevallard (2006) beskriver hur forskare på en institution letar efter svaret på en viss fråga inom ett forskningsområde, som är helt frånkopplat skolans värld. Vetenskaperna som studeras på institutionerna har ett tydligt syfte men sedan tvingas dessa kunskaper att rekonstrueras för att passa in i andra miljöer, som till exempel skolan. Det är detta som Chevallard kallar för *transpositionsprocessen*.

Anledningen till att det sker en förändring, eller transposition, av kunskap är för att skolans mål är att lära ut och reproducera de befintliga kunskaper som skapas av forskare inom olika akademiska discipliner. Efter att forskare lyckats besvara en fråga, och därmed skapat kunskap, sker ett gediget omformningsarbete av kunskapen i vad Chevallard kallar för ”the noosphere” (Fig. 1) (Chevallard & Bosch, 2013). I det här steget har alla som ”tänker” kring utbildning en röst, det vill säga experter och medborgare såväl som politiker. Målet är att göra kunskapen tillgänglig för elever samtidigt som vetenskapen känns som ”äkta” och som något som faktiskt har en funktion utanför skolans värld (Bosch & Gascón, 2006). För att programmering ska bli undersökningsbart i skolan, måste ämnet brytas ner i beståndsdelar och noggrant studeras för att kunna plocka ut de delar som kan göra någon nytta för elevernas kunskapsutveckling samtidigt som att programmeringen behåller sin essens. Resultat av transponeringen från den akademiska kunskapen programmering är de skrivningar som finns i kurs- och ämnesplaner idag, samt Skolverkets moduler som beskriver anledningen till att olika delar har valts ut och hur det är tänkt att det ska läras ut. Kunskapen har gått från att vara rent akademisk till att bli kunskap som ska läras ut, fas 2 av transpositionsprocessen. Den andra transpositionsprocessen som sker är när lärare och läromedelsförfattare ska tolka och applicera styrdokumenten i klassrummen, vilket blir den kunskap som faktiskt undervisas (Chevallard & Bosch, 2013). Den sista transpositionsprocessen sker när eleverna erhåller kunskaper som undervisats, alltså den lärda kunskapen. Den kunskapen kan alltså, enligt teorin, skilja sig från kunskapsstoffet i de tidigare faserna av transpositionsprocessen.



Figur 1. Schematisk bild över transpositionsprocessen och de fyra faser som kunskap genomgår (fritt tolkat och översatt från Chevallard och Bosch, 2013)

Den här studien kommer främst undersöka transponeringen mellan fas 2 och fas 3, då lärarna utgör källan för kunskap. Till viss del kommer även transponeringen mellan steg 3 och steg 4 att uppmärksammas men då utifrån lärarnas synvinkel och inte genom att eleverna testas på vad de faktiskt har lärt sig.

3.2 Instrumentell ansats

Den instrumentella ansatsen (eng: *instrumentall approach*) uppkom i samband med att tekniska, datoriserade redskap började användas i matematiken, och därmed i matematik-undervisningen, under senare delen av 1900-talet (Trouche, 2005a). Guin och Trouche (1998) refererar till Artigue som förklarar att ansatsen bygger på konstruktivistiska tankesätt om att utövaren (även kallat *subjektet*) spelar en avgörande roll för kunskapsbildande hos den själv. Den andra avgörande rollen i kunskapsbildande spelas av själva objektet, som ska tillåta subjektet att uppnå sitt mål (Verillon & Rabardel, 1995).

I Frankrike började forskare undersöka elevernas användande och inlärning, vilka fallgropar som finns och vad lärare kan göra för att transformera en artefakt till ett matematiskt verktyg för eleverna (Trouche, 2005a). Symbolhanterande miniräknare är det som främst har varit föremål för studier med en instrumentell ansats i tidigare år men nu kan även programmering vara aktuellt att undersöka utifrån denna ansats.

Guin och Trouche (1998) understryker att behärska ett matematiskt instrument kräver en komplex *instrumentationsprocess.* Den här processen sker inte automatiskt, även om studenter snabbt kan ta till sig ett nytt verktyg. Det finns en risk att subjekten använder en artefakt utan att reflektera kring vad de gör och då kan deras matematiska förståelse snarare skadas än förbättras. Samtidigt visar författarna resultat från studier inom avancerad matematik, där datorer har förstärkt meningsfullheten hos användare, i och med möjligheten att visualisera matematik i andra miljöer. Guin och Trouche kommer till slutsatsen att hur undervisningen bedrivs måste noga planeras av läraren, för att elevernas matematiska kunskaper ska kunna utvecklas. Eleverna kan ha svårt att se matematiska anknytningar utanför den tekniska microvärlden och läraren spelar då en avgörande roll i att relatera handlingarna som görs i datorn till matematiken.

3.2.1 Verktyg, artefakt eller instrument

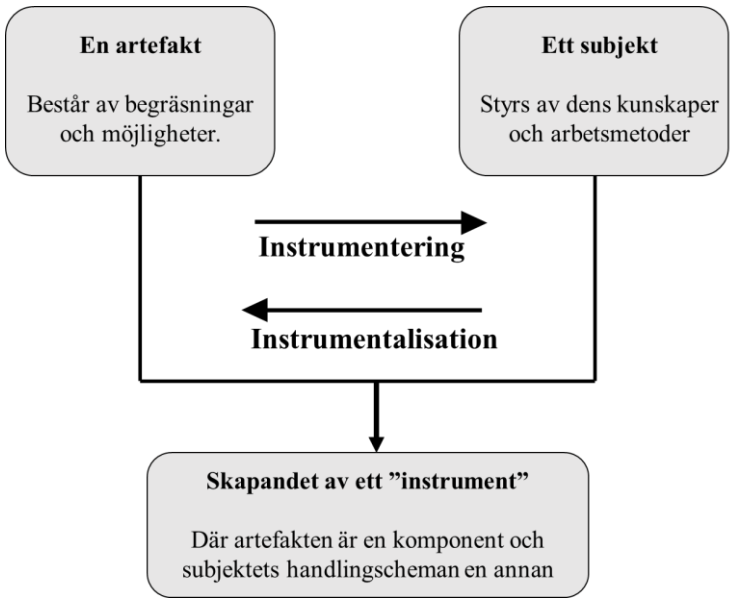
För att skapa en bättre förståelse för den instrumentella ansatsen, är det relevant att särskilja de olika begreppen verktyg, artefakt och instrument. Trouche (2004) använder begreppet ”verktyg” för att beskriva något som människor skapat för att nyttja med ett bestämt syfte. Däremot menar Trouche att dessa inte måste vara fysiska ting, utan även språket ses som ett verktyg. Skulle man däremot syfta till själva saken, utan att ta hänsyn till dess

användningsområde eller i vilket syfte den används, används begreppet ”artefakt”. Artefakter är enbart det fysiska objektet, skapat av människan, medan ett ”instrument” är en kombination av själva artefakten och av en psykologisk komponent. (Verillon & Rabardel, 1995). Vidare menar Verillon och Rabardel att ett instrument aldrig kan existera i sig själv. Ett instrument snarare skapas när subjektet lyckats integrera en artefakt i sin aktivitet på ett sätt så att artefakten hjälper subjektet att uppnå ett mål (Toruche, 2004).

När elever lär sig att använda symbolhanterande räknare, eller programmering, innebär det ofta att eleverna måste behärska flera olika instrument eftersom artefakterna innehåller flera olika verktyg (Trouche, 2005a). Hela artefakten blir inte ett instrument förrän subjektet behärskar alla dess delkomponenter. Däremot kan flera olika instrument utvecklas, utifrån hur uppgiften ser ut och vilket syfte artefakten används till. I grafräknare kan till exempel grafer studeras så väl som funktioner och det är möjligt att enbart behärska någon av dessa funktioner tillräckligt väl för att kalla det ett instrument.

3.2.2 Instrumentell genesis

Den komplexa process som krävs för att skapa ett instrument brukar forskare kalla för en *instrumentell genesis* (Verillon & Rabardel, 1995; Trouche, 2004). Som tidigare nämnt, består ett instrument dels av en psykologisk komponent, dels av en artefakt som komponent. Av den anledningen har Trouche (2004) valt att skilja på två processer i den instrumentella genesisen: en som riktar sig mot subjektet (*instrumentering*) och en som riktar sig mot artefakten (*instrumentalisation*) (Fig. 2). Dessa två processer verkar parallellt och är stundtals svåra att särskilja i en instrumentell genesis eftersom det ständigt finns en stark dialektik mellan subjektet och artefakten.



Figur 2. En schematisk representation av instrumentell genesis och hur intrumentalisation och instrumentering samverkar för att skapa ett instrument. (Fritt översatt från Trouche, 2005a)

Instrumentalisation är den process där subjektet till en början endast upptäcker artefaktens olika funktioner och slutligen organiserar och personifierar verktyget på ett sätt så att det passar

subjektet själv (Trouche, 2005a). Det kan liknas vid hur datoranvändare föredrar att organisera sitt skrivbord eller vilka funktioner som användaren väljer att spara i miniräknarens minne. Subjektet lär sig att utnyttja olika egenskaper hos objekten, ibland på ett annat sätt än vad skaparna av artefakten avsett (Trouche, 2004). Trouche tror även att hur subjektet uppfattar att skaparnas intention med ett visst verktyg var, kan styra hur den väljer att använda artefakten. Detta kan i så fall leda till att subjektet inte tillåter sig att gå vidare i processen i att göra artefakten till ett personligt verktyg. Instrumentalisationen kan alltså både utveckla artefakten och leda till dess försämring (Trouche, 2005).

Instrumentering är den andra delprocessen av den instrumentella genesisen som skapar en förändring hos subjektet genom artefaktens *begränsningar* och *möjligheter* (Trouche, 2004). Under den instrumentella genesisen kommer subjektet att skapa mentala *scheman* i samband med att artefakten används. När ett schema utvecklats har agenten nya kunskaper som gör att den kan utföra en viss handling (den *pragmatiska* funktionen av ett schema), förvänta och planera handlingen (den *heuristiska* funktionen) och förstå vad handlingen gör (den *epistemologiska* funktionen). Trouche väljer att skilja på två olika typer av scheman. De kan vara *användningsscheman* eller *instrumentella handlingsscheman.* Användningsscheman utvecklas när subjektet lär sig att hantera och memorera artefaktens knappar eller funktioner medan instrumentella handlingsscheman är förståelse för hur artefakten kan användas för att lösa en uppgift. Teorin om scheman har först utvecklats av Piaget men har senare omdefinierats av Vergnaud och det är Vergnauds definition som även Trouche använder sig av. Det kan vara svårt att studera scheman i samband med elevernas kunskapsutveckling eftersom det är en mental process som endast visar sig i form av *gester* (Trouche, 2004).

3.2.3 Instrumentell orkestrering

Att förvandla en artefakt till ett pedagogiskt och kognitivt instrument, kan ta tid och sker inte av sig själv. För att kunna använda ett verktyg som ett medel för att uppnå ett matematiskt mål, krävs det att studenter lär sig matematiskt innehåll från olika miljöer. Det kan vara genom text, beräkningar för hand och grafräknare eller programmering (Guin & Trouche, 1998). Trouche (2004) menar att läraren har en viktig roll i att artikulera en artefakts begränsningar och möjligheter, observera elevernas gester och knyta samman verktyget med matematiska koncept. Hur läraren väljer att göra detta kallas för *instrumentell orkestrering.*

Trouche (2004) ser instrumentell orkestrering som en nödvändig aspekt av att bygga ett instrument, i alla fall när det görs i skolundervisningen. Trouche förklarar begreppet som ”den externa styrningen av elevers instrumentella genesis” (Trouche, 2004, s. 296, egen översättning) och anser att det ofta saknas i debatten kring digitala hjälpmedel i undervisningen. Det för bort fokus från själva artefaktens utformning och didaktiska förklaringsmodeller som läraren har och betraktar i stället organiseringen av klassrummen. Drijvers m.fl. (2010) förtydligar att det är en social process att utveckla scheman och skapa ett instrument. Skapandet sker i en klassrumskontext med en kollektiv process av instrumentell genesis, något som läraren ska vara medveten om och dra nytta av i undervisningen.

Orkestreringen ska definieras av *didaktiska gestaltningar*, alltså hur läraren väljer att presentera och placera artefakten i omgivningen beroende på vilken nivå av det matematiska innehållet som behandlas (Trouche, 2004). Hur läraren väljer att använda och utnyttja de tillgängliga gestaltningarna kallas för *användningsformer* och kan ses som delmål av hela orkestreringen. Vidare väljer Trouche att dela upp instrumentell orkestrering i olika nivåer där den första nivån berör själva artefakten, den andra nivån är psykologisk och behandlar *instrumentet* och den

tredje nivån berör *relationen* mellan artefakten, dess användning och subjektet. Vid undervisning med avancerade, tekniska föremål bör orkestreringen se olika ut vid de olika nivåerna.

3.2.3.1 Exempel på de olika nivåerna av orkestrering

Undervisning av den första nivån med instrumentell orkestrering syftar till att hjälpa eleverna förstå hur mjukvaran i artefakten fungerar. Trouche (2005b) ger ett exempel på hur en lärare visar och beskriver de olika verktygen en symbolhanterande räknare har för att beräkna gränsvärden. Läraren kan uppmärksamma eleverna på vilka begränsningar som finns hos räknare, hur de kan modifiera dem till att passa sina behov och ge dem en guide på hur räknaren kan användas vid gränsvärdesberäkningar.

Guin och Trouche (1998) berättar om ett forskningsexperiment där lärare bedrev undervisning med grafräknare utifrån instrumentell orkestrering på den andra och tredje nivån. Eleverna i studien var mellan 17 och 18 år och målet med lektionerna var att eleverna skulle lära sig hantera en grafritande räknare. Vid orkestrering på den andra nivå undervisar läraren räknarens funktioner och användningsområden genom att använda tavlan, samtidigt som en av elevernas räknare visas på helskärm för resten av klassen. Studenten som styr den kallas en "sherpa-student" och vem som får den rollen varieras vid varje lektion. Klassen antecknade vad läraren sa i en anteckningsbok samtidigt som de följde lärarens anvisningar på deras egna räknare. Läraren kan då se hur sherpa-studenten uppfattar hens instruktioner och upptäcka eventuella missförstånd. Detta var ett sätt att bryta de traditionella rollerna i ett klassrum och skapa en mer social klassrumsmiljö men forskarna menar att det förutsätter att läraren har en god förståelse för artefakten.

Efter genomgång av redskapet fick eleverna matematiska problem som de kunde jobba med i grupper om två eller tre elever. Här gick undervisningen över till att bedriva instrumentell orkestrering på den tredje nivån (Guin & Trouche, 1998). Grupperna fick ta hjälp av räknare, penna och papper och sina anteckningar för att lösa problemet. Under tiden som eleverna arbetade med det agerade läraren handledare. Eleverna skulle skriva och lämna in en rapport över hur de gick till väga när det tog sig an problemet. Syftet med rapporten var dels att få eleverna fokuserade på matematiken genom att de fick skriva ner alla steg de tog under tiden de arbetade, dels kunde läraren enklare följa eleverna instrumentella genesis. På så sätt skapar eleverna en relation till hur matematiken samverkar med artefakten.

Vilken nivå av orkestrering som lektionen ska innehålla, beror dels på vad syftet med lektionen är, dels på själva artefakten (Trouche, 2004). Från exemplen ovan kunde forskarna se att eleverna uppskattade att arbeta tillsammans i sociala miljöer och att få se manipulationer av artefakten på helskärm. Forskarna kunde även se hur tekniska hjälpmedel på ett sätt kan dölja att en elev inte har förstått ett visst matematiskt koncept men på ett annat sätt så kan de också avslöja sådana brister. Sammanfattningsvis konstaterade forskarna att effektiva tekniker på räknaren behöver kombineras med arbete med penna och papper för att eleverna ska kunna passera upptäckningsfasen och göra artefakten till ett instrument (Guin & Trouche, 1998).

4. Metod

I det här avsnittet kommer valet av datainsamlingsmetod, urval och analysmodell att beskrivas. De forskningsetiska krav som ställts på studien kommer att redogöras för samt hur olika val tros har påverkat trovärdigheten av studien. Vidare kommer dessa val att diskuteras mer kritiskt under avsnitt 6.1.

4.1 Datainsamlingsmetod

För att uppnå studiens syfte har en kvalitativ metod valts. Dalen (2015) beskriver hur målet med kvalitativa studier är att få en insikt om personens sociala verklighet och hur de anpassar sig till sin livssituation. Det målet stämmer väl med studiens mål, då syftet är att få en insikt om hur undervisning med programmeringsmoment bedrivs. Lärare är de mest lämpade att besvara frågor om de didaktiska avvägningar som görs i klassrummen. Genom intervjuer kan de uttrycka sina tankar och upplevelser kring införandet av programmering, styrdokumentens formuleringar och Skolverkets intentioner. Intervjuerna ger också en möjlighet att utforska elevernas upplevelser, dock sett från lärarnas synvinkel. Tack vare intervjuer från lärarnas perspektiv kan då fas 2, 3 och 4 i den didaktiska transpositionsprocessen analyseras (se Fig. 1). Det var dessa tre faser som även utgjorde de tre teman som frågeställningarna och intervjuguiden utgick från.

För studien har en semistrukturerad intervjumetod använts. Semistrukturerade intervjuer lämpar sig för oerfarna intervjuare och när samtalen riktar sig mot förutbestämda teman (Dalen, 2015). Vid den här typen av intervjuer tillåts informanterna att sväva ut i sina svar eftersom frågorna anses vara av öppen karaktär, samtidigt som intervjuaren kan styra samtalet mot studiens olika teman genom att frångå intervjuguiden och ställa följdfrågor. Detta gör att intervjuerna kan variera i längd och omfång, då informanten kan vara hur utförlig den vill i frågorna.

4.2 Utarbetning av intervjuguide

Intervjuguiden ska omvandla studiens frågeställningar till olika teman (Jacobsson & Skansholm, 2019). Intervjuguiden ska sedan fungera som ett redskap under intervjun för att se till att varje tema avhandlas uttömmande. Temana ska ha en tydlig anknytning till den teori som har presenterats i tidigare avsnitt av studien.

Genom att noggrant studera tidigare forskningsresultat som är relevanta för studien, uppkom förslag på frågor till intervjuguiden. Vid bearbetning av frågorna kunde de sedan sorteras till olika teman. Valet av teman föreföll sig naturligt utifrån Chevallards ramverk om didaktisk transposition och blev därför styrdokumenten, undervisningen och elevernas lärdomar. Dessa teman motsvarar fas 2, 3 och 4 i Chevallards transpositionsteori och gör det möjligt att besvara de frågeställningar som presenterats under studiens syfte. För att kunna analysera undervisningen utifrån en instrumentell ansats, skulle temat kring undervisningen grunda sig i undervisningsmaterial som lärarna ombads att ta med till intervjun (se bilaga 1).

Utöver dessa tre teman innehöll intervjuguiden även en bakgrundsdel, där frågor kring lärarnas utbildning och arbeten besvarades. Anledningen till att fråga om lärarnas bakgrund under intervjun, i stället för att erhålla informationen över till exempel mejl, är för att Dalen (2015) förespråkar intervjuguider med lättare frågor inledningsvis, den så kallade ”områdesprincipen”. Det är för att värma upp informanterna till kommande frågor som är mer centrala för arbetet

och som kan vara av mer känslig karaktär. Efter att frågorna sorterats efter tema utfördes ett redigeringsarbete i samråd med handledare. En del frågor omformulerades för att undvika mångtydigheter och andra frågor ströks då de inte bidrog till svar på studiens frågeställningar. Innan det första utkastet av intervjuguiden var färdigt skrevs en kort inledning till intervjuerna där studiens syfte presenterades, samtyckeskravet behandlades och kraven om konfidentialitet och information förmedlades.

För att kunna uppskatta tiden för intervjun och för att testa intervjufrågorna, hölls en pilotintervju med en medstudent över Zoom. Pilotintervjuer har som syfte att upptäcka eventuella teman som ej behandlas, frågor som är otydliga men även att träna rollen som intervjuare (Jacobsson & Skansholm, 2019). I och med att ett av intervjuguidens teman grundade sig i material som läraren tar med sig, kunde denna del av intervjun inte testas. Däremot gav pilotintervjun en uppfattning om de andra temanas tidsomfång och även hur frågorna upplevdes. I samråd med den frivillige lärarstudenten omarbetades vissa frågor och följdfrågor som ansågs relevanta skrevs ner. Därefter ansågs intervjuguiden färdig (se bilaga 2). Pilotintervjun gav också en möjlighet att testa de tekniska hjälpmedel som skulle användas vid intervjuerna. Ljudupptagningen från programmet Zoom hade lägre kvalité än det externa inspelningsverktyget och stundtals var det svårt att uppfatta vad informanten sa. Det ledde till att dubbla inspelningsverktyg skulle användes vid digitala intervjuer för att inte gå miste om information.

4.3 Urval

Eftersom resultatet från en kvalitativ undersökning ej kan generaliseras, är det vanligt att urvalet inte är lika omsorgsfullt uttänkt som vid kvantitativa undersökningar (Dalen, 2015). Detta i kombination med bristande kontaktnät i forskningsfältet ledde till att informanterna valdes utifrån ett ”kriterieurval”. Dalen bedömer det som ett bra urval för mindre erfarna forskare och baseras på att de valda informanterna endast behöver uppnå vissa kriterier för att kunna delta. Kriterierna som var nödvändiga för att delta i studien var att den tillfrågade läraren någon gång sedan revideringen av kurs- och ämnesplanerna har undervisat programmering och att de undervisar, med behörighet i matematik, på antingen högstadiet eller gymnasiet. Eftersom intervjuerna kunde hållas digitalt om informanterna önskade, var inte studien begränsad till att enbart innehålla informanter från Göteborgsregionen.

Av de sju lärare som tillfrågades tackade fyra ja till att ställa upp på intervju. De tillfrågade valdes ut av bekvämlighetsprincip, det vill säga att de låg nära till hands (Bryman, 2002). Av de tillfrågade lärarna hade två inte undervisat i programmering, och lämpade sig därmed inte för studien, och en tackade nej. Detta resulterade i att tre grundskollärare och en gymnasielärare ställde upp i intervjuer. När de tillfrågade tackade ja, skickades ett missivbrev ut med ytterligare information om intervjun och om studien (se bilaga 1). Deras arbetserfarenhet av läraryrket varierade från drygt ett år upp till över 30 år och flera lärare hade andra uppdrag på skolan utöver lärarrollen, som matematikutvecklare och förstelärare till exempel. Av de som intervjuades var två kvinnor och två män och tre arbetade inom Göteborgsregionen medan en arbetade i region Skåne.

4.4 Genomförande

Vid mailkontakt fick informanterna ge förslag på tider som passade dem och själva besluta om de ville genomföra en digital intervju eller mötas personligen på en lämplig plats. En av

intervjuerna hölls digitalt via Zoom, två av dem hölls på informanternas arbetsplatser och en i informantens hem. Längden på intervjuerna varierade från 30 upp till 45 minuter.

Intervjuerna inleddes med att informera deltagarna om studiens syfte och de forskningsetiska principer som studien grundar sig på. När informanterna tagit del av informationen, accepterat format för ljudupptagning och gett sitt samtycke till att medverka kunde själva intervjun inledas. Intervjuaren höll en låg profil under intervjuerna, något som informanterna hade förvarnats om i inledningen. Detta dels för att underlätta transkriberingsarbetet, dels för att inte påverka informanterna. Även pauser välkomnades, då det ger informanten tid att tänka över sina svar. Vid de tillfällena är det extra viktigt att intervjuaren inte bryter tystnaden (Dalen, 2015).

Transkriberingen av ljudfilerna påbörjades så tätt inpå intervjuerna som möjligt. Under transkriberingen fördes även minnesanteckningar som stöd vid analysprocessen. Intervjuerna transkriberades ordagrant och även stakningar och upprepningar transkriberades för att få en bättre känsla av informantens mening bakom uttalandet. Däremot transkriberades inte bekräftande uttalanden som bekräftade informanterna som till exempel “mmm” och inte heller utfyllnadsord som “eh”, eftersom det hade gjort materialet mer svåröverskådligt.

4.5 Forskningsetik

Vid första mailkontakten med lärarna förmedlades att deltagande i studien var helt frivilligt. De lärare som tackade ja till att delta erhöll information om studien i missivbrevet. Där beskrevs studiens dess syfte, metod och huvudman samt uppskattad tid för intervjun (se bilaga 1). Efter att informanten mottagit brevet, kontaktas den igen för att komma överens om en tid och plats för intervjun. Vid intervjutillfället fick informanterna ta del av informationen enligt 16 § (SFS, 2003:460) igen, innan de gav sitt samtycke.

Informanternas personuppgifter har inte röjts någon gång under arbetet, utan har behandlats med största möjliga konfidentialitet. Resultaten som valts ut i studien går inte att härleda till en viss person och är inte av känslig natur så att utsatta grupper skulle kränkas. Uppgifter och material från studien, som inte presenteras, kommer inte heller att delas till obehöriga.

4.6 Trovärdighet

För att säkra studiens trovärdighet har den utförts med begreppen *validitet* och *reliabilitet* i åtanke under hela processen. Det innebär att alla överväganden och beslut har varit transparenta samt att studien undersöker det som den avser i syfte och frågeställningar.

4.6.1 Validitet

Jacobsson och Skansholm (2019) förklarar att begreppet validitet handlar om vad studien undersöker. Det som undersöks ska vara relevant för att kunna besvara forskningsfrågorna och för att göra det har metodval och teoretiska utgångspunkter motiverats och redogjorts för. För studien har det varit nödvändigt att definiera begreppet programmering för att läsaren ska förstå vad som undersöks. Även kriteriekurvalet har påverkat studiens validitet positivt eftersom syftet är att undersöka programmeringsundervisning och deltagarna hade alla hållit i sådan undervisning samt erhöll lärarlegitimation i ämnet matematik. Deltagarna fick svara på öppna frågor utan att intervjuaren påverkade dem, vilket styrker deras uttalanden. För att stärka validiteten har studien varit transparent i de val och överväganden som gjorts och har en logisk struktur i hur detta presenteras.

4.6.2 Reliabilitet

För att en studie ska hålla god reliabilitet ska mätningarna ske på ett tillförlitligt sätt (Jacobsson & Skansholm, 2019). För kvalitativa studier kan det innebära att genomförandet av intervjuerna inte sker under undermåliga förhållanden och att alla som medverkar i studien är trygga i sina respektive roller. Genom att informanterna fick bestämma plats och medie för intervjun, garanterades deras trygghet samtidigt som att valfriheten att ställa upp betonades vid den första kontakten så att lärarna inte kände sig tvingade till att delta.

4.7 Analys

För studien har en teoridriven tematisk analys valts (Braun & Clarke, 2006). Denna metod passar bra in på studien eftersom den med lätthet kan anpassas på de ramverk som studien grundar sig på och lämpar sig för oerfarna forskare. Analysprocessen påbörjades redan under intervjun, där informanternas svar analyserades på plats och möttes upp med lämpliga följdfrågor (Dalen, 2015). Detta kan ses ingå i vad Braun och Clarke (2006) anser vara steg ett i analysfasen, nämligen att lära känna materialet. Här ingår även transkriberingsprocessen som Dalen (2015) uttrycker som tidskrävande men nyttig för att vidare bekanta sig med den insamlade datan. Under tiden som transkriberingen ägde rum, fördes även anteckningar över intressanta uttalanden i likhet med Braun och Clarkes (2006) råd för denna fas. Efter att ha läst om materialet igen efter transkriberingen har även fler anteckningar tillkommit.

I steg två av analysen har materialet kodats och organiserats (Braun & Clarke, 2006). Kodningen skedde genom att intressanta citat valdes ut från en informant och ett tema av intervjuguiden i taget. Varje citat skrevs in i en tabell och kodades med kännetecken för just det uttalande. När det behövdes sattes även citaten in i en kontext för att inte meningen av dem skulle bli missförstådd. I den här fasen var citaten var långa och oredigerade.

I steg 3 betraktades koderna utifrån de tre teman som studien tog avstamp i. Temana var grovt benämnda *styrdokumenten, undervisningen* och *elevernas lärdomar* i den här delen av analysen. I likhet med hur Braun och Clarke (2006) beskriver det här stadiet, har även här koderna sorterades in i underteman till de tre huvudteman genom att göra mind-maps över varje tema. Samtliga koder fick en plats vid varje tema, även om det fanns koder som inte kunde länkas till varandra. Det var först i det fjärde steget som underkategorier valdes ut efter vilka som ansågs besvara studiens forskningsfrågor bäst. Genom en iterativ utprövning av möjliga sätt att sortera koderna, beslutades tillslut tre undertema till varje huvudtema som relaterade till tillhörande forskningsfråga och Chevallards ramverk. Steg fyra och fem i Braun och Clarkes tematiska analysmetod har gått in i varandra eftersom namngivningen av temana skedde parallellt med att olika underteman valdes ut.

I det sjätte och sista steget av analysprocessen valdes relevanta citat ut från varje tema (Braun & Clarke, 2006). Eftersom citaten som valdes ut skulle representera hela studiens resultat, var det viktigt att urvalet gjordes med omsorg vilket gjorde det här steget till ett tidskrävande moment. Citaten som valdes ut skulle måla en rättvis bild av informantens utsaga, samtidigt som det skulle vara lättläst och lättöverskådligt. Av den anledningen redigerades citaten och upprepningar och tilläggsord sorterades bort. Överflödig och orelevant information klipptes bort från citaten och ersattes med notationen [...] men inte på ett sätt som drabbade citatets innebörd utan snarare förstärkte den. Resultatet presenteras genom att kombinera nyckelcitat med text som sätter citatet i ett sammanhang. I texten redovisas också resultat som var relevanta för studien men som saknade målande citat.

5. Resultat

Nedan följer resultatet av den tematiska analysen av intervjuerna. Inledningsvis presenteras alla lärare med den bakgrundsinformation som gavs under intervjun. Utifrån Chevallards ramverk har tre huvudkategorier skapats. Det är *lärarnas syn på styrdokumenten*, *undervisning med programmeringsmoment* och *elevernas lärdomar*. Under varje huvudkategori tillkommer tre underrubriker utifrån likheter och skillnader i lärarnas uttalanden. Citaten i det här avsnittet är redigerade för at öka läsbarheten, mer om detta går att läsa i avsnitt 4.7.

5.1 Bakgrund av informanterna

Vid intervjutillfället fick informanterna besvara frågor om de själva och deras programmeringsbakgrund. Svaren på dessa har sammanställts nedan tillsammans med övrig information om dem som ansågs vara relevant.

L1: Läraren har jobbar som lärare i 14 år med behörighet i matematik och fysik och har enbart undervisat på gymnasiet. Idag arbetar hen på naturvetenskapliga programmet på en skola i region Skåne och undervisar kurserna Ma1c, Ma3c och Fy2. Hen undervisar även i grundskolematematik på introduktionsprogrammet för elever som inte varit i Sverige så länge. Läraren tillhör även en grupp av förstelärare på skolan samt är ämnesansvarig i fysik. Läraren uttrycker en viss oro kring att undervisa programmering och anser sig inte ha tillräckliga kunskaper om det, trots att hen har läst en kurs i ”beräkningsprogrammering” på högskolan och gått en kurs i programmet GeoGebra. Inför införandet hade kommunen även anordnat en kurs i programmering för alla matematiklärare för att förbereda dem för uppdraget. Kursen bestod av fem träffar varav de tre första enbart behandlade programmet GeoGebra. Läraren ansåg sig inte lära sig något nytt under de träffarna, utan fick snarare agera hjälplärare åt de andra på kursen. Under de två sista träffarna blev lärarna introducerade till, och fick arbeta i, Python men även här kunde läraren inte se att det skedde en särskilt stor kunskapsmässig vinning, då tiden var för knapp. Fortbildningsarbetet skulle fortsätta på skolan men läraren vittnar om att det inte blev av i den utsträckning som var planerat, då mer akuta ärenden tog upp planeringstiden.

L2: Läraren har jobbat som lärare i över 30 år och undervisar på högstadiet med behörighet i matematik och NO-ämnena. I dagsläget undervisar hen en åttondeklass och en niondeklass i både matematik och NO och utöver lärarrollen arbetar hen även som matematikutvecklare. Lärarens programmeringsbakgrund började redan i slutet av 70-talet, under tiden som hen läste teknisk fysik på Chalmers. Då lärde sig läraren att programmera i Fortran och i Pascal. Utöver det gick hen en programmeringskurs runt 1997 och en kurs i programmering för lärare på 7.5 högskolepoäng som Göteborgs kommun betalade för i samband med införandet av programmering i styrdokumenten 2018. Läraren tyckte att kursen förberedde hen väl för att undervisa i det textbaserade språket Python, som kursen riktade sig mot. Läraren uttrycker stor entusiasm över att programmering nu är en del av undervisningen men har inte undervisat det i matematiken det senaste året utan enbart i teknikämnet.

L3: Läraren tog examen från ämneslärarprogrammet i januari 2020 och har sedan dess arbetat som lärare på högstadiet i matematik, fysik, biologi och kemi i Göteborgsregionen. Idag undervisar hen i årskurs 6 och 8 och har inga andra ansvarsområden på skolan utöver lärarrollen. Läraren var student när införandet började gälla och tycker inte att lärarutbildningen bidragit till hens programmeringskunskaper, utan ansåg att allt prat kring programmering utan att

faktiskt lära ut det mest skrämde upp studenterna. De programmeringskunskaper som läraren har idag, kommer från att hen har suttit och arbetat med det själv på sin fritid. På hens arbetsplats har det inte heller skett någon vidare fortbildning sen läraren började där och kollegorna uppvisar ett svagt intresse för det. Det har snarare blivit så att läraren har gjort uppgifter och sedan delat dessa med resterande matematiklärare. Läraren känner att den behärskar Python ganska väl men kan inte andra programmeringsspråk, vilket gör att Python blir ett naturligt val att undervisa programmering i.

L4: Läraren är inne på sitt 26:e läsår som lärare med behörighet i matematik och NO för årskurserna 4 till och med 9. Under sina år som lärare har hen även fått kombinera rollen som undervisande lärare med att vara biträdande rektor under ett år och tillförordnad rektor ett annat. Idag undervisar hen matematik och NO i en åttondeklass och en niondeklass. Läraren jobbar även med IKT på skolan och har hand som elevrådet, utöver sin roll som lärare. När läraren själv gick i skolan var programmering en del av undervisningen och har erfarenhet av BASIC-programmering på Compisdatorer samt Pascal och HTML-programmering för att nämna några exempel. Kunskaperna var inte så färska inför det nya införandet av programmering 2018 men hen kunde enkelt fräscha upp kunskaperna med hjälp av den 7.5-poängs distanskurs i Python som anordnades av kommunen i samband med styrdokumentens förändring. Läraren tyckte att kursen gav hen bra förståelse för hur Python fungerar men inte om hur lärare ska anpassa det på matematikundervisningen. Läraren tycker att hen behärskar programmering tillräckligt väl för det som förväntas av hen men har det senaste läsåret enbart programmerat på tekniklektionerna.

5.2 Tema 1: Lärarnas syn på styrdokumenten

För att kunna identifiera hur kunskapsstoffet förändras i Chevallards (2006) olika faser, har lärare fått besvara frågor som belyser deras syn på införandet av programmeringsmomenten och hur de förhåller sig till de nya skrivningarna i styrdokumenten. Inledningsvis presenteras resultatet av lärarnas förståelse för sitt uppdrag i relation till styrdokumenten och Skolverkets mål (se avsnitt 2.1). Det följs av en redogörelse för hur lärarna definierar och skiljer på begreppen programmering och digitala verktyg. Slutligen sammanställs citat som beskriver hur lärarna ser på möjligheten att lära ut programmering som ett matematiskt verktyg.

5.2.1 Förståelse för sitt uppdrag

Lärarnas uppgift är att omsätta styrdokumenten i praktiken och att bedriva en likvärdig undervisning utifrån de kunskaper som Skolverket beslutat för att ingå. För att detta ska ske behöver lärarna förstå Skolverkets intentioner med införandet av programmering och tolka skrivningarna om programmering på ett korrekt sätt.

Samtliga lärare som medverkade i studien verkade ha god förståelse för Skolverkets motiv till införandet, även om de endast resonerade kring vad det kunde bero på utan att med säkerhet veta anledningen. Samtliga lärare nämner någon gång samhällsnyttan i att lära ut programmering och två lärare resonerar som följer:

> Man hör ju att det blir vanligare i fler och fler yrkeskategorier till exempel. Därför försöker de nog få in det i skolan på något sätt. Men ja, jag har nog inte funderat så mycket över det. (L3)

> Allting runt omkring oss är ju programmerat och vi styrs ju av datorer och mobiler och allting som är programmerat och det kan väl vara bra att ha någon form av kännedom om det. Sen kommer kanske inte alla att bli programmerare eller jobba med det. (L1)

En annan lärare visar också på en förståelse för införandet men lägger större emfas vid att tidigt i livet skapa ett intresse av programmering. Läraren uttalar sig på följande sätt:

> Att man har programmering på gymnasiet är väl ganska självklart eftersom det är många som kommer jobba med det. Då måste man nästan ha börjat på grundskolan för att eleverna ska veta om det är något som de kommer tycka är intressant eller roligt. (L2)

Förståelsen för varför programmering har blivit en del av undervisningen i skolan, verkar inte vara ett bekymmer bland lärarna i studien. Samtliga lärare nämnde även någon gång under intervjun, den naturliga kopplingen som de ansåg finnas mellan programmering och matematik. En lärare beskrev sambandet på följande sätt:

> Det är en annan synvinkel på matematiken. Du får ännu en djupare förståelse av matematik genom att programmera. Det tror jag (L2)

Trots att lärarna visade förståelse för programmeringens del i skolan, hade de problem med hur införandet av det hade ägt rum. Två av lärarna, L3 och L4, förstod inte poängen med att ha det både i matematikämnet och i tekniken och en av dem säger:

> Jag tror att om man hade valt att lägga det under ett ämne, till exempel i teknik där det kanske egentligen mer hör hemma [...]. Då hade man kunnat välja vad det är eleverna ska ha ett hum om, vad ska de kunna mer ordentligt och rent praktiskt programmera, när de lämnar lågstadiet. Var vill vi egentligen att de ska stå när vi lämnar högstadiet? (L4)

Att inte förstå vad målet med programmeringen är, och därmed inte vara helt införstådd i ens uppdrag, återspeglar sig i en annan lärares syn på sin egen kunskap om programmering. När läraren fick frågan om hen ansåg sig ha tillräckliga kunskaper för att undervisa i programmering gav den detta som svar:

> Det beror helt just eftersom jag har svårt att förstå vad kraven på mig är, hur mycket jag ska lära ut och hur mycket jag ska kunna. Jag klarar av att göra en del saker, jag klarar inte av att göra allt. För det jag gör har jag tillräckliga kunskaper men de har jag också fått lära mig själv på egen tid. (L3)

Att inte ha tillräcklig kunskap i programmering är en aspekt av införandet som alla lärare förutom L4 ser som ett bekymmer. L2 anser sig själv ha tillräckliga kunskaper men har mött lärare som uttrycker en oro kring att undervisa i det. L2 understryker också hur fortbildningsarbetet borde ha prioriterats på skolan, något som L1 också påpekade. L1 sammanfattar här sin inställning till hur det har implementerats:

> Jag är egentligen inte emot införandet men när helt plötsligt alla Sveriges mattelärare ska göra det och de allra flesta mattelärarna inte har några kunskaper i det, då blir det lite klumpigt eller konstigt (L1)

Det är inte bara målet med programmeringsundervisningen som visade sig svår att förstå, utan även själva skrivningarna om programmering. Styrdokumenten specificerar inte vilka språk som ingår i programmering och detta menar en lärare kan göra att uppdraget tolkas olika.

> Sen omfattningen eller vilka program eller språk det gäller, är ju otydligt då eftersom vi kan tolka det olika. Vissa kan ju säga att de har använt den grafritande räknaren och då har de ju programmerat! (L1)

Omfånget och hur mycket tid som ska läggas på just programmering är något som blir tydligt under intervjuerna att lärarna har svårt att förstå utifrån styrdokumentens formuleringar. L2 och L4 har under det senaste läsåret endast undervisat programmering i teknik och L1 har också nämnt att det kan bli ett moment som prioriteras bort, särskilt när man tar hänsyn till det gånga pandemiåret.

5.2.2 Definition av programmering och digitala verktyg

Eftersom programmering står tillsammans med digitala verktyg i ämnets syfte för grundskolan och i centralt innehåll för de flesta matematikkurser på gymnasiet, är det intressant att se hur lärarna skiljer på och definierar dessa begrepp. När lärarna blev utfrågade om skillnaden blev de ofta ställda och det var ingen som tydligt kunde redogöra för vad skillnaden var. Lärarna blev aldrig explicit tillfrågade att definiera vare sig programmering eller digitala verktyg men tre av dem gav ändå sin syn på vad programmering är för dem. En hade en väldigt bred definition av det:

> Jag tror också att programmering är ju ett sätt att tänka. [...] Jag måste liksom börja tänka som en dator. (L2)

Det hänger till viss del ihop med hur en annan lärare definierar det, fast med mer uttalad koppling till just algoritmer.

> Jag tänker att programmering är när man börjar komma in på att arbeta med algoritmer. Då ser nog jag det som att man börjar programmera. (L3)

En av lärarna skiljer sig en aning i sin definition och betonar generaliserbarheten och själva uppmaningen att få ett program att göra något.

> För mig är det nog lite mer programmering när man skriver in kommandon eller formler eller så [...] Man ber programmet, oavsett vilket program det är, att utföra en handling kan man säga (L1)

L1 och L3 tog även upp om huruvida kalkylark räknas till programmering eller ej. Båda lärarna ansåg att det klassas som programmering. Däremot sa resonerande en av dem kring verktyget GeoGebras tillhörighet på följande vis:

> Man kan ju använda kalkylark i GeoGebra också och då kan man ju typ programmera i det men annars så ser jag väl det kanske mer som ett digitalt verktyg. Å andra sidan gick jag som sagt en så kallad programmeringskurs där 3/5 var i GeoGebra. Men jag kanske egentligen ser det mer som ett digitalt verktyg. (L1)

Vad gällde begreppet digitala verktyg var lärarna mer överens och samtliga gav miniräknare eller grafritare som ett exempel. Även om lärarna verkade ha svårt att dra tydliga gränser mellan begreppen, försökte sig ändå en av dem på att definiera vad ett digitalt verktyg är:

> Det är lurigt. Jag skulle nog säga att ett digitalt verktyg, i sin enklaste definition, är någonting som är programmerbart. Det kan vara en sak man programmerar en gång och

> därefter är det fast eller något som är omprogrammeringsbart så att jag själv kan välja hur den ska vara. (L4)

5.2.3 Programmering som ett matematiskt verktyg

Lärarna tog ställning till möjligheten att undervisa programmering som ett verktyg för matematisk problemlösning, vilket är Skolverkets avsikt för programmering i de berörda åldersgrupperna. Alla lärare uttrycker vid någon punkt hur det är ett bra mål och en av lärarna uttrycker hur programmering kan vara en fördel för matematiken på följande vis:

> Att trycka på programmering som ett verktyg i form av att lösa problem eller använda vid upprepade beräkningar, ger mer förståelse för matematiken än om man bara skulle lära sig saker utantill. (L1)

Trots lärarnas, över lag, positiva inställning till att lära ut programmering som ett verktyg berättar de också hur det kan vara svårt att nå dit. En av lärarna delar med sig hur hen har tolkat styrdokumentens krav om programmering nedan:

> Eftersom det står under centralt innehåll ska eleverna få pröva på det och lära sig hur man kan använda olika saker i programmering. Jag försöker alltid tänka på att det är matematik vi ska lära oss och se till att programmeringen tjänar ett matematiskt syfte [...] och då tycker jag att jag har använt det på ett sätt som varit positivt. Ibland kan man se att programmeringen hamnar för långt bort från matematiken och då ser elever inte längre matematiken utan de ser bara programmeringen. Därför har jag medvetet hållit mig borta från att programmera spel till exempel. [...]. Det ska fortfarande vara kopplat till matematiken, det tycker jag är viktigt. (L3)

Det blev tydligt att flera lärare ser programmering som ett tidskrävande moment, framför allt om syftet är att det ska kunna användas som ett matematiskt verktyg. Inga lärare kan se att eleverna erhåller någon förkunskap i programmering när de ska börja undervisa om det, utöver några enstaka elever som har sysslat med det på sin fritid. Följande citat ger en inblick i hur två av lärarna ser på den problematiken:

> Än så länge känns det som att vi fortfarande behöver lära dem grunderna i programmering. Det kan vara så att jag skriver ett program som jag visar dem. […]. Men det handlar om en tidsbesparing, för tiden är viktig Vi har inte tid att sitta där i tio veckor medan de funderar på hur man löser det här programmet (L4)

> Det som är svårt är att man ofta kommer in på det här med hantverket programmering […] De kanske blir bättre när de har gått igenom det på lågstadiet och mellanstadiet. Om det nu blir så! Förhoppningsvis blir det så, men jag tror inte det. Om de kan lite och förstå lite av programmering, då kan man använda det som ett verktyg. Och det skulle vara jätteroligt! (L2)

I citatet ovan belyser L2 hur progressionen i programmering från låg-och mellanstadiet inte uppnås idag, något som andra lärare också har vittnat om. Läraren uttrycker en tveksamhet till om det faktiskt kommer uppnås, något som också L1 är tveksam till.

> Så det tar ganska många lektioner innan man kommer till den matten som man faktiskt skulle hålla på med, vilket gör att man kanske inte riktigt prioriterar det. (L1)

5.3 Tema 2: Undervisning med programmeringsmoment

Under det andra temat kommer lärarna ge en insikt om hur lektioner med programmering har bedrivits. Lärarna blev ombedda att ta med undervisningsmaterial som de har använt i sin undervisning, gärna i samband med problemlösning. L3 var den enda läraren som själv konstruerat den uppgiften som hen tog med till intervjun och L1 hade utgått från en uppgift i en annan mattebok än den som eleverna använder och modifierat den lite. Dessa två uppgifter gå att hitta under bilaga 3 och 4. L2 och L4 har använt sig av boken "Räkna med kod - programmering i matematik" från Sanoma Utbildning och tog med den till intervjun. Vilka metoder och val som lärarna gjort i samband med programmeringslektionerna kommer redovisas nedan.

5.3.1 Val av programmeringsmiljö

Eftersom Skolverket inte har specificerat vilka språk eller miljöer som programmering ska undervisas i, är det av intresse att ta reda på hur lärarna har resonerat kring detta. Samtliga högstadielärare har främst valt att programmera i språket Python under matematiklektionerna men L2 och L4 har även genomfört lektioner med blockprogrammering som Scratch och MakeCode. Grundskolelärarna har antingen inkluderat programmering i årskurs 8 eller årskurs 9.

Lärarna som programmerade i Python ansåg alla att textbaserad programmering kändes mer "på riktigt" jämfört med blockbaserad programmering. L3 trodde att eleverna tycker det är roligare när det känns äkta och L2 tyckte att det är ett bra språk för att lätt att se vad kommandona gör. Däremot kan blockprogrammering vara ett bra sätt att "komma in i tänket" i programmering, anser L2. Nedan följer ett uttalande från L4 om de olika miljöerna:

> Blockprogrammering är enklare att jobba med tillsammans med eleverna, jämfört med när vi använder textbaserad programmering. [...] Men med blockbaserade språk är det som att bygga pussel, det är lite slump var olika delar sitter och skulle något sitta på fel ställe säger programmet ifrån. Det är inte samma grej som när du skriver kod i textbaserat språk. (L4)

L1 undervisar i matematik på gymnasiet och har valt att inkludera programmering i Ma1c-kursen i samband med procent- och ränteberäkning. Läraren har då valt att använda Google Kalkylark som miljö. Läraren sammanfattar hens val på följande sätt:

> Innan jul jobbade vi med ränteberäkningar och då tycker jag det är rätt skönt med kalkylark på något vis. Vid upprepade procentuella förändringar kan man lägga till om det är aviavgifter och man får en struktur över det i ett kalkylark. [...] Man ska vara ganska van med Python eller liknande för att se strukturen i det. [...] Men har du ett Excel-ark eller kalkylark kan du få en annan struktur, en annan överblick. (L1)

5.3.2 Mål med lektionen/lektionerna

Målen med programmeringslektionerna varierade mellan lärarna. Både L1 och L3, som endast hade en respektive två schemabrytande lektioner med programmering, hade främst matematiska mål. Båda lärarna inkluderade programmering i undervisningen för att underlätta vid upprepade beräkningar, något som L4 också nämnt att hen gjort tidigare. L3 använder det även för att förtydliga det matematiska begreppet sannolikhet och berättar nedan hur hen förberedde klassen:

> Det här är en programmeringsuppgift som är till för att belysa vad sannolikhet är. Vi hade haft lite diskussioner innan om vad som menas med sannolikhet, att sannolikhet inte alltid är en sanning. Det är inte så att man ska få EN sexa om man slår tärningen sex gånger. Den här handlar lite mer om att eleverna ska få se vad det innebär. (L3)

Även L1 hade som mål att fördjupa förståelsen av ett begrepp som används i traditionell matematik men hade också mer allmännytta mål. Läraren sammanfattar det så här:

> Målet var ju att förstå ränteberäkningar eller räntebegreppen. Både upprepade (beräkningar) och förstå de olika delarna. Vad är amortering och ränta? Men kanske också att resonera lite kring hur mycket mer man betalar totalt när man har lånat pengar i förhållande till om man hade kunnat spara pengarna och betala kontant så att säga utan att låna pengar. […] Kanske inte så matematiskt det sista där. (L1)

L2 undervisade utifrån boken "Räkna med kod" parallellt med geometri och hade tydliga mål med vad skulle uppnå inom geometri medan målen med programmering var något hen inte hade tänkt över lika noga utan var mest nyfiken på hur det skulle gå att undervisa i det. Läraren berättar vad den önskar att programmeringen skulle generera för färdigheter som de kunde ha nytta av i matematiken:

> Ja, man vill ju att de ska kunna pröva och rätta och testa sig själva och få liksom ett resonemang kring sitt eget. "Vad var det som gick rätt och vad var det som gick fel, hur kan jag korrigera?" (L2)

L4 särskiljer sig lite från de övriga lärarna och ser mer hur hen vill ge eleverna några grunder i programmering och att de ska förmå att skriva egna, enkla program vid slutet av lektionerna. Läraren har valt att undervisa programmering i teknikämnet det senaste året och har använt cirka 12 lektionstimmar åt det. Läraren upplever att bedömningen är något som ställer till det för hens undervisning.

> Det är ett litet bekymmer i och med att det är ett moment som det står inte riktigt hur vi ska bedöma. Där är vi tillbaka på tydligheten, vilken nivå vill vi att de faktiskt ska ligga på när de lämnar olika delar av grundskolan. Så det är kanske en tydlighet som behöver bli bättre i kursplanen. (L4)

Även L1 och L3 pratar om bedömningen som en svårighet. L1 har valt att göra momentet till en inlämningsuppgift som eleverna antingen blir godkända på eller inte. L3 hade övervägt att ha uppgiften som en inlämningsuppgift men har i stället valt att hålla en helklassdiskussion i slutet av lektionen. Så här resonerar L3 kring bedömningen:

> Det här var en väldigt "icke bedömmig"-lektion, så att säga. Det handlade om att skapa en insikt hos eleverna, vad innebär det med sannolikhet? Vad är det vi gör? […] Det handlade mest om förståelse snarare än bedömning. Det var ett inlärningstillfälle. (L3)

5.3.3 Lektionernas upplägg

Lärarnas lektioner har inte bara olika mål utan även olika upplägg. Eftersom lärarna inte kunde se att eleverna hade någon förkunskap i programmering, krävde inte heller deras lektioner att de skulle kunna några kommandon på förhand. De två lärarna som inte baserade sina lektioner kring boken "Räkna med kod" hade endast matematiska kunskaper som förkunskaper i deras programmeringslektioner och en av dem beskriver förberedelserna så här:

> De kan ”sannolikheten är antalet gynnsamma utfall delat med antalet möjliga utfall”. Det är egentligen det enda de behöver veta. De har koll på lite vanliga sannolikheter, en sexa på en tärning är en sjättedel, singla slant är 50%. Och sen har de har de programmerat lite grann i Python, så de vet hur man startar programmet och kopierar och klistrar in och hur man kör programmet och så. (L3)

De två andra lärarna höll fler, sammanhängande lektioner som innehöll programmering och utgick från boken “räkna med kod” men kompletterade med några egna programmeringsuppgifter utöver de som fanns i boken. En av lärarna beskriver bokens upplägg på följande sätt:

> I och med att jag har använt den här boken ”räkna med kod” som mall så är målet ”här ska vi kunna det här kommandot”. Det börjar med ett kommando, print-kommandot, sen kommer variabler. Då ska man kunna använda variabler ihop med print-kommando, allting bygger på att ta sig framåt och det finns ett avgränsat mål med varje lektion [...] efter den här lektionen ska du egentligen ha koll på hur man använder detta, för nästa gång behöver vi bygga vidare på hur man använder det kommandot för att komma lite längre. Det är lite så den boken ”räkna med kod” är uppbyggd. (L4)

Trots skillnader i upplägg, kunde vissa likheter mellan lärarnas undervisningsstil identifieras. L1 och L4 inledde med att visa olika kommandon på helskärm som eleverna sedan får testa själva.

> Antingen har vi haft det på distans, att jag delat (skärm) och visat min fil eller så har jag använt projektorn och tagit upp det på en stor skärm. Då har jag visat och gått igenom hur man kan göra, hur kommandot fungerar och sen får de då testa själva. (L4)

L3, som hade visat Python för eleverna tidigare, presenterade bara uppgiften för eleverna och L2 började med att titta på ett klipp om programmering och efter det fick eleverna programmera varandra i klassrummet som introduktion till programmering.

> Vi tittade på den där mackan. Har du sett den? Pappan ska göra en sirapsmacka eller vad det är, något annat äckligt som han ska göra. Och just det här att man måste ta det i rätt steg, annars så blir det inte rätt. Och det fick de göra först och sen så gjorde vi alla sånt där ”gå framåt tre steg, vänd och tillbaka” (L2)

Under lektionerna gav alla lärare sina elever möjligheten att arbeta i par eller hjälpa varandra, även om de satt vid varsin dator och arbetade. Lärarna uppgav också att deras roll blev att handleda eleverna genom att gå runt och ställa utmanande frågor. En annan likhet var att samtliga lärare tyckte det var viktigt att stegvis utvidga elevernas kunskaper i programmering.

> I den här uppgiften skriver de nästan ingenting själva. Utan det här handlar bara om att de får en färdig kod och lägger in den. Undan för undan, lägger de in fler och fler grejer. Och så ska de tolka vad som händer, vad koden gör liksom. (L3)

> Då i detta fallet var det ju samma amortering hela tiden och sen att man frågar. ”Vad är det nu man vill ta reda på? Ja, räntan. Hur ska vi beräkna den?” alltså att man stegvis bygger upp det tillsammans med eleverna. (L1)

Det här upplägget kunde dock medföra olika svårigheter vad lärarna kunde se. En av lärarna tyckte att den största svårigheten med det stegvisa upplägget var att eleverna kunde hamna efter om de missat något moment.

> Men sen är det ju svårt, kommer man efter i ett steg så faller det liksom. Så det gäller ju verkligen att ha koll på att alla är med, vilket är svårt när man själv står framme vid sin egen dator och är själv fokuserad på om jag har gjort det i rätt ordning. [...] Om de missade något steg, var det svårt att komma vidare. Då gäller det att de säger ifrån, att de vågar säga ifrån helt enkelt och inte ha den inställningen ”nu måste jag hänga på”. För det går inte att i varje moment kolla alla elevers datorer så att de faktiskt har gjort det som jag sa. (L1)

Läraren löste problemet genom att pausa och gå runt bland eleverna men betonade elevernas ansvar i att säga ifrån om de skulle missa något steg. De övriga lärarna kunde också se hur eleverna hade ett ansvar i att stanna upp och reflektera över om de faktiskt förstod kommandona eller vad som hände i programmet, eftersom det lätt blev att eleverna bara körde på. En av dem ansåg att boken premierar procedurkunskapar och förklarar det på följande sätt:

> Den där boken är ju inte bra. Den är ju som alla andra böcker. ”Gör så här”. Och jag gjorde ju lite efter den, också för jag inte har så mycket erfarenhet. Men sen så fick de (eleverna) några uppgifter som inte är med där. Det här (uppgifterna) om att jämföra (area/omkrets). [...] Det blev ganska svårt helt plötsligt. [...] Så att följa en sån bok och att räkna och skriva in, de tyckte de ju va roligt. Men ja… hur mycket man lär sig på det, det vet jag inte. (L2)

Läraren i citatet ovan försökte få eleverna att representera beräkningar de gjort med penna och papper i programmeringsmiljö när eleverna började tycka att det var svårt. L4, som också utgick från boken, försökte undvika problemet med boken genom att skriva egna versioner av lektioner som finns på Google Colab.

> Jag har tittat på hur en lektion ser ut och så har jag skrivit en egen version utav det med exempel som jag har visat upp. Så när vi går igenom ett kommando, pratar jag lite och så står det text och så har vi en programmeringsdel där jag kan visa vad som händer. Sen får de övningar i en annan fil av simulator. Så att de kan köra och skriva men så har de mitt material att kunna gå tillbaka till […] Det kan vara att jag har tagit en liten del utav det som är här och en del som är där. Och när de ska skriva det, ska de egentligen koppla ihop de här två delarna. De kan inte ta mina exempel rakt upp och ner för det svarar inte på den frågan som de ska göra riktigt. (L4)

En annan svårighet som lärarna lyfte mer allmänt om programmering var just att mycket tid går att lära sig kommandona och på att rätta till fel i syntaxen och då hamnar lektionerna väldigt långt från matematiken.

> Man vill att de ska få någon matematisk insikt men hela lektionen har gått åt till att felsöka för att de inte har satt ut någon punkt rätt. Och det kan leda till frustration hos elever som man bara sitter och hjälper med att sätta ut rätt mellanslag, i stället för att prata om matematik. (L1)

En av lärarna såg hur hens lektion undvek det genom att vara utformad ett sätt där eleverna inte behövde producera koden själva:

> Eftersom de inte skulle skriva någonting själva, utan de fick mest bara reflektera över (kodningen), tror jag att de ibland kunde förstå bättre än om de skulle skriva själva. Skriver de koden själva är det så mycket fokus på ”okej, vad ska jag skriva nu”. […] Nu kunde vi resonera oss fram till många saker. Det märkte jag hos eleverna, att det kunde de ändå. (L3)

5.4 Tema 3: Elevernas lärdomar

Det tredje och sista temat behandlar hur lärarna i studien uppfattade att programmering bidrog till någon kunskapsmässig utveckling för eleverna. Lärarna talade om detta på två olika sätt. Dels vad de tror att programmering kan utveckla för förmågor hos eleverna, dels vad de kunde se att deras lektioner hade för effekt på elevernas lärande. Flera av lärarna valde också att prata om hur eleverna upplevde programmeringsmomenten, något som indirekt kan ses som hur eleverna förhåller sig till programmering som nytt kunskapsområde.

5.4.1 Förmågor som programmering kan utveckla

Lärarna i studien hade lättare för att teoretiskt prata om vilka förmågor som elever hade möjlighet att utveckla med hjälp av programmering. En av lärarna verkar ha tänkt över detta noggrant och förklarar det på följande sätt:

> Det (programmering) tränar upp en noggrannhet hos eleverna och det tycker jag är väldigt bra. Man måste förstå alla led, man kan inte hoppa över någonting och man kan inte slarva för då blir det fel. Det tränar felsökning, som man kan förhoppningsvis ha nytta av när det blir fel i en vanlig matematikuppgift [...]. Det tycker jag programmering är bra för. (L3)

En annan lärare tycker i stället att programmering är bra för att träna eleverna i problemlösningsmoment, något som läraren kan se att många elever brukar ha svårt för och därför är programmering ett bra inslag, menar hen.

> Jag skulle säga vad det kan förbättra är deras problemlösningsförmåga, att stycka upp problem i delar. [...] För problemlösning handlar om att identifiera vad jag har fått för någonting, vilka steg behöver jag göra för att komma fram till det jag vill. […] Det spelar ingen roll om jag gör det med penna och papper eller om jag tänker till hur ett program ska se ut och kunna göra det där. (L4)

En tredje lärare urskiljer sig från dessa två genom att i stället se programmering som ett sätt för eleverna att träna på att generalisera matematiken, vilket beskrivs i citatet nedan:

> Det är lätt att eleverna fastnar i att ”nu ska jag räkna det här och det blir fem. Ja, du har rätt det blir fem.”, i stället för att förstå metoden runt omkring. Och om man ska programmera, eller i alla fall gör något generellt program, så behöver man förstå generaliseringen på ett annat sätt (L1)

5.4.2 Synliga effekter på elevernas lärande

I slutet av intervjun fick lärarna svara på frågan om de kunde se att eleverna utvecklade några särskilda förmågor under de programmeringslektionerna som intervjun fokuserade på. Alla lärare fick fundera en stund innan de gav något svar. De två lärarna som hade tydliga, matematiska mål med undervisningen, L1 och L3, kunde se hur eleverna nådde dessa. Däremot kunde de inte riktigt säga några andra förmågor som utvecklades, förutom att möjligtvis att kunna resonera kring det matematiska innehållet. Samtliga lärare hade svårt att sätta fingret på om eleverna faktiskt utvecklade några förmågor efter lektionerna och två av lärarna besvarade frågan på följande sätt:

> Nej, det skulle jag väl inte säga att jag har sett några jättestora tecken den här gången […] Däremot kan jag se att de kanske kan ge ökad förståelse för vissa saker, till exempel att hantera större datamängder. (L4)

> Nu var det ett tag sen, jag kommer inte direkt ihåg. Det är klart att de utvecklade färdigheter men inte så att jag kunde se att de kom nya eller att jag kunde använda mig utav (dem). Men jag tror att det beror på att det var första gången jag gjorde det. Och jag har inte gjort det mer. Sen har jag gjort det i tekniken. (L2)

5.4.3 Elevernas upplevelse och inställning

Det har tidigare redogjorts för hur eleverna sällan hade några förkunskaper i programmering. Det innebär att lärarna ofta blir de första att introducera programmering för dem och kan därför på ett relativt rättvist och ofärgat sätt tolka hur eleverna upplever momenten.

Lärarna i studien är ganska överens om att de flesta elever de möter ändå tycker att programmering är ett roligt inslag, trots att det är något nytt för en stor majoritet av dem. L2, en av lärarna som utgick från boken "Räkna med kod", uppger följande iakttagelse:

> Alltså det var väldigt många som var väldigt med på det och som kunde väldigt mycket. Alltså de som kunde, de kunde ju bra. Det var väldigt många som drog i väg och som räknade ut hela den där (boken). De ville ha mer och mer och mer och mer. (L2)

Detta är dock något som varierar, menar lärarna, både från klass till klass och från individ till individ. En av lärarna delade med sig av sin observation och teori till vad det kan bero på:

> Vissa tycker att det är jätte, jätteroligt och vissa tycker att det är fruktansvärt. De som brukar ha något emot det, som uttryckligen har något emot det, är de eleverna som tycker om ”vanlig” matematik och som är ganska duktiga i vanlig matematik. För det här är inte det de är bra på och då tycker de inte om att göra det. (L3)

L4 som har undervisat programmering under flera, sammanhängande lektioner, har en annan bild av elevernas upplevelse.

> Flera tycker att det är intressant tills det blir jobbigt (skratt). Nej, men det är några som tycker att det är roligt sen är det alltid några som tycker ”varför måste jag lära mig det här?”, för de ser det bara som en sak till som är jobbig att lära in. Eller också kan det vara att det helt enkelt inte är intresserade av att skriva ett program som gör någonting. Men det är samma problem egentligen som man har med allt annat. (L4)

6. Diskussion

Nedan kommer studiens resultat att diskuteras utifrån den tidigare forskning och bakgrund som presenteras under avsnitt 2. Lärarnas likheter och skillnader kommer att betraktas i förhållande till de två ramverk som studien baseras på. Valet av ramverk kommer även att diskuteras under metoddiskussionen, tillsammans med andra metodval som kan ha påverkat studiens resultat. Avslutningsvis kommer förslag på framtida forskningsområden ges tillsammans med de didaktiska konsekvenser som arbetet har resulterat i.

6.1 Metoddiskussion

För studien har en kvalitativ datainsamlingsmetod använts för att bäst kunna besvara frågeställningarna och uppnå studiens syfte. Metoden lämpar sig väl för att undersöka lärarnas tolkningar av styrdokumenten och upplevelser av programmeringsundervisningen. Däremot är det svårt för en utomstående att på ett rättvist sätt bedöma hur själva undervisningen har genomförts. Det kan bero på att lärarna endast svarar på några, utvalda frågor utifrån minnet och hade inte i förväg blivit förberedda på vilka frågor som skulle diskuteras, utan endast på att ta med material som använts vid undervisningen. I vissa fall uppfattades det som att lärarna inte kom ihåg, speciellt i de fall där de inte undervisat programmering i samband med matematik under det senaste året. Det gör att deras utsagor mindre tillförlitliga än om de till exempel skulle förhålla sig till frågorna direkt efter avslutande programmeringlektioner och redan i förväg visste vad de skulle diskutera. Hade de vetat frågorna i förväg, hade de haft tid att fundera kring sina val och metoder och motiverat dem på ett bättre sätt. Av den anledningen blev det även svårt att analysera materialet utifrån en instrumentell ansats. Beskrivningen av lektionerna saknade en viss detaljrikedom som hade gjort analysarbetet något enklare. Det kan också bero på att frågorna inte var tillräckligt specificerade för att lärarna skulle kunna återge dem på ett detaljerat sätt. Just den delen av intervjun gick inte heller att testa i pilotintervjun, vilket kan ha varit anledningen till att frågorna som användes vid intervjun inte genererade de bästa svaren.

Ramverket om didaktisk transposition av Chevallard har varit en bra teori för studien, eftersom det har gett arbetet en tydlig struktur och röd tråd. Däremot kan det vara svårt att upptäcka och synliggöra transpositionsprocesserna eftersom kunskapsstoffet inte definieras för varje fas. Den akademiska kunskapen “matematik” definieras på nästan identiskt sätt som de kunskaper i matematik som eleverna erhåller, vilket kan skapa förvirring stundtals.

Den kvalitativa datainsamlingsmetoden är inte heller möjlig att generalisera till en hel grupp av matematiklärare. Studien undersöker alltså bara de tillfrågades verklighet och de likheter och skillnader som presenteras säger ingenting om hur det ser ut i resten av landet. Däremot har urvalet av lärare valts ut med hänsyn till deras undervisningsbakgrund och programmerings-erfarenhet, för att ge en god spridning i resultatet. Av naturliga skäl har främst lärare inom Göteborgsregionen tillfrågats och de lärare som kunde delta utifrån kriterieurvalet var främst högstadielärare. Detta eftersom programmering är ett obligatoriskt moment på högstadiet till skillnad från många matematikkurser på gymnasiet. Den kvalitativa metoden förutsätter också att flera tolkningsprocesser har skett under studiens gång. Lärarna som deltog tolkade intervjufrågorna och jag tolkar deras svar under analysprocessen. Möjligheten finns att missuppfattningar har skett som inte uppmärksammats.

Även om deltagande i studien var helt frivilligt, är det rimligt att anta att lärarna kände en viss press till att delta. Eftersom studien utgått från ett bekvämlighetsurval och de tillfrågade är

personer i mitt personliga nätverk som kanske snarare ställer upp för att de vill mig väl snarare än att de känner sig motiverade till att besvara frågor om ämnet. Det positiva med urvalet är att det kan visa en bättre bild av verkligheten ute på skolor, jämfört med den tidigare forskning som gjorts med utgångspunkt i lärare som är drivna i frågan. Den negativa effekten blir dels att lärarna kan uppleva ett visst tvång, dels att jag har svårt att vara i rollen som intervjuare när jag vill ta hänsyn till dem. Det kan till exempel handla om att jag inte vill dra ut på tiden för mycket genom att ställa för många följdfrågor, vilket gör det svårare utvinna relevant information. Däremot var många frågor av en öppen karaktär och lärarna fick tala ganska fritt om ämnet. För många följdfrågor hade också kunnat göra att informanterna leddes in på banor de inte tänkt på i förväg.

6.2 Resultatdiskussion

Genom att implementera programmering i matematikämnet, har en transponering av båda kunskapsområdena gjorts av Skolverket. Matematikämnet har förändrats i undervisnings-sammanhang för att ha ett större fokus på tillämpad matematik, där skrivningarna om programmering räknas in. Den akademiska kunskapen programmering har också förändrats då det nu i skolan ska ses som att tillhöra matematikämnet, i stället för att vara en egen disciplin. När en transponering av kunskaper sker, är det viktigt att aktörerna är tydligt hur processen gått till och vad deras intentioner är. Lärarnas syn på styrdokumenten blir ett kvitto på om Skolverket har lyckats med detta.

Lärarna som deltog i studien visade god förståelse för Skolverkets intentioner med de nya skrivningarna om programmering i styrdokumenten. Samtliga lärare höll med om att eleverna behöver möta programmering redan i grundskolan för att de ska kunna intressera sig för det och därmed välja utbildningar innehållandes programmering. Precis som Helenius m fl. (2019) motiverar programmering, säger lärarna att undervisning i programmering är ett måste för att täcka den framtida arbetsmarknaden men även för att få en bättre förståelse för eleverna digitala omvärld. Det fanns lärare i studien som ansåg att programmering kan erbjuda eleverna en möjlighet att få djupare förståelse för matematik. Andra informanter tyckte att programmering hade en mer given plats i teknikämnet och att det blev svårt att veta vad målet med programmering är när det finns i två skilda ämnen. Utifrån materialet kunde inga slutsatser dras om att lärare skulle se de två olika akademiska kunskapsområdena som ett och samma efter införandet, även om de kunde se att programmering eventuellt kan bidra till lärdomar som går att anpassa på matematik. Resultatet stämmer således överens med tidigare forskning där lärare ofta ser det som två olika discipliner, även om kopplingar mellan dem finns (Misfeldt m fl., 2019; Kilhamn m fl., 2021b).

Den transponering av kunskapsstoffet som skett i fas 2 av den didaktiska transpositions-processen, verkar inte ha påverkat de intervjuade lärarnas syn på vad programmering är. Det kan bero på att lärarna hade svårt att förstå styrdokumentens skrivningar om programmering. Samtliga lärare hade svårt att förstå vad skillnaden, enligt styrdokumenten, skulle vara mellan digitala verktyg och programmering. Detta liknar resultatet från Bråting m fl. (2021), där lärare kände sig osäkra på om GeoGebra och kalkylark räknades som programmering. En intressant observation är också hur en av lärarna gått en fortbildningskurs i programmering som i huvudsak bestod av GeoGebra. Det vittnar om att det inte enbart är lärare som har svårt att tolka skrivningarna om programmering utan att andra aktörer verkar tolka det transponerade ämnesstoffet på ett annat sätt än vad Skolverket egentligen avser. I avsnitt 2.2 presenteras hur begreppet programmering har tolkats och då täcks inte användning av GeoGebra in i den

definitionen, om inte möjligtvis dess kalkylark används. De reviderade ämnesplanerna för gymnasieskolan, förtydligar kraven som ställs på programmeringsinnehåll i matematikundervisningen. Förhoppningsvis får lärare möjlighet att tillsammans diskutera och analysera dessa så att tolkningen blir likvärdig och stämmer överens med Skolverkets avsikter.

Två av lärarna i studien tog hjälp av en lärobok för att bedriva programmeringsundervisningen. Forskning har tidigare visat att matematiklärare känner sig oförmögna till att formulera sina egna programmeringsuppgifter (Bråting m fl., 2021) och att läroböcker har stort inflytande över lektionsinnehållet (Bråting & Kilhman, 2021). Att skapa lektioner utifrån externa aktörer innebär att kunskapsstoffet har tolkats av läromedelsförfattarna i stället för enbart av den undervisande läraren. Från vad jag kunde se, förmedlade den nämnda boken främst rena programmeringskunskaper och gjorde sällan explicita, matematiska anknytningar. Detta har visats tidigare för lägre årskurser (Bråting & Kilhamn, 2021) och kräver därför att läraren kompletterar med att göra dessa kopplingar mellan ämnena, likt vad författarna Misfeldt och Ejsing-Duun (2015) beskriver. Det uppdraget känns dock svårt att uppnå när lärarna i dagsläget har svårt att tolka vilka krav som ställs på programmeringsmomenten och saknar didaktiska kunskaper i programmering.

Kilhamn m fl. (2021a) tolkade sitt resultat som att lärare mer och mer verka se programmeringsmoment som en matematisk handling. Författarna resonerar att det beror på att lärarna, i börja av införandet, behövde fokusera mer på vad som skulle programmeras och hur de momenten skulle gå till. Medan det på senare år har övergått till att mer handla om att tillämpa matematiken och betrakta programmering som ett matematiskt verktyg. Jag delar författarnas, och den här studiens informanters, förhoppningar om att det kommer bli så men utifrån det resultat som gavs här sågs inga tecken på att det har uppnåtts. Det beror kanske främst på att lärarna inte upplever att eleverna har några förkunskaper och måste därför lägga mycket tid på att lära dem grunderna. Att två av de fyra tillfrågade lärarna inte har undervisat programmering i samband med matematik det senaste året, är ett bevis för hur lätt det blir bortprioriterat. Det är extra tydligt när en pandemi har påverkat det senaste årets skolgång och mer akuta kunskaper, som eleverna faktiskt bedöms i, tar upp lärarnas undervisningstid. Att programmering inte prioriteras innebär att progressionen mellan årskurserna kanske aldrig kommer uppnås, i alla fall inte så länge styrdokumenten ser ut som de gör och inte har programmering som kunskap som ska bedömas.

Syftet med studien var att få en inblick i hur undervisning med programmering bedrivs i dagsläget. I missivbrevet (se bilaga 1) stod det uttryckligen att lärarna gärna fick ta med uppgifter där problemlösning var i fokus. Huruvida lärarna ansåg att deras medtagna undervisningsmaterial var av problemlösningskaraktär eller inte, adresserades inte vid intervjutillfället. Programmering står endast under problemlösning i gymnasiets ämnesplaner och antingen under problemlösning eller algebra i grundskolans kursplaner. Läraren från gymnasiet får antas anse att hens uppgift innefattar problemlösning och L3, som konstruerat sin egen uppgift, låg närmare problemlösning än algebra. De två lärarna som utgick från läroboken är det således svårt att dra några slutsatser om. Det hade varit intressant att veta om lärarna anser att eleverna arbetar med problemlösning utifrån det material som lärarna tog med till intervjun.

Målen med lärarnas undervisning i programmering skildes vida åt och urvalet är för litet för att dra några slutsatser från det. I likhet med Kilhamn m fl. (2021a) studie rörande lesson studies kan olika typer av mål urskiljas. I två fall ville lärarna fördjupa elevernas syn på matematiska

begrepp och en av lärarna ville att eleverna skulle få några grundkunskaper i programmering vid lektionens slut. De olika typerna av mål är tecken på hur lärarna transponerat kunskapsstoffet som styrdokumenten uppmanar dem till att lära ut. Det är naturligt att dessa skiljs åt i och med de tolkningsmöjligheter som finns av skrivningarna om programmering men tyvärr kan likvärdigheten i undervisningen drabbas av det. En positiv, överraskande likhet är att samtliga högstadielärare väljer att programmera i samma språk trots att det inte finns några bestämmelser för det. De var överens om att blockprogrammering kan ha fördelar som att det är enklare men att textprogrammering känns mer verkligt. Enligt Chevallard (2006) är det också viktigt att det transponerade ämnesstoffet fortfarande har en genuin karaktär av det akademiska kunskapsstoffet. Detta tycker jag att lärarna i studien också tagit hänsyn till när de motiverar deras val av programmeringsmiljö.

Enligt den instrumentella ansatsen har en artefakt förvandlats till ett instrument när subjektet kan använda artefakten för att uppnå sitt mål. Som nämnt i avsnitt 3.2.2 sker det en instrumentell genesis hos eleverna vid denna förvandling som består dels av instrumentalisation, dels instrumentering. Dessa två processer verkar samtidigt och kan vara svåra att skilja åt. Utifrån lärarnas lektioner kan det antas att eleverna inte har kommit så långt i instrumentalisationsprocessen eftersom samtliga uppgifter handlar i huvudsak om att upptäcka och upprepa olika kommandon. I slutetskedet av instumentalisationen ska elever själva kunna anpassa verktyget så att den passar just dem. Att nå hit är något som kräver mycket arbete och tid, vilket gör det naturligt att de inte har kommit längre.

Instrumenteringen bygger på att utveckla scheman som utgör en förändring av subjektet. Scheman talar om när en elev kan utföra en handling, förvänta och planera handlingen och förstå vad handlingen gör. Från resultatet blir det tydligt att lärarna i studien hade svårt att få eleverna att utveckla scheman eftersom eleverna ofta hade svårt för att stanna upp och reflektera över vad det är som händer. Trouche (2004) skiljer på två typer av scheman, användingsscheman och instrumentella handlingsscheman. Lärarna som utgick från boken "Räkna med kod" kan tänkas främst utveckla elevernas användingscheman i och med att boken lär dem att använda olika kommandon men inte lika mycket hur dessa kommandon kan användas för att lösa uppgifter. De två andra lärarna verkade ha större fokus på att utveckla elevernas handlingsscheman, eftersom uppgifterna de tog med uttryckligen är till för att använda programmering för att lösa matematiska uppgifter. Det kan vara svårt att hitta uppgifter som lämpar sig för den typen av problem. Utan särskilt mycket didaktisk erfarenhet i programmering blir konsekvensen att lärare använder och accepterar de läromedel som finns till sitt förfogande, även om de egentligen inte tycker att de är särskilt bra. Jag tror att det är viktigt att lärarna får stöd i sin programmeringsundervsining med uppgifter och lektionsplaneringar som lämpar sig för att undervisa programmering som ett matematiskt verktyg. Sådana uppgifter bör även innehålla exempel på hur programmeringsinnehållet kan relateras till matematik, vilket jag anser saknas idag.

Vid närmare analys av lärarnas instrumentella orkestrering av lektionerna, kan även här en skillnad urskiljas av vilken nivå lärarna gör det på beroende på om de använder läroboken eller inte. Eftersom boken närmast behandlar själva artefakten och dess olika kommandon, stannar undervisningen vid den första nivån om lärarna enbart använder boken vid sin undervisning. Det är något som lärarna verkar medvetna om i och med att de har kompletterat undervisningen med andra uppgifter som skulle få eleverna att använda programmering som ett verktyg, och inte bara memorera olika kommandon. L3 visar exempel på hur hen försöker skapa en relation mellan eleverna och programmering, genom att ha ett tydligt mål med att eleverna ska förstå

vad programmeringen gör för matematiken. Det visar tecken på den tredje nivån av instrumentell orkestrering. Det är således svårt att rättvist utvärdera lärarnas lektioner utifrån de tre nivåerna eftersom jag inte observerade lektionerna själv och lärarna endast har återberättat hur det var från minnet.

Utifrån forskningen om instrumentell orkestrering som Guin och Trouche (1998) presenterade, är det positivt att många lärare ger elever en möjlighet till samarbete eftersom detta har visat sig vara uppskattat vid skapande av instrumentell genesis. Samma sak gäller att visa manipulationer av verktyget på helskärm och kombinera undervisningen med tekniska artefakter med uträkningar med penna och papper, vilket några av lärarna hade valt att göra. För att nå längre i elevernas skapande av instrument hade dessa metoder kunnat framhävas ytterligare under lärarnas instrumentella orkestrering, om man vill lyssna till Guin och Trouches forskning. Återigen tror jag det har att göra med den brist av didaktisk kunskap som lärare verkar ha. Bara för att lärare är kunniga inom ett didaktiskt fält, innebär inte det att de metoderna går att anpassa på ett nytt ämne. Trouche (2004) säger att instrumentell orkestrering kräver god förståelse för artefakten och lektionerna måste noga planeras för att matematiska lärdomar ska kunna etableras hos eleverna. Det räcker inte med att ge lärarna kunskaper i programmering, utan de behöver också handfasta, didaktiska metoder för att kunna undervisa det på ett måluppfyllande sätt.

Vad eleverna faktiskt tar med sig för kunskaper från programmeringslektionerna är svårt att säga. Lärarna i studien kunde inte direkt se att eleverna utvecklade några särskilda förmågor under deras lektioner. Detta skiljer sig något från resultatet i Nouri m fl. (2020), där lärarna uppgav att eleverna utvecklade flera generella förmågor som logiskt tänkande och samarbetsförmåga. Detta trodde författarna beror på att lärarna inte är vana att bedöma elevernas förmågor i samband med programmering, vilket även går att applicera på resultatet från den här studien. Lärarna verkade se många möjligheter för vilken den lärda kunskapen, fas 4 enligt Chevallard (2006), blir från att programmera. Det går däremot inte att visa vilka kunskaper det är utifrån intervjuerna. I och med det tycker jag det är viktigare att ta fasta på att många elever ser det som ett roligt inslag i skolan, vilket bådar gott för framtidens behov av programmerare.

6.3 Framtida forskning

Det främsta behovet av vidare forskning är fält som berör elevernas kunskapsutveckling vid programmeringsmoment. Utifrån resultatet från den här studien, är det inte tillräckligt att intervjua lärare för att ta reda på vilka kunskaper och förmågor elever faktiskt tar med sig vid den här typen av undervisning. Därför behövs mer konkret forskning som ämnar till att undersöka elevernas vunna, matematiska kunskaper efter undervisningen med programmering. Det skulle vara intressant om forskningsprojekt kring programmering som ett matematiskt verktyg kunde bedrivas på ett liknande sätt som forskning har gjorts kring instrumentell orkestrering, vilket visas i avsnitt 3.2.3.1. Då hade forskningen kunnat bidra med råd för hur programmeringsundervisningen ska bedrivas i klassrummen.

Det vore även intressant att vidare fråga lärare om deras syn på programmering vid just problemlösning. Hur lärare tolkar begreppet problemlösning och hur de implementerar programmering vid problemlösningsmoment, är i dagsläget inget som är kartlagt av forskare. Sådan typ av forskning kan påvisa skillnader inom tolkningar av styrdokumenten som leder till

skillnader i undervisningen. Det kan på sikt leda till att Skolverket blir tvungna att omformulera eller på andra sätt förtydliga de skrivningar om programmering som finns i dagsläget.

6.4 Didaktiska konsekvenser

Samtalen med lärarna har gett mig uppfattningen av att programmering är något som de flesta elever tycker är roligt att arbeta med, vilket jag kommer ta med mig in i lärarrollen. Även om det känns svårt och otäckt att undervisa i kunskaper som jag inte fullt ut behärskar, tror jag ändå det är viktigt att göra det. För om progressionen i programmering mellan årskurserna inte uppnås, riskerar elever att helt gå miste om att uppleva programmering. De kommer då heller inte förstå vilka fördelar programmering kan ha för matematikämnet eller om det är något som de kan tänka sig arbeta med i framtiden.

Jag tar även med mig att undervisa i programmering inte behöver vara särskilt avancerat eller omfattande för att nå upp till några matematiska mål. I dagsläget ställs inga krav på att eleverna ska kunna koda, i och med att det inte är en kunskap eleverna bedöms på. Det kanske är bra att det är så för då kan man som lärare i stället skriva programmen åt eleverna vilket minskar chansen att onödigt fokus läggs på att kunna syntaxen och programmeringsspråket. Den främsta svårigheten kan vara att hitta problem där programmering lämpar sig att använda, framför allt vid problemlösning. En möjlig lösning på det problemet är att planera och utvärdera programmeringlektioner tillsammans inom arbetslaget och på så sätt hjälpas åt i att utveckla en god undervisning av programmeringsmoment.

Referenslista

Bosch M, Gascón J (2006) Twenty-five years of the didactic transposition. ICMI Bulletin 58: 51-64

Braun, V., & Clarke, V. (2006). Using thematic analysis in psychology. *Qualitative Research in Psychology, 3* (2), 77-101.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 Annual Meeting of the American Educational Research Association* (pp. 1–25). American Educational Research Association.

Bryman, A. 2002. *Samhällsvetenskapliga metoder*. Malmö: Liber

Bråting, K., & Kilhamn, C. (2021). The Integration of Programming in Swedish School Mathematics: Investigating Elementary Mathematics Textbooks. *Scandinavian Journal of Educational Research, Ahead-of-print(Ahead-of-print)*, 1-16.

Bråting, K., Kilhamn, C., & Rolandsson, L. (2021). Integrating programming in Swedish school mathematics: description of a research project. I *Sustainable Mathematics Education in a Digitalized World. Proceedings of MADIF12. The Twelfth Research Seminar of the Swedish Society for Research in Mathematics Education*, 101–110. Hämtad från http://urn.kb.se/resolve?urn=urn:nbn:se:uu:diva-439333

Chevallard, Y. (2006). Steps towards a new epistemology in mathematics education. In M. Bosch (Ed.*), Proceedings of the Fourth Congress of the European Society for Research in Mathematics Education, CERME 4* (pp. 21–30). Barcelona: FUNDEMI IQS-Universitat Ramon Llull.

Chevallard, Y., & Bosch, M. (2013). Didactic transposition in mathematics education. In S. Lerman (Ed.*), Encyclopedia of Mathematics Education*. Berlin: Springer

Dalen, M. (2015). *Intervju som metod.* Malmö: Gleerups.

Drijvers, P., Kieran, C., Mariotti, M., Ainley, J., Andresen, M., Chan, Y., . . . Lagrange, J. (2010). Integrating Technology into Mathematics Education: Theoretical Perspectives. In C., Hoyles, & J.B., Lagrange (Ed) Mathematics Education and Technology-Rethinking the Terrain: The 17th ICMI Study (Vol. 13, New ICMI Study Series, pp. 89-132). Boston, MA: Springer US.

Guin, D., & Trouche, L. (1998). The Complex Process of Converting Tools into Mathematical Instruments: The Case of Calculators. *International Journal of Computers for Mathematical Learning*, 3(3), 195-227.

Helenius, O., Misfeldt, M. & Rolandsson, L. (2019). *Programmering i matematik.* Hämtad från Skolverket: https://larportalen.skolverket.se/LarportalenAPI/api-v2/document/path/larportalen/material/inriktningar/1-matematik/Grundskola/438_matematikundervisningmeddigitalaverktygII_%C3%A5k7-9/del_04/Material/Flik/Del_04_MomentA/Artiklar/MA2_7-9_04A_01_programmeringi.docx

Helenius, O., Misfeldt, M., Rolandsson, L. & Ryan, U. (2018). *Om programmering i matematikundervisningen*. Hämtad från Skolverket: https://larportalen.skolverket.se/LarportalenAPI/api-

v2/document/path/larportalen/material/inriktningar/1-matematik/Gymnasieskola/448_matematikundervisningmeddigitalaverktygII_GY/del_01/Material/Flik/Del_01_MomentA/Artiklar/MA2_Gy_01A_01_omprogrammering.docx.

Hickmott, D., Prieto-Rodriguez, E., & Holmes, K. (2018). A Scoping Review of Studies on Computational Thinking in K–12 Mathematics Classrooms. *Digital Experiences in Mathematics Education, 4*(1), 48-69.

Jacobsson, K., & Skansholm, A. (2019). Handbok i uppsatsskrivande : För utbildningsvetenskap (Upplaga 1 ed.). Lund: Studentlitteratur

Kilhamn, C., Bråting, K., & Rolandsson, L. (2021). Teachers' arguments for including programming in mathematics education. In G. A. Nortvedt, N. F. Buchholtz, J. Fauskanger, F. Hreinsdóttir, M. Hähkiöniemi, B. E. Jessen,…, A. Werneberg (Eds.), *Bringing Nordic mathematics education into the future. Preceedings of Norma 20 The ninth Nordic Conference on Mathematics Education. SMDF,* Svensk Förening för MatematikDidaktisk Forskning, Nr 14.

Mannila, L. (2017). *Att undervisa i programmering i skolan: Varför, vad och hur?* (Upplaga 1 ed.). Lund: Studentlitteratur

Misfeldt, M., & Ejsing-Duun, S. (2015). Learning Mathematics through Programming: An Instrumental Approach to Potentials and Pitfalls. In K. Krainer, & N. Vondrová (Eds.), *CERME9* (pp. 2524-2530). Charles University in Prague, Faculty of Education and ERME.

Misfeldt, M., Jankvist, U. T., Geraniou, E., & Bråting, K. (2020). Relations between mathematics and programming in school: Juxtaposing three different cases. In A. Donevska-Todorova, E. Faggiano, J. Trgalova, Z. Lavicza, R. Weinhandl, A. Clark-Wilson, & H.-G. Weigand (Eds.), *Proceedings of the 10th ERME Topic Conference on Mathematics Education in the Digital Era, MEDA 2020* (pp. 255–262). Johannes Kepler University.

Misfeldt, M., Szabo, A., & Helenius, O. (2019). Surveying teachers' conception of programming as a mathematical topic following the implementation of a new mathematics curriculum. In U. Jankvist, M. Van den Heuvel-Panhuizen, & M. Veldhuis (Eds.), *Proceedings of the Eleventh Congress of the European Society for Research in Mathematics Education, CERME11* (pp. 2713–2720). Freudenthal Group & Freudenthal Institute, Utrecht University and ERME

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry*, 11(1), 1–17. https://doi.org/10.1080/20004508.2019.1627844

Veerasamy, A., D'Souza, D. & Laakso, M. (2016). Identifying Novice Student Programming Misconceptions and Errors From Summative Assessments. *Journal of Educational Technology Systems*, 45(1), 50-73.

Verillon, P., & Rabardel, P. (1995). Cognition and artifacts: A contribution to the study of though in relation to instrumented activity. European Journal of Psychology of Education, 10(1), 77-101.

Skolverket. (2016) *Redovisning av uppdraget om att föreslå nationella it-strategier för skolväsendet – förändringar i läroplaner, kursplaner, ämnesplaner och examensmål.* Hämtad 2020-05-19 från: https://www.skolverket.se/getFile?file=3668

Skolverket. (2017). *Kommentarmaterial till ämnesplanen i matematik i gymnasieskolan.* Hämtad 2020-05-04 från: https://www.skolverket.se/download/18.6011fe501629fd150a2893a/1530187438471/Kommentarmaterial_gymnasieskolan_matematik.pdf

Skolverket. (2018a). *Ämnesplan för matematik på grundskolan 2011: reviderad: 2018.* Hämtad 2020-05-04 från: https://www.skolverket.se/undervisning/grundskolan/laroplan-och-kursplaner-for-grundskolan/laroplan-lgr11-for-grundskolan-samt-for-forskoleklassen-och-fritidshemmet?url=1530314731%2Fcompulsorycw%2Fjsp%2Fsubject.htm%3FsubjectCode%3DGRGRMAT01%26tos%3Dgr%26p%3Dp&sv.url=12.5dfee44715d35a5cdfa219f

Skolverket. (2018b). *Ämnesplan för matematik på gymnasiet 2011: reviderad: 2018.* Hämtad 2020-05-04 från: https://www.skolverket.se/undervisning/gymnasieskolan/laroplan-program-och-amnen-i-gymnasieskolan/gymnasieprogrammen/amne?url=1530314731%2Fsyllabuscw%2Fjsp%2Fsubject.htm%3FsubjectCode%3DMAT%26tos%3Dgy&sv.url=12.5dfee44715d35a5cdfa92a3#anchor1.

Skolverket. (2021a). *Kommentarmaterial till kursplanen i matematik*. Hämtad 2020-05-04 från: https://www.skolverket.se/getFile?file=7840

Skolverket. (2021b). *Kommentarmaterial till ämnesplanen i matematik.* Hämtad 2020-05-04 från: https://www.skolverket.se/getFile?file=7841

Skolverket. (2021c). *Ändrade kursplaner – bättre arbetsverktyg för lärarna.* Hämtad 2020-05-04 från: https://www.skolverket.se/om-oss/var-verksamhet/skolverkets-prioriterade-omraden/reviderade-kurs--och-amnesplaner/andrade-kursplaner-i-grundskolan#Grundskolan

SFS 2003:460. *Lag om etikprövning som avser människor*. Hämtad 2020-05-24 från: https://www.riksdagen.se/sv/dokument-lagar/dokument/svensk-forfattningssamling/lag-2003460-om-etikprovning-av-forskning-som_sfs-2003-460

Trouche, L. (2004). Managing the Complexity of Human/Machine Interactions in Computerized Learning Environments: Guiding Students' Command Process through Instrumental Orchestrations. *International Journal of Computers for Mathematical Learning*, *9*(3), 281-307.

Trouche, L. (2005a). An Instrumental Approach to Mathematics Learning in Symbolic Calculator Environments. I Trouche, L., Guin, D., & Ruthven, K. (Ed) *The Didactical Challenge of Symbolic Calculators: Turning a Computational Device into a Mathematical Instrument* (Vol. 36, Mathematics Education Library, pp. 137-162). Boston, MA: Springer US

Trouche, L. (2005b). Instrumental genesis, individual and social aspects. I Trouche, L., Guin, D., & Ruthven, K. (Ed) *The Didactical Challenge of Symbolic Calculators: Turning a Computational Device into a Mathematical Instrument* (Vol. 36, Mathematics Education Library, pp. 198-224). Boston, MA: Springer US

Bilaga 1: Missivbrev

Hej XX!

Jag heter Liv och jag skriver just nu mitt andra examensarbete på lärarprogrammet i Göteborg. Arbetet kommer vara en kvalitativ studie som utforskar lärares tankar kring programmering i matematikämnet.

Datum för intervjun kan vi anpassa men gärna så snart som möjligt. Intervjun förväntas ta 45-60 minuter och hålls över Zoom. Jag ser gärna att vi diskuterar programmering utifrån en uppgift som du har använt dig av i din undervisning, gärna med problemlösning i fokus. Därför önskar jag att du tar med dig en uppgift som du har arbetat med i din undervisning. Det vore jätteroligt att få ta del av dina tankar!

Hör av dig om det skulle vara något du undrar eller om du behöver boka om tiden. Min e-mailadress är XX och mitt telefonnummer XX.

Tack på förhand!

Med vänliga hälsningar
Liv Ejsing

Bilaga 2: Intervjuguide

Introduktion

Jag skriver nu mitt andra examensarbete på ämneslärarprogrammet vid Göteborgs universitet. I det första arbetet gjorde jag tillsammans med två studenter en litteraturöversikt om svårigheter och möjligheter med programmering i matematikämnet. I den här studien kommer jag i stället intervjua lärare för att ta del av deras uppfattningar och erfarenheter kring programmering. Syftet är att få en insikt om hur lärare har tolkat styrdokumentens skrivningar gällande programmering och tillämpat dem i sin undervisning.

Alla deltagare i studien kommer att vara helt anonyma under hela processen och inspelningarna kommer att raderas efter det att intervjuerna har transkriberats. Jag kommer också försöka vara ganska tyst under intervjun, det är inte för att jag inte är intresserad utan för att underlätta transkriberingen. Att medverka är helt frivilligt och du kan när som helst välja att avbryta din medverkan. Med det sagt, godkänner du att jag spelar in den här intervjun och därmed ger samtycke till att delta?

Bakgrund

- Hur länge har du jobbat som lärare?
- Vilka klasser undervisar du idag?
- Har du några andra ansvarsområden på skolan utöver lärarrollen?
- Vilka andra ämnen undervisar du i?
- Vad har du för programmeringsbakgrund och hur lärde du dig att programmera?
- Vilket språk eller miljö har du valt att undervisa programmering i?

Styrdokumenten

- Vad tror du ligger bakom införandet av programmering i skolan?
- Hur reagerade du på förändringen när du fick veta?

  För gymnasiet:
  I kursplanerna på gymnasiet står programmering endast med som en punkt i det centrala innehållet under ”problemlösning”. Formuleringen lyder så här: *”Strategier för matematisk problemlösning inklusive modellering av olika situationer, såväl med som utan digitala verktyg och programmering*.”

  För högstadiet:
  I styrdokumenten står det under ämnets syfte så här angående programmering:
  ”Eleverna ska genom undervisningen ges möjligheter att utveckla kunskaper i att använda digitala verktyg och programmering för att kunna undersöka problemställningar och matematiska begrepp, göra beräkningar och för att presentera och tolka data.”

- Vad anser du är skillnaden mellan digitala verktyg och programmering? Hur skiljer man de åt?

  För högstadiet:
  Under centralt innehåll så står programmering både med under algebra och under problemlösning. Skrivningarna är snarlika men formuleringen för problemlösning lyder:
  ”Hur algoritmer skapas, testas och förbättras vid programmering för matematisk problemlösning”

- Hur har du tolkat skrivningen och sen överfört den till din undervisning?

- Hur motiverar du programmering för dina elever?
- Vilka fördelar kan du se att programmering har för matematikämnet?
- Vilka nackdelar kan det innebära att lära ut programmering som ett matematiskt verktyg?
- Ser du några matematiska områden där programmering lämpar sig bättre?

Undervisningen

- Berätta kort om uppgiften!
- Vilken årkurs/mattekurs är den för?
- Vilket matematiskt område behandlas?
- Var ligger den lektionen i grovplaneringen?
- Varifrån fick du inspiration till uppgiften/lektionen?
- Vad var lärandemålen för lektionen?
- Hur introducerades uppgiften för klassen?
- Vad hade eleverna för förkunskaper?
- Hur ville du att eleverna skulle arbeta tillsammans?
- Vad fungerade bra och vad fungerade mindre bra under lektionen?

Elevernas lärdomar

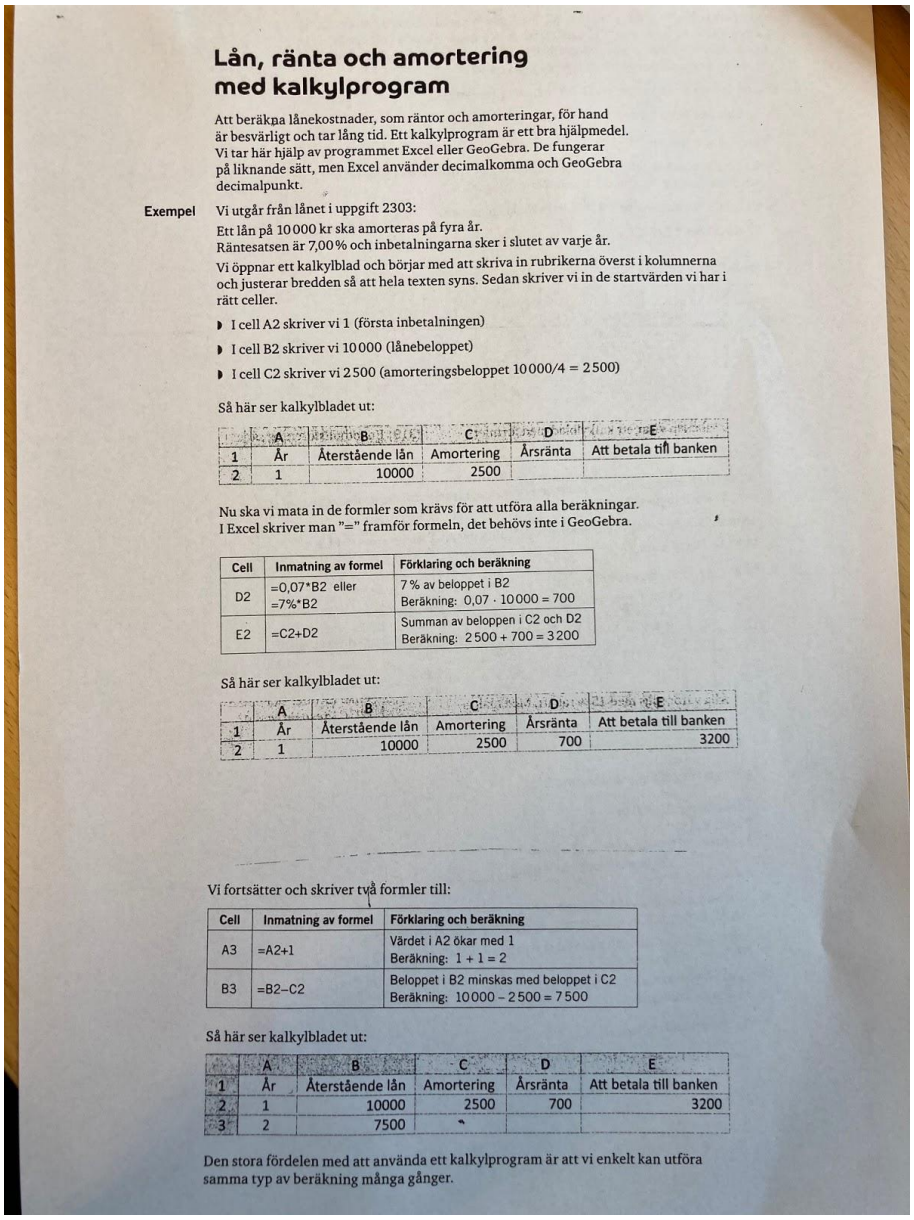
- Vilka svårigheter kunde du se att eleverna stötte på i din lektion?
- Vilka färdigheter kunde du se att eleverna utvecklade? Matematiska eller generella.
- Hur bedömde du eleverna kunskaper?

Avslutning

- Finns det något du vill tillägga?

Bilaga 3: Uppgiftsunderlag från L1

Läraren fick inspiration till uppgiften från en annan mattebok än den som användes i undervisningen. Läraren fotade av uppgiften som låg till grund för hens undervisning.



Lån, ränta och amortering med kalkylprogram

Att beräkna lånekostnader, som räntor och amorteringar, för hand är besvärligt och tar lång tid. Ett kalkylprogram är ett bra hjälpmedel. Vi tar här hjälp av programmet Excel eller GeoGebra. De fungerar på liknande sätt, men Excel använder decimalkomma och GeoGebra decimalpunkt.

Exempel Vi utgår från lånet i uppgift 2303:
Ett lån på 10 000 kr ska amorteras på fyra år.
Räntesatsen är 7,00 % och inbetalningarna sker i slutet av varje år.
Vi öppnar ett kalkylblad och börjar med att skriva in rubrikerna överst i kolumnerna och justerar bredden så att hela texten syns. Sedan skriver vi in de startvärden vi har i rätt celler.

- I cell A2 skriver vi 1 (första inbetalningen)
- I cell B2 skriver vi 10 000 (lånebeloppet)
- I cell C2 skriver vi 2 500 (amorteringsbeloppet 10 000/4 = 2 500)

Så här ser kalkylbladet ut:

	A	B	C	D	E
1	År	Återstående lån	Amortering	Årsränta	Att betala till banken
2	1	10000	2500		

Nu ska vi mata in de formler som krävs för att utföra alla beräkningar.
I Excel skriver man "=" framför formeln, det behövs inte i GeoGebra.

Cell	Inmatning av formel	Förklaring och beräkning
D2	=0,07*B2 eller =7%*B2	7 % av beloppet i B2 Beräkning: 0,07 · 10 000 = 700
E2	=C2+D2	Summan av beloppen i C2 och D2 Beräkning: 2 500 + 700 = 3 200

Så här ser kalkylbladet ut:

	A	B	C	D	E
1	År	Återstående lån	Amortering	Årsränta	Att betala till banken
2	1	10000	2500	700	3200

Vi fortsätter och skriver två formler till:

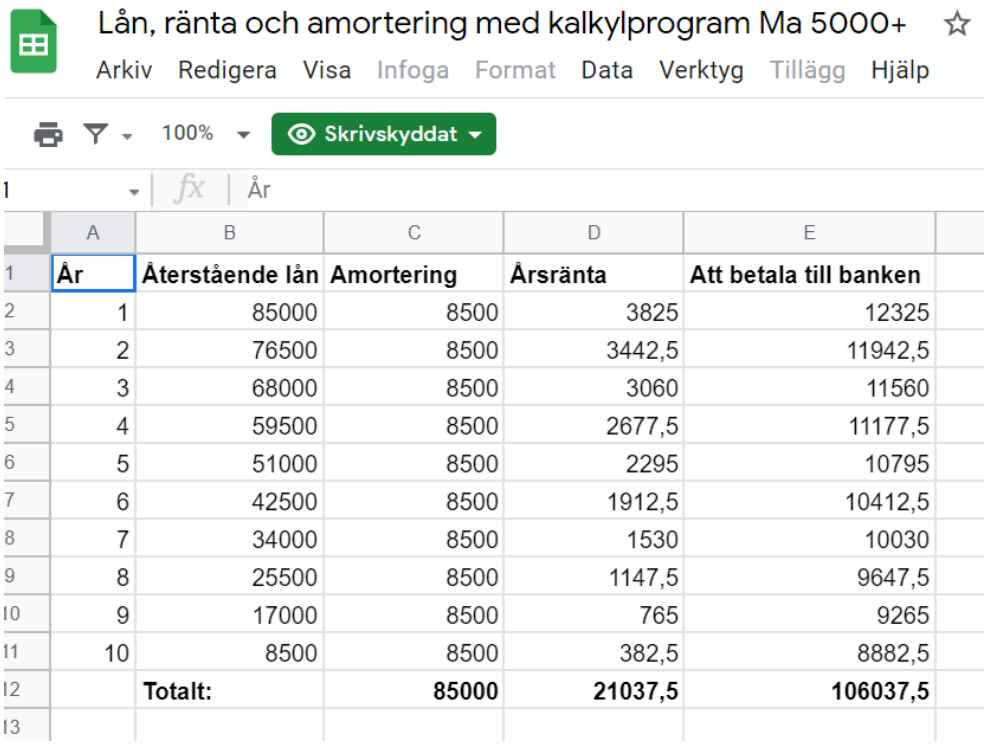
Cell	Inmatning av formel	Förklaring och beräkning
A3	=A2+1	Värdet i A2 ökar med 1 Beräkning: 1 + 1 = 2
B3	=B2–C2	Beloppet i B2 minskas med beloppet i C2 Beräkning: 10 000 – 2 500 = 7 500

Så här ser kalkylbladet ut:

	A	B	C	D	E
1	År	Återstående lån	Amortering	Årsränta	Att betala till banken
2	1	10000	2500	700	3200
3	2	7500			

Den stora fördelen med att använda ett kalkylprogram är att vi enkelt kan utföra samma typ av beräkning många gånger.

Läraren visade också hur dokumentet såg ut som de arbetade med under lektionerna. Bilden nedan visar vad eleverna fick möta under den första lektionen.



	A	B	C	D	E
1	**År**	**Återstående lån**	**Amortering**	**Årsränta**	**Att betala till banken**
2	1	85000	8500	3825	12325
3	2	76500	8500	3442,5	11942,5
4	3	68000	8500	3060	11560
5	4	59500	8500	2677,5	11177,5
6	5	51000	8500	2295	10795
7	6	42500	8500	1912,5	10412,5
8	7	34000	8500	1530	10030
9	8	25500	8500	1147,5	9647,5
10	9	17000	8500	765	9265
11	10	8500	8500	382,5	8882,5
12		**Totalt:**	**85000**	**21037,5**	**106037,5**
13					

Bilaga 4: Uppgiftsunderlag från L3

Programmeringsuppgift - vad är sannolikhet.

Sannolikheten för att få en sexa på en tärning är ⅙. Chansen att få krona när man singlar slant är 50%. Men vad innebär det? I den här uppgiften ska vi försöka förstå det lite bättre.

1. Öppna länken till repl.it och ta bort allt i den vänstra rutan. Länk till repl.it

2. Klistra in koden nedan och kör programmet genom att trycka på **Run**. Kör programmet några gånger. Vad gör programmet? Vad innebär de olika variablerna?

```
import random
antal_ettor = 0
antal_gånger = 5
for n in range (antal_gånger):
 tal = random.randint(1,5)
 print(tal)
```

3. Vi lägger till lite saker i koden så att den nu ser ut såhär:

```
import random
antal_ettor = 0
antal_gånger = 5
for n in range (antal_gånger):
 tal = random.randint(1,5)
 print(tal)
 if tal == 1:
   antal_ettor = antal_ettor + 1

print("antalet ettor är", antal_ettor, "stycken")
```

Tryck på **Run**. vad händer nu? Pröva flera gånger.
Vad händer om du ändrar variabeln `antal_gånger`?

4. Räkna ut hur stor sannolikheten i procent det är att få en etta om man slumpar fram ett tal mellan 1 och 5 på papper.

5. Vi ska nu låta programmet pröva om detta stämmer. Klistra nu in denna kod. Vad gör raderna längst ner?

```
import random
antal_ettor = 0
antal_gånger = 5
for n in range (antal_gånger):
 tal = random.randint(1,5)
 print(tal)
 if tal == 1:
   antal_ettor = antal_ettor + 1

print("antalet ettor är", antal_ettor, "stycken")

procent = antal_ettor/antal_gånger*100
print(procent, "% är ettor")
```

6. Pröva nu att köra programmet med antal_gånger = 10, sedan 100, 1000, 10 000, 100 000, 1 000 000. Vad händer med procenten?

7. Kan du dra någon slutsats om sannolikhet med hjälp av detta experiment?