MALMÖ UNIVERSITET
NATURVETENSKAP–
MATEMATIK–SAMHÄLLE

Examensarbete i fördjupningsämnet matematik och lärande
15 högskolepoäng, avancerad nivå

Strategier för arbete med matematisk problemlösning genom programmering

Strategies for teaching mathematical problemsolving through programming

Nour-Mariam Merhi

Ämneslärarexamen med inriktning mot arbete i gymnasieskolan, 300hp
Datum för slutseminarium (2019-06-03)

Examinator: Jöran Petersson
Handledare: Peter Bengtsson

Förord

Jag vill ägna ett stort tack till min handledare Peter Bengtsson som har handlett och korrekturläst arbetet under hela arbetsprocessen och jag vill tacka min examinator Jöran Petersson som bidragit med förslag på ytterligare förbättringar av detta examensarbete. Jag vill även ägna ett stort tack till alla lärare som har delat med sig av sina värdefulla kunskaper och erfarenheter vilket har gjort denna undersökning möjlig att genomföra. Jag vill också tacka Nadia Anjam som tillsammans med mig skrivit ett självständigt arbete tidigare. Delar av begreppsdefinitionerna (3.2.1-3.2.3.) i detta examensarbete har hämtats och inspirerats av detta självständiga arbete.

Ett speciellt tack vill jag ägna till min familj som har både stöttat mig och uppmuntrat mig under hela min lärarutbildning. Pappa och mamma som inspirerar mig i allt jag gör och mina syskon och min man som ger mig energi att fortsätta kämpa.

Abstrakt

Eftersom samhällets och skolans digitalisering sker i hög takt och Skolverket menar att programmeringskunskaper kommer att vara värdefulla i framtiden så har de beslutat år 2017 att programmering ska integreras i flera ämnen i skolan. Läsåret 18/19 skulle arbetet med programmeringen införas i kursplanerna för matematikämnet. Eftersom att detta beslut fattades under min tid på ämneslärarutbildningen så fick jag ett intresse att undersöka hur lärare ska undervisa i matematik och integrera programmeringen för matematisk problemlösning. Syftet med studien har således varit att belysa hur lärare, nyexaminerade såväl som erfarna, kan arbeta med programmering som en strategi för problemlösning genom att integrera programmering i matematikundervisningen. Undersökningen genomfördes genom ett antal intervjuer med verksamma lärare som har infört programmeringen i sin undervisning sedan det beslutades. Resultatet har analyserats med en modell om didaktisk funktionalitet av digitala verktyg i matematikundervisning och studien har gett svar om hur man som lärare kan använda olika undervisningsstrategier för att elever ska använda programmering i problemlösning. I studien ges exempel på olika uppgifter och arbetssätt som kan erbjuda pedagogiska möjligheter.

Nyckelord: Matematik, problemlösning, programmering, gymnasieskola, didaktisk funktionalitet

Innehållsförteckning

1. Inledning

Dagens samhälle och skola digitaliseras allt mer och enligt de nya styrdokumenten ska programmering ingå som en strategi för problemlösning i vissa matematikkurser men olyckligtvis kan inte många elever och lärare programmera om de inte läst en kurs i programmering på gymnasiet eller universitetet. Det nya tillskottet i det centrala innehållet under rubriken *problemlösning* i matematik 1c, 2c, 3b, 3c, 4, 5 och matematikspecialisering är ”Strategier för matematisk *problemlösning* inklusive modellering av olika situationer, såväl med som utan digitala verktyg och *programmering*” (Skolverket, 2018a, s. 9). Efter revideringen i kursplanen åk 7-9 så ingår punkten: ”Hur algoritmer kan skapas, testas och förbättras vid programmering för matematisk problemlösning (Skolverket, 2018b, s. 7). Då dagens ungdomar är omgivna av datorer och tekniska instrument så är det möjligt att det i framtiden kommer att finnas jobb som nu är oupptäckta där programmeringskunskaper kommer att vara värdefulla. Dessutom så finns det inte tillräckligt med forskning kring *hur* man ska arbeta med just matematisk problemlösning med programmering i gymnasie- och grundskolan. Detta gäller framförallt forskning på nationell nivå men även på internationell nivå. Det är relevant att undersöka hur man bör implementera programmering i matematikundervisning eftersom att flera studier indikerar att arbete med programmering kan utveckla problemlösningsförmågan (Clements & Gullo, 1984; Resnick et al., 2009; Saeli, Perrenet & Jochems, 2011; Clement, Lochhead & Soloway, 1980).

I en diskussion med en verksam gymnasielärare i matematik och fysik yttrade sig denna lärare: ”Ska man undervisa eleverna i programmering eller ska man undervisa dem i matematik?” eftersom att flera lärare har sett att eleverna inte kan programmera. Många lärare som jag har haft samtal med under den verksamhetsförlagda utbildningen förstår inte varför man ska integrera programmering i matematikkurserna. Dessutom så känner sig 8 av 10 matematiklärare på högstadiet osäkra inför att undervisa i ämnet (Larsson, 2017, september).

Examensarbetet som jag har skrivit handlar om hur befintliga lärare arbetar och hur man som lärare kan låta elever arbeta med matematiska problem och utveckla deras problemlösningsförmåga med hjälp av programmering. Uppsatsens relevans är en självklarhet eftersom att programmering har införts i styrdokument men det har inte riktigt implementerats i skolvärlden. Jag vill i samband med mitt examensarbete ta reda på *varför* programmering har införts som en strategi för problemlösning i matematikundervisningen och *hur* man kan arbeta med detta.

2. Syfte och frågeställningar

Syftet med detta examensarbete är att belysa hur lärare, nyexaminerade såväl som erfarna, kan arbeta med programmering som en strategi för problemlösning genom att integrera programmering i matematikundervisningen. Denna studie riktar sig främst för arbete i gymnasieskola men kan med fördel även användas i grundskola. Eftersom att arbetet med programmering är i fokus i denna uppsats så har jag valt att fokusera på strategier i undervisningen som lärare har valt att använda för att eleverna ska lära sig detta. Jag har valt detta eftersom att jag vill skriva om något som jag kan ha nytta av och använda i min framtida lärarprofession.

- Hur har verksamma lärare ute på fältet börjat integrera programmering i undervisning för arbete med matematisk problemlösning?
- Hur arbetar dessa lärare med att utveckla elevernas problemlösningsförmåga med hjälp av programmering?

3. Bakgrund

3.1. Varför har man infört programmering i kursplanerna?

Programmering har nu tillkommit som ett innehåll i den reviderade kursplanen för matematik (Skolverket, 2018b). I skolvärlden är de flesta överens om att dagens elever ska klara ett liv i en digitaliserad värld vilket innebär att de ska lära sig att hantera olika digitala verktyg både i ett praktiskt och pedagogiskt syfte. Men förutom detta så ska eleverna också lära sig att designa, konstruera och manipulera digitala verktyg och det är det som är tanken med den införda programmeringen i skolan. Tanken är att vi ska bli bättre rustade på att klara oss i det digitaliserade samhället genom att förstå hur digitala system är uppbyggda och fungerar. En annan anledning till Skolverkets införande av programmeringen är för att kunna svara mot framtidens arbetsmarknadsbehov. Idag kan man läsa särskilda kurser i just programmering med särskilda lärare men den stora förändringen är att programmeringen ska knytas till befintliga ämnen som matematik och naturvetenskapliga ämnen. Precis som alla andra länder gör ett val och integrerar programmeringen i olika ämnen så har Sveriges skolmyndighet också gjort detta val (ibid.). Enligt Skolverket (2018c) så erbjuder kombinationen programmering-matematik ämnesmässiga synergieffekter. Det kan t.ex. handla om att utforska matematiska strukturer med hjälp av programmering eller att använda matematiska kunskaper för att utveckla digitala produkter (ibid.).

3.2. Begreppsdefinitioner

3.2.1. Problemlösning

Enligt Skolverket (2011) är en problemlösningsuppgift en uppgift som är utmanande och kräver ansträngning. Uppgiften kan inte lösas rakt upp och ner som en rutinuppgift. Även Mouwitz (2007) hävdar att problemuppgifter inte går att lösa som en rutinuppgift, han menar då att när elever löser rutinuppgifter utvecklar de snarare sin *procedurförmåga* än sin *problemlösningsförmåga*. Enligt Skolverket (2011) innebär procedurförmågan ”*att tillämpa olika matematiska procedurer, rutiner så att säkerhet, precision och effektivitet stärks efterhand*” (s.1).

3.2.2. Problemlösningsförmåga

Att ha problemlösningen som ett *mål* av undervisningen innebär att eleverna ska utveckla en god förmåga i att lösa problem (Skolverket, 2011). “*Problemlösningsförmåga innebär att kunna analysera och tolka problem vilket inkluderar ett medvetet användande av*

problemlösningsstrategier som att t.ex. förenkla problemet, införa lämpliga beteckningar, ändra förutsättningarna" (Skolverket, 2011, s.2). Att lösa ett problem innebär att ha en process som är tillförlitlig och som visar att resultat blir korrekt utan att titta på svaret, men samtidigt få ut ett korrekt resultat av problemlösningen. Att lösa ett problem innebär också att kunna utvärdera både processen och resultatet, för att ibland få ändra om sitt arbetssätt. Att ha en god problemlösningsförmåga innebär också att själv, och tillsammans med andra kunna formulera och utveckla problem (Skolverket, 2011). Mouwitz (1999) påstår att de värdefulla problemuppgifterna är dem som utmanar eleverna genom att använda sina teoretiska kunskaper på ett nytt sätt som t.ex. genom att sätta matematiken i ett vardagligt perspektiv.

3.2.3. Problemlösningsstrategi

Pólya (1970) har utvecklat en process baserad på fyra steg vid problemlösning. Denna går ut på att först förstå problemet, sedan utveckla en plan som följs av att genomföra planen och sist att se tillbaka på lösningen. Polya understryker vikten av att läraren ställer många och väl genomtänkta frågor till eleverna under problemlösningsprocessen som exempelvis om hur man kan dela upp problemet i mindre delar. Lester (1985) påstår att läraren bör uppmana eleverna att se tillbaka på sin lösning, snarare än sitt resultat för att sedan förstå problemet och kunna applicera sina lärdomar på nya problem.

3.2.4. Programmering

Att skapa ett datorprogram är att programmera och ett datorprogram är ”ett antal logiskt ordnade stegvisa instruktioner som talar om för datorn vilka operationer den ska utföra när programmet körs” (Skolverket, 2018c, s. 3). Dessa stegvisa instruktioner finns färdiga i en sekvens som datorn följer i ett självständigt förlopp när programmet *exekveras*, dvs. när det körs. Instruktionerna som skrivs in i ett *programspråk* kallas för *kod* och för att datorn ska förstå koden så krävs det att den följer rätt grammatik, dvs. vissa skrivregler enligt en bestämd *syntax*. Om vi vill få datorn att utföra något specifikt så måste den veta vilka förutsättningar som finns, alltså måste den få någon slags indata och ett program som exekveras kan då även lämna utdata (ibid.). Dessa centrala begrepp i programmeringen beskrivs närmre längre ner.

3.2.5. Programspråk och programmeringsmiljöer

Programspråk är språk som man använder för att skriva kod (Skolverket, 2018c). Det finns både textbaserade programspråk som Pascal, Java, C++ och Python och visuella programspråk som Scratch, LOGO och Microbit. Visuella programmeringsmiljöer där det finns fördefinierade grafiska element eller bilder som sätts samman kallas för blockprogrammering, exempelvis i Scratch (ibid.).

3.2.6. Programmering som en strategi för problemlösning

Enligt Skolverket (2018c) kan programmeringen i några fall erbjuda nya möjligheter att lösa matematiska problem som eleverna är bekanta med sedan tidigare. Men om man inte från början arbetar med matematisk problemlösning i klassrummet så kan det vara svårt att ”kunna illustrera programmeringens förtjänster som matematiska problemlösningsverktyg” (2018c, s. 2). De menar också att den problemanalys som görs innan man börjar liknas vid den matematiska problemlösningen. Exempelvis som när man identifierar delproblem och löser dessa först innan man får lösningen på hela problemet. Detta problem behöver i sin tur presenteras i en logisk följs så att datorn förstår problemet (ibid.).

3.2.7. Algoritm

En algoritm är en uppsättning instruktioner som är stegvis ordnade med tydliga kategorier av indata och utdata som löser en väldefinierad uppgift (Skolverket, 2018c). Algoritmer är centrala i programmering då de beskriver handlingar på ett sätt som kan översättas till en kod som datorn förstår (ibid.).

3.2.8. Kod, syntax och semantik

En beskrivning av ett datorprogram i ett programspråk som kan tolkas av en dator kallas för kod. För att en programkod ska fungera krävs det att den skrivs korrekt och med rätt grammatik, med rätt syntax. Syntax kan variera mellan olika programspråk. Med semantik menas innehållet i programmet, dvs. vad programmet gör.

3.2.9. Iteration

Iteration är ett annat ord för upprepning (”Iteration”, 2018, 17 november). Inom matematiken och i programmering handlar detta om att en funktion eller process åstadkommer något genom att upprepa beräkningar eller andra operationer tills ett önskat resultat uppnåtts (ibid.).

3.2.10. Loop

En loop är en instruktion som itereras, det vill säga repeteras, och används för att vi ska slippa skriva samma del av ett program flera gånger (Skolverket, 2018c). En for-loop används inom programmering för att upprepa koden så att man slipper skriva in den igen (Vorderman, 2015). En while-loop upprepas tills ett visst villkor uppfylls. Villkoret kallas då för ett loop-villkor och är antingen sant eller falskt (ibid.).

3.2.11. Tinkering

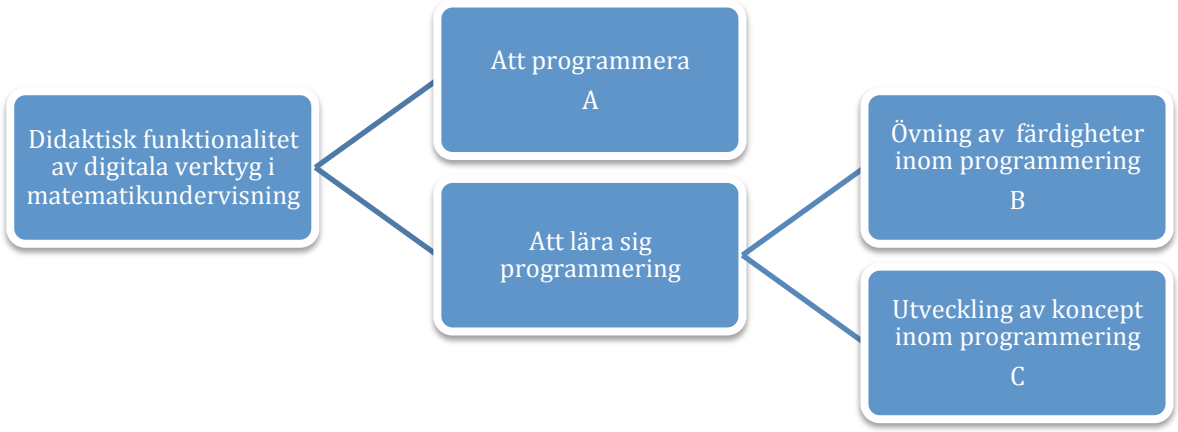
Att eleverna arbetar utforskande och prövar sig fram i sitt lärande genom att lära känna verktyg och begrepp benämns ofta som *tinkering* eller också *bricolage* (Skolverket, 2018d). Det betyder att man kommer underfund med något genom att mixtra med programmet (ibid.).

3.2.12. Simulering

Simulering är ett sätt att återskapa en verklighet i en kontrollerad miljö så långt det går (”Simulering”, 2018, 3 maj). Syftet med simuleringar är att användarna ska kunna dra slutsatser om hur saker verkligen är (ibid.).

4. Modell om didaktisk funktionalitet

Den teoretiska vinklingen i denna uppsats är en modell som urskiljer tre didaktiska funktionaliteter av digitala verktyg i matematikundervisning (Drijvers, 2018; Drijvers, Boon & Van Reeuwijk, 2011; Drijvers, 2015). Den didaktiska funktionaliteten ligger inte i egenskaperna hos det digitala verktyget utan handlar mer om vilket sätt som verktyget används på i undervisnings- och lärandeprocessen. I detta fall avses programmering som digitalt verktyg. I modellen (se figur 1) är den ena didaktiska funktionen att syssla med programmering (A) och urskiljs från den andra didaktiska funktionen som är att lära sig programmering för att öva på färdigheter (B) och utveckla koncept (B) inom programmering. Dessa två senare grenarna fokuserar mer på lärande, användandet av programmeringsverktyg kan leda till att man övar färdigheter och utvecklar koncept. Huvudsyftet är således inte att programmera utan att lära sig begrepp och utveckla färdigheter. Målet med digitala verktyg som t.ex. miniräknare var ursprungligen att frigöra eleverna från tekniska procedurer i syfte att låta dem fokusera på tolkning och problemlösning (Holyes, 2018). Digitala verktyg har mycket att erbjuda när det gäller övning av matematiska färdigheter, dels variation och slumpmässiga aktiviteter, automatisk och intelligent återkoppling men även en personlig miljö i vilken man kan göra misstag och lära från dessa (Drijvers, 2018). Just användningen av digitala verktyg för att utveckla koncept är den mest utmanande didaktiska funktionen men det största målet är att hjälpa lärare och pedagoger så att de blir medvetna om dessa typer av funktionaliteter som kan utnyttjas vid implementering av digitala verktyg i matematikundervisning (ibid.). När man programmerar så övar man sålunda på att skriva kod och få rätt syntax men syftet är att lära sig programmering för att utveckla färdigheter och begrepp inom programmeringen som exempelvis vad en loop innebär och vad den används till.



Figur 1 - Modell om didaktisk funktionalitet av digitala verktyg i matematikundervisning.

5. Tidigare forskning

I flera studier dras slutsatser om att arbete med programmering bidrar till att man utvecklar problemlösningsförmågan (Clements & Gullo, 1984; Clement, Lochhead & Soloway, 1980; Resnick et al., 2009; Saeli, Perrenet & Jochems, 2011). Enligt Skolverket (2011) innebär problemlösningsförmåga att man medvetet använder problemlösningsstrategier som exempelvis att: förenkla problemet, införa lämpliga beteckningar och ändra förutsättningarna. Dessa problemlösningsstrategier i programmering har framkommit i flera studier. Elever använder sig av problemlösningsstrategier som: att iterera händelser i programmeringskod, gör små ändringar och pröva att lösa problemet genom att gissa sig fram (Calder, 2010; Clements, 2002; Lang-Ree, 2016; Papert, 1980). En annan problemlösningsstrategi som framkommer i studie är att elever lägger märke till hinder och då modifiera sitt tillvägagångssätt (Papert, 1980). Papert (1980) menar även att strategier inom programmering liknar Polyas strategier vid problemlösning som t.ex. att man ställer sig frågan om problemet kan delas upp i mindre delproblem. Fler problemlösningsstrategier som uppfattas i studier är att elever förstår att de kan dela upp problemet i mindre delar som blir mer hanterliga (Lang-Ree, 2016; Papert, 1980) och även att dessa mer hanterliga delar ska lösas en åt gången för att sedan kombineras (Rich, Bly & Leatham, 2014; Clements & Gullo, 1984). Enligt (Calder, 2010) så erbjuder programspråket Scratch erbjuder inte bara kreativ problemlösning utan också logiskt resonerade.

Även om det finns studier som styrker att problemlösningsförmågan kan öka vid arbete med programmering så kvarstår dilemmat med att många lärare och elever saknar erfarenhet i programmering och detta har Skolverket (2018c) tagit hänsyn till vid programmeringens införande. I forskning nämns också att det finns ett behov av en god lärarkompetens som kan påverka hur programmeringen används i undervisning (Clements, 2002). Nu när programmeringen införts i kursplanerna för vissa matematikkurser så blir det krav på att programmeringen ska ingå i undervisningen. För de lärare som känner sig osäkra på att lära ut programmering medför detta problem som huvudmännen måste hjälpa till med att lösa (Larsson, 2017, september). Detta kan lösas genom att lärare får tid att dela sina verktyg med andra lärare, vilket är svårt ibland även om det finns många engagerade och kompententa lärare inom det området (ibid.).

I en artikel (Saeli, Perrenet & Jochems, 2011) beskriver de varför man ska undervisa programmering, vad det är som ska läras ut, vilka svårigheter i att lära ut programmering som finns och hur man ska lära ut programmering i gymnasieskolan. En anledning till varför man som elev ska lära sig programmering enligt två forskare inom datavetenskap, Seymour

Paperts och Alan Kay, är att detta leder till att man lär sig kraftfulla problemlösnings-, modellerings- och tankestrategier (Soloway, 1993). Detta sker eftersom att när eleven programmerar så måste denne först hitta en lösning till ett problem för att sedan reflektera över hur denna lösning ska kommuniceras med datorn genom användning av syntax och grammatik genom ett specifikt sätt att tänka (Papert, 1980; Szlávi and Zsakó, 2006). Det sistnämnda bidrar till att elevens naturliga språkförmåga utvecklas eftersom att det krävs att eleven lär sig att berätta, på ett tydligt sätt, vad de vill att datorn ska utföra (Hromokovic, 2006). Romeike (2008) hävdar att programmering handlar om problemlösning och om att skapa en programkod som en lösning. Du Boulay (1986) identifierar fem typer av svårigheter för inlärning och undervisning av programmering som beror på både motivation och tekniska aspekter. Elevers svårigheter är: 1) att förstå syftet dvs. att ta reda på vad programmering är användbart för och vilka fördelar som finns med att kunna programmera, 2) att förstå de generella egenskaperna av maskinen som de lär sig att kontrollera, 3) att förstå hur den fysiska maskinen beror på den fiktiva maskinen, 4) notation som inkluderar problemen som innebär olika formella språk som syntax och semantik, strukturer dvs. att förstå schema eller planer som kan användas för att uppnå små delmål (t.ex. när man använder en loop) och slutligen 5) att behärska programmeringens pragmatik (att lära sig förmågan att specificera, utveckla, testa och felsöka ett program genom att använda de tillgängliga verktygen).

Angående hur programmering bör läras ut, förutom att försöka förebygga de svårigheter som nämns ovan, är att fånga elevernas intresse på ett effektivt och engagerande sätt (Saeli, Perrenet & Jochems, 2011). Hromovic (2006) påstår att programmering ses som en kommunikationsförmåga genom att en programmerare på ett tydligt sätt ger instruktioner till en ointelligent dator. Om denna process äger rum genom att ett relativt enkelt programspråk används (t.ex. Python) som erbjuder enklare syntax än andra programspråk, kan elever fokusera mer på semantiken av programmet och göra mindre fel i syntaxen (Mannila et. al., 2006). Ett annat sätt att starta denna inlärningsprocess kan vara att använda praktiska exempel som att skriva om matlagningsrecept för en maskin som lagar mat (Hromovic, 2006). Detta sätt skulle hjälpa elever att först skriva ett enkelt program och sedan kombinera dem enkla lösningarna tillsammans för att få en lösning till mer komplicerade problem (Abelson & Sussman, 1996). Detta närmande har dubbelt syfte, att låta eleven uppleva den historiska utvecklingen men även att lära sig två koncept: modellering och återanvändning. En vanlig företeelse i programmeringsundervisning är att läraren först undervisar grunderna i programmering och sedan guida elever mot effektiva strategier för hela programmeringsprocessen (Ala-Mutka, 2004).

Av denna tidigare forskning kan man dra slutsats om att elever kan utveckla problemlösningsförmågan genom att använda problemlösningsstrategier som nämns i de olika studierna. Genom att identifiera och förstå elevers svårigheter som uppstår vid programmering kan man som lärare skapa en lärandemiljö som förebygger dessa svårigheter.

6. Metod

I detta avsnitt presenteras den valda metoden för undersökningen. Genomförandet av undersökningen, urvalet av undersökningsdeltagarna, validitet, reliabilitet och forskningsetik beskrivs och diskuteras.

6.1. Metodval

Eftersom att programmeringen infördes relativt nyligen och att många lärare saknar kunskap i hur man ska arbeta med matematisk problemlösning med programmering i enlighet med de reviderade styrdokumenten så valde jag att skriva om detta. I syftet beskrivs hur undersökningen kan användas och eftersom att det inte bara finns ett sätt att arbeta med programmering så formulerades två breda frågeställningar som förhoppningsvis ska ge svar på detta. Enligt Larsen (2009) förekommer det ofta att man väljer en bred frågeställning när det inte har genomförts mycket forskning på området tidigare. Eftersom syftet med studien är att undersöka hur man kan arbeta med programmering för matematisk problemlösning så har frågeställningarna formulerats som öppna frågor. Att dessa har formulerats med en viss öppenhet kan motiveras med att man kan arbeta med programmering på många olika sätt, det är bara upp till vad läraren anser vara lämpligt och var denne drar gränserna. Då det inte finns ett entydigt svar på frågan och svaret kan bestå av t.ex. olika verktyg och undervisnings-strategier ansåg jag att semistrukturerade intervjuer med ett antal verksamma lärare som har infört programmering i deras undervisning skulle ge en stor bild av hur man kan arbeta med detta. Undersökning var även en rimlig att genomföra under den givna tidsperioden.

Enligt Alvehus (2013) erbjuder intervjuer ett effektivt redskap för den kvalitativa forskaren då det finns utrymme att försöka förstå respondenternas förhållningssätt till deras agerande. Dokumentation av intervjuerna har skett med ljudinspelning för att underlätta faktasammanställningen och ge en bättre överblick genom att allt som blir sagt transkriberas. Genom att spela in intervjuerna så kan man underlätta dokumentationen under intervjuerna men det ställer även till en viss problematik. Det kan även påverka den intervjuade och begränsa respondentens öppenhet, om man å andra sidan bara skulle ta hjälp av sina anteckningar finns risken att det som sägs blir förändrat längs vägen (Alvehus, 2013). Om

man inte spelar in intervjun kan det även leda till att man inte får med alla detaljer i intervjun. Samtidigt som det fanns fördelar med att spela in intervjuerna så tog det väldigt lång tid att transkribera intervjuerna, som sedan skulle analyseras med ett teoretiskt perspektiv. Teorin som valts i studien är en modell som visar den didaktiska funktionaliteten av digitala verktyg och tar hänsyn till att arbete med programmering i matematikundervisning bidrar till utveckling av matematiska färdigheter och koncept som t.ex. problemlösningsförmåga och centrala begrepp i matematiken. Resultat och analysdelen är uppdelat i olika teman som följer intervjufrågorna i stort sätt.

Från början var tanken att skicka in intervjufrågorna (se Bilaga 2) till informanterna för att förbereda dem och för att få ut mer av intervjun men efter vidare eftertanke så ansåg jag att man kanske får mer spontana svar och att lärarna förmodligen inte kommer att lägga mycket tid på att förbereda sina svar, därför så valde jag att inte skicka in frågorna. Förutom frågorna i intervjuguiden ställdes neutrala frågor som: ”Kan du utveckla lite mer? Vad menar du med detta?” som enligt mig inte ska ha påverkat resultatet. I förfrågan via mail valde jag dock att ta med dem mest centrala frågorna så att lärarna kunde fundera på om dem kunde bidra med några bra svar till dem frågorna. Dem utvalda frågorna som skickades i mailet var:

- *Hur arbetar du med programmering i matematikundervisningen? (Undervisningsstrategier, tidsåtgång, programmeringsverktyg/språk)*
- *Hur tror du att programmering ska användas för att stärka elevers problemlösningsförmåga?*
- *Vilka möjligheter ser du med programmeringen? Vilka utmaningar finner du?*
- *Hur hanterar du elever som ännu inte lärt sig programmera?*

6.2. Urval

De valda intervjudeltagarna arbetar på gymnasiet och de fick förfrågan i ett mail först för att bekräfta att de kunde vara med i undersökningen. Urval skedde genom att jag kontaktade 5 lärare som arbetar på olika gymnasieskolor och som har arbetat eller arbetar just nu med programmering i matematikundervisningen och fyra av dem som svarade ville delta. Jag tog alltså reda på om deltagarna hade använt programmering med sina elever, även om det bara var lite grann. Två av de valda lärarna är förstelärare i matematik på skolorna de arbetar på. Men jag ansåg att informanterna kunde tas från både gymnasiet och högstadiet, alltså skulle man kunna intervjua någon som arbetat eller arbetar med programmering i matematiken på grundskolan i åk 7-9. En anledning till detta är att det finns en artikel från Skolvärlden (Larsson, 2017, september) där man skriver om en undersökning gjord av Lärarnas Riksförbund om att *både* högstadie- och gymnasielärare saknar kompetens i programmering.

En av lärarna som intervjuades har t.ex. arbetat med programmering på grundskolan tidigare frivilligt. Det är möjligt att kunskaper om hur man arbetar på högstadiet kan ge möjligheter för hur man kan arbeta på gymnasiet.

Under ett handledningsmöte så kom vi fram till att för en sådan undersökning är det bra om man involverar tre lärare minst men att fyra eller fem blir ännu bättre. Enligt Alvehus (2013) så beror antalet intervjuer på hur problemställningen är formulerad, så länge man får svar och bra underlag för sitt svar är det bra. I detta fall så är det egentligen ingen begränsning på hur mycket kunskap man kan få om arbete med programmering, däremot finns det inte utrymme för 10 intervjuer exempelvis. Jag har intervjuat två lärare från olika skolor och två lärare från samma skola men med olika erfarenheter för att få olika infallsvinklar. En risk om man genomför intervjuer på en och samma skola är att lärarna pratar ihop sig innan intervjuerna och att man får många svar av samma perspektiv, men i detta fall elimineras denna risk genom att jag har valt att intervjua lärare från olika skolor eller med olika erfarenheter. Om man undersöker hur lärare arbetar på olika sätt kan man öka den yttre validiteten som i sin tur kan leda till generaliserbarhet än om man bara intervjuar flera lärare från en och samma skola (Brinkkjaer & Høyen, 2013). Majoriteten av lärarna som intervjuades hade stor erfarenhet av att arbeta då tre av dem hade arbetat i minst 15 år och den fjärde läraren hade arbetat i fyra år. Alla lärare som intervjuades hade mer eller mindre genomgått en utbildning i programmering på universitetsnivå men endast två av lärarna hade fått en kort utbildning i programmeringsdidaktik och en av lärarna hade själv utbildat andra lärare i programmering och programmeringsdidaktik. Lärarna har programmerat tidigare i bl.a. Scratch, Microbit, Python, JavaScript, Modula, Matlab, C++.

6.3. Validitet och reliabilitet

För att diskutera om kvaliteten hos ett vetenskapligt arbete är högt brukar metoden diskuteras utifrån validitet och reliabilitet. Reliabelt kan uttryckas synonymt för pålitligt (Alvehus, 2013) eftersom att då kan arbetet upprepas och ge samma resultat. Att ett arbete har validitet betyder att det är giltigt och att man faktiskt undersöker det man ska och vill (ibid.) För att kunna uppnå studiens syfte krävdes det alltså att jag innan intervjuerna tog reda på om lärarna arbetar eller har arbetat med detta tidigare för att öka *validiteten* i undersökningen (Brinkkjaer & Høyen, 2013). T.ex. har samtliga matematiklärare i en av de undersökta kommunerna som tjänstgjort under läsåret 17/18 genomfört en utbildning i programmering och programmeringsdidaktik. Eftersom jag är intresserad av *hur* man kan arbeta med programmering så är det av stor relevans att undersöka hur några få erfarna lärare gör och ställa djupare frågor, därför har de intervjuade valts ut noga genom att de har någon större

kunskap i programmering än andra matematiklärare. En av lärarna hade dessutom ingått i den grupp som utbildade andra lärare i programmering och programmeringsdidaktik. För att resultatet av den kvalitativa undersökningen ska vara så givande som möjligt och svara på frågeställningen måste frågorna vara tydliga och fokuserade på undersökningsområdet eftersom att det är väldigt lätt att man börjar prata om annat som faller utanför ämnet.

Jag undviker medvetet att påverka undersökningen, genom att t.ex. att inte tala om för läraren att man kommer att ha ett specifikt lärandeperspektiv när man analyserar intervjun. Då kan man stärka studiens *reliabilitet* då studien kan ge samma resultat även om den upprepas (Brinkkjaer & Høyen, 2013). I analysen har jag valt att fokusera på *varför* lärarna gör som de gör och vilka argument de har för *hur* de gör. Teorin i studien har valts ut efter intervjuerna av den anledningen att denna teori framträdde i intervjuerna. Lärarna som intervjuades verkade alla tro på att elever lär sig genom att samarbeta genom att lösa problem tillsammans och få hjälp på vägen.

6.4. Forskningsetik

Studien är genomförd i enighet med de forskningsetiska principerna som delas upp i fyra krav: informationskravet, samtyckeskravet, konfidentialitetskravet och nyttjandekravet (Vetenskapsrådet, 2002). Det första kravet, informationskravet, har uppfyllts genom att intervjudeltagarna har blivit informerade om deras betydelse i undersökningen och att de när som helst kan avbryta deras deltagande. Denna information har getts till informanterna vid början av intervjutillfället då de fick läsa igenom en samtyckesblankett och skriva under denna (se Bilaga 1) vilket uppfyller samtyckeskravet. I denna samtyckesblankett informerades de om studiens syfte och att all information kommer att behandlas konfidentiellt och att all dokumentation kommer att förvaras oåtkomligt på Malmö Universitets server tills examensarbetet är godkänt för att sedan raderas och då är konfidentialitetskravet uppfyllt. I samtyckesblanketten informerades även informanterna om att all insamlad data endast kommer att användas i studiens forskningssyfte vilket slutligen uppfyller nyttjandekravet. Citat som har använts har avidentifierats så att ingen kan identifiera vem som har deltagit i studien och angivit informationen (Vetenskapsrådet, 2002). Även benämningen hen har använts vid referens till vad en lärare har sagt för att inte kunna knyta personen till ett kön. För att referera till de olika lärarna används beteckningarna L1, L2, L3 och L4.

7. Resultat och analys

I detta kapitel presenteras resultatet och intervjuerna analyseras med teorin som beskrivs i kapitlet teoretiska perspektiv. Kapitlet är uppdelat i teman som utgår från intervjufrågorna.

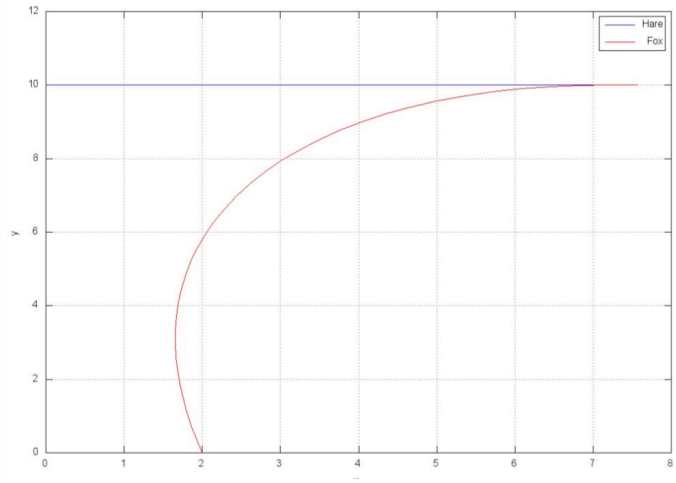
7.1. Programmering som en strategi för problemlösning

När lärarna blev tillfrågade om innebörden av att använda programmering som en strategi för problemlösning ansåg de att de visste med stor säkerhet. Det är lätt att lärare tolkar det på olika sätt eftersom att de har olika uppfattningar om hur man ska få in programmeringen i matematiken men de intervjuade lärarna verkade ha en gemensam uppfattning. Denna uppfattning var att eleverna skulle använda programmeringen som ett verktyg för lösning av problem som är av mer relevant karaktär eller problem som är svåra att lösa för hand.

L2 påstod att man ibland vill lösa problem som blir väldigt smidigt att lösa med hjälp av programmering istället för att göra det i ett kalkylblad, speciellt i tillämpad matematik som t.ex. i fysik. Det blir smidigare genom att man kan använda listor och vektorer men också att man kan använda if-satser för att lösa problem på ett annat sätt när man använder programmering än när man använder kalkylblad. I kalkylblad så får man göra det i många olika steg som medför att det blir oöverskådligt. L2 har även använt programmeringen i fysik-undervisningen i kursen fysikspecialisering eftersom att läraren anser att det är bra om de har med sig kunskaper i programmering när de läser vidare på universitet. I fysikspecialisering har läraren upplevt att dem har löst problemen enklare med programmering.

L3 har en tolkning av vad som menas med att man ska använda programmering som en strategi för problemlösning men att det nödvändigtvis inte behöver vara rätt tolkning. Hen tror att eleverna i grundskolan ska lära sig matematik via programmering men på gymnasiet ska eleverna använda programmeringen i problemlösning. Programmeringen ska vara en del av problemlösningen *och* modelleringen, det får man inte glömma bort, eftersom att det är väldigt lätt att fokusera på problemlösningen och glömma bort modelleringen. L3 tror att det handlar om att hitta problem som kan lösas med hjälp av programmering och gav ett exempel på en uppgift: Visuell beskrivning av en jaktkurva. Då är det ett rovdjur, en räv, som är på jakt efter ett byte, en kanin, och dessa befinner sig inte på en rät linje efter varandra utan de kan vara på två godtyckliga ställen. När räven får syn på kaninen så börjar den att springa mot kaninen och efter ett tag så upptäcker kaninen räven som gör att den börjar springa mot sitt gömställe. Kaninen tar den snabbaste vägen till hålan med en accelererad rörelse och räven försöker hela tiden att springa mot den plats där kaninen befinner sig och då är frågan: Hinner räven ikapp kaninen? Här kommer modelleringstanken in, om man vet hur snabbt båda djuren

rör sig och man gör vissa antaganden, t.ex. om hur deras hastigheter varierar, kan man ställa upp ganska komplicerade differentialekvationer. Men det behöver man inte göra, genom att använda räta linjens ekvation och Pythagoras sats så kan man bygga en stegmodell. I denna kan man se hur snabbt dem springer, hur långt kaninen måste springa innan räven hinner ikapp och då kan man se att räven kommer ikapp när kaninen t.ex. har sprungit 15 m och om hålan var 10 meter bort, så fick alltså räven inget byte. I simuleringen så kan man se hur djurens rörelse, alltså jaktkurvan, ser ut när den byggs upp från början till slut.



Figur 2 - Här visas den visuella jaktkurvan, simuleringen är ritad i Python.

L3 berättar att denna typ av uppgift lämpar sig i t.ex. Ma2 och då måste man som lärare förstå när det är vettigt att använda programmering, precis som när det blir vettigt att använda miniräknare. Det är inte vettigt att använda miniräknare när eleverna ska utföra multiplikationen 2·3, det är inte det som är syftet. Om man tänker just på modellering så passar det bra att använda programmeringen för att göra vissa simuleringar som t.ex. att simulera tärningskast eftersom att det är tidskrävande att kasta en tärning 10 000 gånger. Men med hjälp av en dator så kan vi simulera det väldigt snabbt. Bara på några sekunder får vi ett resultat.

Det finns alltså en underförstådd tanke med att vi ska arbeta med större datamängder eftersom att det är då det blir vettigt med miniräknare eller programmering. Det är även vettigt när man vill göra enkla beräkningar många gånger som när man vill undersöka och hitta primtal. Med programmering kan primtalsfaktorinsering göras mycket snabbare jämfört med en människa. Men det gäller ju även att det blir någon slags problemlösning av det och inte bara att man utforskar ett begrepp med hjälp av programmering. Programmeringen kan även användas för att utforska matematiska begrepp, det finns inget i kursplanen som säger att man inte kan göra det eller att det är fel. Men som lärare lurar man sig själv om man tror att det räcker med att undersöka ett matematiskt begrepp med programmering och sedan anta att

man är färdig enligt L3. Om man kopplar detta till den teoretiska modellen så avser lärandet med programmering att man utvecklar begreppsförståelse men då har man inte arbetat med problemlösning som är syftet med införandet av programmeringen.

7.2. Att utveckla elevers problemlösningsförmåga med programmering

L1 anser att om eleverna hade haft tillräckliga kunskaper i programmering innan de t.ex. började läsa Ma2c så kunde man arbeta med problembaserade uppgifter som kan lösas för hand men som kanske också är svårare att göras för hand. Men detta kräver fortfarande en hög nivå på programmeringskunskaper som elever i detta skede ännu inte behärskar betonar L1. Detta kan förklaras med modellen om didaktisk funktionalitet, för att utveckla färdigheter och koncept inom programmering krävs det att eleverna har lärt sig programmering och för att detta ska möjliggöras så krävs det att eleverna får syssla med programmering.

L2 beskriver att när man ibland strukturerar upp ett problem, så vet man hur man ska lösa det. I vissa fall om man använder miniräknare för att beskriva någonting så kan det bli onödigt klumpigt men med programmering blir det mycket lättare att lösa vissa problem om man använder if-satser och funktioner som upprepar händelser. Exempelvis sådant som ska upprepas så att man kommer närmre någonting, dvs. iteration av olika slag. L2 berättar att elever som lyckas lösa problemen i Python påstår att logiken är samma som när de löser ett problem för hand men de upplever att om de ska upprepa något många gånger så blir det mycket smartare med programmering. Men det beror mycket på var eleverna befinner sig kunskapsmässigt, de elever som hade programmeringsvana sedan tidigare gick det mycket bättre för och läraren har då sett att dem tänker mer generellt, dvs. dem gör inte lika mycket ”testa lösningar” som man annars gör om man kommer direkt från grundskolan. Eleverna förstår programmeringens förtjänster när de ska upprepa beräkningar många gånger.

L4 anser att det utvecklar problemlösningsförmågan genom att eleverna förstår att de måste använda speciella metoder för att komma till nästa steg i sin lösning men det är också en motivationsfråga. Om eleverna tycker att problemen är intressanta så kommer dem att vilja lösa fler eller att lära sig mer. I programmeringen så blir det, precis som i matematiken att man använder sig av systematiska algoritmer och stegvist tänkande.

7.3. Att undervisa i matematik med programmering

Här beskrivs hur lärarna undervisade och hur eleverna fick arbeta med programmering. Beskrivningen är uppdelad i tre delar: programspråk (1), undervisningsstrategier och tidsåtgång (2) och exempel på uppgifter och aktiviteter med programmering (3).

7.3.1. Programspråk

Bland tre av fyra intervjuade lärare så var Python det självklara valet av programspråk eftersom att de flesta använder det. L4 hade använt Scratch på högstadiet i teknikämnet. Lärarna ansåg att Scratch är mer lämpligt att användas i grundskolan eftersom att det är väldigt grundläggande och enkelt att använda. Här ser man att lärarna anpassar programmeringsverktyget till elevernas ålder och vad dem tror att de klarar av vilket kan ses som en anpassning av läraren för att de ska lära sig programmering och utveckla koncept.

7.3.2. Undervisningsstrategier och tidsåtgång

L1 som är förstelärare och använde programmering i Ma1c och Ma2c utgick från att eleverna aldrig hade programmerat och prövade att ge eleverna en enkel kod på två till fem rader som utförde något och bad eleverna att tolka koden genom att använda sina matematiska kunskaper. Syftet med detta var att eleverna skulle försöka förstå vad koden visar och vad den gör. Frågor som läraren ställde till eleverna kunde vara: ”Vad händer här? Vad skulle hänt om vi skulle ändra bara en sådan här liten grej?”. Elever som var duktiga i matematik kunde lista ut vad koden gjorde men det fanns elever som började ställa frågor som: ” Varför skriver man så?” och ” Vad betyder det?” och då fick läraren berätta att dem skulle vänta med det. Läraren ville inte att eleverna skulle fokusera på syntaxen i Python eftersom att de inte kunde det, utan fokusera på matematiken. I den teoretiska modellen urskiljs själva programmeringssysslan och lära sig programmering. Om eleverna inte får tid att själv programmera blir det svårt för dem att lära sig programmeringen så att dem kan få förståelse för koncept inom programmering som t.ex. vad en loop är eller hur det fungerar. För att eleverna ska lära sig programmering så krävs det att eleverna får programmera. Förvirringen hos många elever gjorde att L1 kände att eleverna tjänade mer på att diskutera problemet ur matematisk synvinkel och försöka lösa det med hjälp av sina matematiska kunskaper istället för med programmering vilket resulterade i att läraren valde att vänta med programmeringen. Men vid de tillfällen som L1 försökte få in programmeringen så valde hen att inleda med någonting som dem kunde fortsätta med i de kommande lektionerna, t.ex. när de började på ett nytt kapitel eller avsnitt till exempel så att de inte tar något helt annat centralt innehåll än det man skulle behandla enligt planeringen. L1 använde 15 min av varannan eller var tredje lektion,

till att visa koden för eleverna och försöka diskutera vad koden gör, men inte mer tid än så. Genom att göra detta i små korta sekvenser successivt så kunde läraren bygga upp en långsam acceptans för programmeringen och eleverna fick tid att vänja sig vid konceptet och möjliggjorde att de skulle få utrymme att utveckla sin kunskap vid nästa tillfälle vilket också leder till att koncept utvecklas enligt den teoretiska modellen.

Ett annat sätt att arbeta med programmeringen är instruktionsinriktat, dvs. att läraren börjar med ett tomt programmeringsblad och talar om för eleverna vad de skall skriva in. Detta hade tre lärare testat i sina klasser och här var det ett sätt för läraren att tala om för eleverna hur det fungerar och förklara processen steg för steg. Men kollegor till L1 var rädda att förlora den dyrbara tiden genom att elever av misstag skriver in fel eller missar ett kommatecken, ett semikolon eller helt enkelt en ny rad. Eftersom att klasserna är på ca 30 elever oftast så blir det lätt att mycket tid försvinner på att man försöker rätta till dessa småfel. Men så småningom fick eleverna större kunskaper och förståelse för att både läsa och skriva kod mer. Här blir det tydligt att när eleverna får programmera så lär dem sig programmering som i sin tur leder till att dem kan utveckla färdigheter inom programmering som att skapa villkors-satser.

L2 hade upptäckt att eleverna som läste Ma1c inte kunde multiplikationstabellen så bra vilket läraren såg som ett bra skäl till att börja använda programmeringen. Eleverna fick skriva egna program som de kunde öva på multiplikationstabellen med. Här användes programmeringsverktyget för att öva på färdigheter som multiplikationstabellen, dessutom var det bra för eleverna att få direkt feedback varje gång de matade in svar på multiplikationen. Direkt återkoppling ses enligt Drijvers (2018) som en intelligent egenskap hos digitala verktyg. Eleverna fick lösa andra enkla problem senare genom att läsa in saker från skärmen och lära sig göra enkla beräkningar. Men eftersom att de flesta elever inte kunde programmera så kom de ingenstans kunskapsmässigt tyckte L2. Även här tyckte L2 att eleverna hade svårt att öva färdigheter inom programmering eftersom att de inte behärskade programmeringen fullt ut. L2 tog in programmeringen under en lektion varannan vecka istället för att arbeta med ren problemlösning som läraren vanligtvis brukar låta eleverna göra för att få ihop klassen lite där syftet mest är att göra det till en social verksamhet mer än att dem lär sig matematik. Detta gör att eleverna lär känna varandra mer när de får arbeta i grupper och bidrar till att det blir mycket lättare för eleverna att gå kvar i klassen. I början av dessa lektioner har L2 gått igenom de olika begrepp eller koncept som de kommer att stöta på i programmeringen. Genom att lära sig detta i förväg så ökar det chanserna för att eleverna förstår vad dem gör under arbetets gång. Efter genomgången har eleverna fått lösa mindre problem tillsammans i grupper och ibland så har läraren påbörjat en lösning och så har eleverna fått avsluta den.

Någon gång så har eleverna fått skriva av en kod och sedan har läraren låtit dem fundera över vad som ska ändras i programmet för att de ska få ett speciellt resultat, dvs. tinkering. De uppgifter som L2 har gett eleverna har utvecklats av läraren själv. Förutom uppgifterna så har läraren gett eleverna instruktioner om exempelvis hur man startar programmet och hur man öppnar ett dokument i Python eftersom att detta tar lång tid om de inte har programmerat innan. L2 upplevde inte att det blev någon tidspress i slutet på kursen i Ma1c eftersom att klassen har matematik fem dagar i veckan varav fyra dagar arbetar de med bokens innehåll och den femte dagen med programmering eller problemlösning. L2 menade att det är lättare att läsa kursen snabbare, vanligtvis så har man kursen fem dagar i veckan med läraren trycker ihop kursen eftersom hen vill att eleverna ska blir snabbare på att räkna och vänja sig vid den höga takten när dem kommer till gymnasiet. Anledningen till att eleverna får läsa teorin snabbare är för att de ska hinna med problemlösningen eller som det senaste året: både problemlösning och programmering.

L3 har använt programmeringen i ma-specialisering där läraren fick tid att anordna en snabbkurs i programmering för eleverna eftersom kursen är lite mer flexibel. Först fick de lära sig programmera i Python under ett visst antal lektioner och sedan har dem arbetat med problemlösning i t.ex. sannolikhetsmodellering. Många av eleverna har Chromebooks och då har dem fått programmera i Google collaboratory som fungerar på samma sätt, rent tekniskt så är det en jupiter notebook med Python. Där finns en variant av programmering precis som det finns en variant av att skriva dokument, kalkylark osv. När de har arbetat med problemlösning har de simulerat skogsbränder och sett hur branden kan sprida sig till olika delar av ett område uppdelat i rutnät. Då kan man se hur stor sannolikheten är att det ska sprida sig från en brinnande ruta till en icke-brinnande ruta. L3 berättade att syftet med detta inte var att eleverna skulle lära sig programmera egentligen, för det gör man inte på dem 10-15 timmarna som dem använde till detta, däremot så får dem förhoppningsvis tillräckligt mycket kunskaper att kunna lära sig att förstå en kod. Här ser man tydligt att L3 satt värde på dessa kunskaper och valt att introducera grunderna och så att eleverna har en grund att stå på innan dem börjar lösa problem. I Drijvers modell (2018) är själva programmeringen en del av den didaktiska funktionaliteten. När eleverna programmerar lär dem sig alltså programmeringen.

När eleverna har programmerat vid några tillfällen då så har L3 använt ”en klassisk lärarstrategi” enligt hen som innebär att dem programmerar tillsammans, dvs. lärarens skärm är projicerad på tavlan och skriver kod som eleverna då skriver av. När de testade att köra programmet så fanns det några som inte fick det att fungera och då fick man hjälpas åt, lärare och elever, för att se vad de hade gjort för fel.

Eleverna har även fått arbeta med tinkering, dvs. att de ändrar i programmet och ser vilket resultat dem får. Detta kan man göra när man väl har en fungerande ramstruktur enligt läraren. Enligt läraren så blir det diskret i programmeringen eftersom att det är ganska enkel matematik. Hen menar att det inte är svårt att få eleverna att programmera en kod som beräknar en summa. L3 har under 10-15 timmar låtit eleverna lära sig att förstå kod och programmera och ytterligare 10 timmar har använts för att programmera lösningar till olika problem som beskrivs under nästa rubrik.

L4 hade använt Scratch och utgick från ett material som gick att hämta från nätet som hette *Internetguiden #70 Scratch*. Läraren hade en kort genomgång med klassen i åk 9 och sedan fick de sitta och arbeta med detta på egen hand och med tillgång till hjälp från både lärare och klasskamrater. Här är det tydligt att läraren anser att eleverna gynnas av att själv programmera för att lära sig programmering. Materialet behandlade de olika grundläggande begreppen inom programmering som loop, kod osv. Att de lär sig behärska olika begrepp inom programmeringen med ett program på grundläggande nivå gör att eleverna har lättare för att öva på färdigheter i programmeringen. Eleverna leddes igenom materialet stegvis där de fick testa små saker som t.ex. att få en katt på skärmen att jama eller liknande. Efter ca 3-4 lektioner med detta så skulle eleverna skapa ett eget spel ett eller enkelt program som skulle uppfylla ett antal kriterier enligt en matris. Genom att göra väldigt grundläggande saker i programmering för att sedan skapa och designa ett eget program fick eleverna möjlighet att utvecklas i deras egen takt. Hela teknikprojektet genomfördes under ca tre veckor totalt i flera klasser eftersom att lärarna ville testa programmering i sin undervisning, detta var inte infört i kursplanen när det testades. Många av lärarna kunde inte själv programmera men gjorde det i takt med att eleverna lärde sig det. Att lärarna också fick programmera leder också till att de lär sig programmeringen och då kan fokus läggas på de två andra didaktiska funktionaliteterna. Eftersom det inte är så avancerat så var det enklare och gick fortare för lärarna än eleverna.

L4 ansåg att även om det inte var just i matematik så handlade det mycket om att göra saker i rätt ordning, vilket är en vanlig företeelse i matematik. Det fanns även vissa villkor som bestämde att en händelse måste ske ett antal gånger innan en annan händelse kunde ske. Eftersom att det inte var infört i kursplanerna för något ämne då så valde läraren att koppla projektet just till kunskapskraven i ämnet teknik. Instruktionerna för elevernas projektdel var utformat av L4 som kollegorna fick ta del av. Matrisen som användes var till för att styra eleverna lite och dem fick även en mall för hur en teknisk rapport skulle skrivas då dem följa den när de skulle förklara sin arbetsprocess. L3 berättade att hen inte ville att eleverna skulle fokusera på betyg utan mer på att lära sig programmeringen. I uppgiften skulle eleverna skriva

en idébeskrivning där de definierar en målgrupp som programmet riktar sig till. Eftersom att eleverna kan hitta koder på nätet så blir det svårt att veta vad dem har gjort själv och vad som har hittats på nätet. Därför visade läraren dem hur man tar små skärmdumpar av sina kodstycken som skulle klippas in i ett dokument. De skulle välja några utdrag som de tyckte var centrala och förklara vad koden gjorde på ett översiktligt sätt på max 3-4 A4-sidor. Eleverna fick skriva i Google dokument så att läraren kunde följa deras arbetsprocess och kommentera den löpande texten. Även om eleverna hade hittat koden på nätet så fick de förklara vad den gjorde så att de fick förståelse. Programmen som eleverna gjorde behövde inte vara spel men många gjorde det eftersom att dem tyckte att det var kul. Några gjorde ett program i form av en informationsbank kring olika energiformer som fungerade som en slags plugghjälp. Att arbeta med detta var en vinst enligt läraren genom att dem fick arbeta med att angripa problem systematiskt. Även om eleverna hade ”stulit” koden från nätet så blev de tillfrågade och tvungna att förklara när läraren misstänkte något, trots att det var stulet så lärde de sig att förstå det som de hade med i sin redovisning. Även om koden var stulen fick dem skriva den i programmet vilket leder till att de imiterar och programmerar vilket i sin tur förhoppningsvis leder till att de lär sig programmeringen och dess koncept.

7.3.3. Exempel på uppgifter och aktiviteter med programmering

Lärarna som deltog i intervjun hade dels hittat på egna uppgifter men även använt uppgifter som fanns att tillgå från de olika utbildningarna som de fick eftersom att kurserna i programmeringsdidaktik handlade om att lärare skulle få material att tillämpa i sina klasser. Om uppgifterna ansågs vara svåra så modifierades dessa av lärarna så att de kunde bli mer tillämpbara i deras undervisning. Dessa kunde även modifieras så att man kunde få svårare varianter när eleverna väl var redo för nästa utmaning. L2 tycker att de uppgifter man väljer ut för att lösa med hjälp av programmering måste vara problem som kan lösas generellt, för precis som att ett matematiskt problem kan lösas på flera olika sätt så kan ett problem lösas på många olika sätt även med programmeringen. Hen menar att det är ointressant lösa ett problem med programmering där man ska beräkna kostnaden för en tröja när den ökar eller minskar med ett visst antal procent. Däremot så vill man ha problem där man t.ex. ska undersöka vad tröjan kostar om procentsatsen ändras. Alltså måste man formulera problemet på ett visst sätt för att det ska bli smidigt att lösa det på ett generellt sätt och hitta en lösning som passar för alla olika förändringsfaktorer. Även uppgifter där eleverna kan rita upp en graf och undersöka något för att sedan tolka resultatet fungerar bra i programmeringen enligt lärarna.

En intressant simulering som man kan göra är att se hur sjukdomsspridningen ser ut i en vaccinerad population. T.ex. har man 50 % av populationen som är vaccinerade och dem är då markerade med blått. De som faktiskt blev infekterade under simuleringen markeras med rött och några i populationen blev skyddade för att de hade vaccinerade runt omkring sig. Men även 50 % av dem som råkade bli smittade var omringade av vaccinerade och det gjorde att smittan inte fördes vidare. Här kunde man se styrkan av en simulering, hur det sprids egentligen och att det då är slumpmässigt men om man hade gjort det flera gånger så hade det sett ut ungefär på det viset varje gång. Detta passar in när man behandlar statistiken i Ma2 enligt L3. Hen påstår att man vill ha någon form av verklighetsanknytning och att detta kan vara både yttermatematiskt, dvs. att man tittar på hur något används i samhället, men även inommatematiskt som innebär att man tittar på något viktigt inom just matematiken.

7.3.4. Material från skolverket, läroböcker eller forskning

L1 hade använt en lärobok: ”Programmering 1 Javascript”. Boken var ett tunt häfte med konkreta exempel där några uppgifter kunde tillämpas i undervisningen men det var långt ifrån alla. L3 påstod att om man använder uppgifter från läroböcker så väljer man den enkla vägen ut eftersom att dessa uppgifter inte fokuserar på just matematisk problemlösning.

L2 berättar att hen inte tar stöd i forskning i sin undervisning med programmering men att hen utgår från det som var ett behov när hen studerade på universitetet och försöker göra det så att eleverna får med sig alla verktyg dem behöver. Detta är en anpassning från läraren som är genomtänkt, dvs. det ingår inte i Skolverkets kursplaner men läraren har valt att göra detta så att eleverna inte får svårigheter att genomföra sina studier efter gymnasiet då dem har fått testa detta tidigare. L3 tyckte inte att Skolverkets moduler hade bra problemlösningsuppgifter, hen tyckte att det var mer fokus på proceduruppgifter i programmering. Däremot ansåg L3 att Skolverket hade ett bra häfte med ett antal problemlösningsuppgifter för varje kurs under deras konferenser.

7.4. Hur säkerställer man att eleverna lär sig det matematiska innehållet och inte bara att programmera

Eftersom att det är i ämnet matematik som programmeringen ska användas så är ju syftet att eleverna lär sig det matematiska innehållet och att programmeringen enbart ska vara ett verktyg för att lösa problem. Men för att eleverna ska kunna utnyttja programmeringens förtjänster så måste de först programmera för att kunna bygga upp färdigheter inom programmeringen för att sedan använda dessa i matematisk problemlösning. Det man kan göra för att säkerställa att eleverna har förstått poängen med uppgiften är att man som lärare

avslutar lektionen med en sammanfattning. Det skulle kunna vara att läraren diskuterar med eleverna om vad dem har gjort på lektionen och se till att uppmärksamhet fästs på det som är viktigast. Om detta då inte har skett så är det ett perfekt tillfälle att rätta till detta och uppmärksamma eleverna på det som är viktigt. Men detta ställer också krav på att allt ska stämma dvs. att lektionen går bra, att eleverna lyckas lösa uppgiften och att dem inte fastnar på fel i programmeringskoden enligt L1. För att säkerställa att eleverna inte bara fokuserar på programmeringen och ”glömmer bort” vad det är dem gör matematiskt så brukar L2 låta eleverna lösa problemet först för hand och sedan med hjälp av programmeringen. När de gör det på datorn sedan så ser dem att det blir mycket lättare att byta värden som dem lägger in och då ser de nyttan med det.

L3 anser att ett sätt att kolla om eleverna har förstått matematiken och inte bara programmeringen är att se om de klarar av att lösa problemet, precis som allt annat i matematiken. Hur kontrollerar man att eleverna har förstått vad derivata är? Jo, man kan låta dem derivera något uttryck eller beskriva vad det är men då kollar man begreppsförståelse. Däremot så kan man låta dem lösa ett problem där derivata ingår som en del av lösningen. Att man ska använda programmering för att lösa ett problem kan lika bra göras med hjälp av Geogebra eller något annat digitalt verktyg. Den didaktiska funktionaliteten enligt Drijvers (2018) fungerar på vilket digitalt verktyg som helst, när man använder det digitala verktyget lär man sig hur man använder det digitala verktyget för att öva färdigheter i matematik och utveckla koncept i matematiken.

Lärare ska alltså examinera eleverna i om dem kan använda programmering vid problem-lösning. Eleverna kan examineras genom att de lämnar in programmet som dem skrev som en lösning till problemet eller att dem får fram ett svar till uppgiften som dem skulle lösa. Om man som lärare ger eleverna ett problem som inte kan lösas analytiskt och eleverna löser den med programmering så har de förstått matematiken enligt en L3. Att kontrollera om eleverna har förstått matematiken bakom programmeringskoden anser läraren är mycket lättare än vad många tror, många ser det som att programmering är något helt annorlunda. Men det är bara att göra precis som innan, hur kontrollerade man att elever kunde lösa problem tidigare? Jo, man kollar deras lösning och deras svar. L3 menar att problemlösningen är fortfarande densamma, det är bara strategin som är ny och i programmeringen så kan lösningen vara en kod som kan se ut på olika sätt eftersom att det finns olika sätt att modellera upp ett problem på för att lösa det.

7.5. Hur ser man och dokumenterar en progression i elevernas problemlösningsförmåga?

Lärarna har inte sett att programmeringen har hjälpt med utveckla elevernas problemlösnings-förmåga i detta skede utan det har mest varit ett krångligt moment att få in i matematiken för dem som inte har kunnat programmera. Det har varit besvärligt eftersom att lärarna har fått många frågor som: ”Varför ska det vara ett kolon här?” vilket gör att de hänger upp sig för att dem inte får rätt på det. Men lärarna tror att det kan hjälpa eleverna om dem kan programmera sedan tidigare, så om 9 år när elever har fått programmera redan i första klass så kommer dem att vara proffs när dem kommer till gymnasiet och då kan man lösa svårare problem. Om de har programmerat så kan de fokusera på den andra didaktiska funktionaliteten som fokuserar på lärande. När de lär sig bemästra programmeringen kan de alltså fokusera på att utveckla förmågor och koncept.

L3 tror inte programmeringen kommer utveckla problemlösningsförmågan mer än något annat verktyg. ”*Som lärare kan man inte kasta in eleverna i detta och förvänta sig att de utvecklar deras problemlösningsförmåga*” menar L3. Det krävs planering och det handlar inte om att man med programmering ska utföra beräkningar för det kan man göra för hand. Om man bara gör beräkningar enligt en regel så tänker man inte igenom det man gör. Men hen anser att man i programmering kan påbörja problemlösningsutvecklingen om man först lär sig grunderna i programmering. Genom att programmera lär man sig teckensystemet i programmeringen och dem olika begrepp som finns och då blir det lättare utveckla färdigheter inom programmering. Att programmera kan innebära att man får en kod och då tänker igenom och diskutera koden så att man förstår vad som händer i koden, L3 menar att det också kan vara problemlösning. L3 påstår att även om lärare ser en progression i elevernas problemlösningsförmåga, hur vet man att det handlar just om programmering? Det kan man inte veta och enligt denna lärare så är inte huvudsyftet att de ska bli bättre på problemlösning. Syftet som hen ser är att eleverna ska lära sig det som matematiker använder sig av i verkligheten, alltså i livet utanför skolan för att lösa problem. Många av dessa problem löser forskare och ingenjörer genom att använda olika typer av simuleringar och programmering. Hen berättar om en bild som på senaste tiden fick mycket uppmärksamhet, den första bilden som visar på svarta hål, som det bakom ligger en fantastisk övning. Det handlar alltså om att skolans problemlösning ska efterlikna verksamheten utanför skolan.

Man kan självklart se en progression i elevernas problemlösningsförmåga när dem har läst flera matematikkurser men det beror mer på att deras matematiska verktygslåda har vuxit samtidigt som att dem har stött på fler problem. Ju fler problem man lyckas lösa på egen hand,

desto fler och svårare problem kan man lösa sedan fortsätter L3. Vinsten av att lösa problemet utan hjälp är således större. Däremot tror hen att dem kommer att stöta på mer intressanta problem som liknar problem i den verkliga världen när de har detta nya matematiska verktyg. L3 upplever att när problemen är mer intressanta för eleverna så blir det mer motiverade att lösa dem och detta kan gynna deras inlärning. En förklaring till varför L3 inte tror att programmeringen kommer att bidra mer till en utvecklas problemlösningsförmåga än något annat verktyg kan liknas vid någon som bygger ett hus:

”En hammare eller en spikpistol, hur kommer spikpistolen att bidra till att man bygger hus av bättre kvalitet? Det kommer dem ju inte att göra. De kommer att bidra till att det går snabbare, kanske även bidra till att man gör lösningar med spik som man inte kan göra enkelt med hammare. Med en spikpistol så kan du t.ex. lägga en platta mot taket och enkelt spika i den med pistolen. Någonting som du inte kan göra med en hammare, lika enkelt. Så jag tror snarare att vi kommer att kunna oss an mer komplexa och mer relevanta problem, problem som hade tagit för lång tid att lösas för hand.” – L3

Vinsten med programmeringsverktyget är att det dels går snabbare men även att man kan lösa andra typer av problem. När det gäller dokumentation av elevers prestationer i programmering, t.ex. när eleverna har fått lösa en större modelleringsuppgift, så kan eleverna enligt L3 få lämna in en motsvarighet till en labbrapport där man skriver hur man har gått tillväga för att lösa problemet och vilka antagandet som man gjort. Då kan man som lärare bedöma detta i lugn och ro, precis som i en labbrapport. Enligt den didaktiska modellen kan eleverna visa att de har lärt sig koncept inom programmeringen genom att ha lärt sig programmera. I en labbrapport kan man visa och förklara hur man resonerat.

L4 märkte att eleverna som vanligtvis brukar fastna på vägen när de löser ett problem i matematiken, många av dem kom vidare i sina lösningar med programmeringen och lyckades få det att fungera. Detta tyckte läraren tydde på att de hade gjort en slags resa i deras problemlösningsförmåga mycket kopplat till att de tyckte att det var roligt.

7.6. Hur hanterar man elever som inte kan programmera

För att hantera elever som inte har lärt sig programmera innan så har alla fyra intervjuade lärarna lagt upp undervisningen och utgått från att eleverna inte kan programmera och lärt dem grunderna. Några lektioner har lagts på att introducera antingen Python eller Scratch så att eleverna vet hur programmering fungerar och vilka olika grundläggande koncept som finns. Eleverna har då lärt sig hur man skriver in och hur man läser ut från programmet. Detta

är nog den vanligaste strategin som lärare använder sig av just nu då majoriteten av eleverna inte har någon tidigare erfarenhet av programmering. Denna undervisningsstrategi visar precis som modellen enligt Drijvers (2018), för att lära sig programmeringen och dess koncept så behöver man helt enkelt programmera. Men detta kräver också mycket tid, om man undervisar samma klass i matematik och fysik så kan man vara flexibel men sina undervisningstimmar enligt av lärarna, men även då är det svårt att hinna med. Lärarnas förutsägelser har stämt bra, eleverna har inte kunnat programmera men det har även funnits någon enstaka elev som har gjort det tidigare. Det vanligaste upplägget har blivit att eleverna har fått en färdig kod där dem ska berätta vad koden gör, tolka den och se vad som händer när man ändrar i koden. Även varianten där eleverna programmerar efter läraren, precis som när läraren löser ett matematiskt problem på tavlan vid traditionell katederundervisning. Läraren skriver och förklarar och eleverna skriver av. Här framkommer en annan undervisnings-strategi där eleverna också får programmera och utifrån detta lär sig programmera för att förstå koncepten inom programmering och utveckla programmeringsfärdigheter.

Men lärarna berättar att många elever har haft problem med att förstå noggrannheten i programmeringen. T.ex. att de inte är konsekventa när de skriver in heltal och decimaltal korrekt eller när det ska vara kommatecken eller punkt. Eleverna blir då ofta omotiverade och undrar varför detta spelar roll, de förstår helt enkelt inte varför syntaxen är så viktig. I Python kan det vara att de glömmer att tabba in koden när man vill använda if-satser, vilket man måste göra enligt programmet för att visa att detta är ett speciellt kommando. Programmet säger då till att något är fel och specifikt vad felet är. Detta är enligt Drijvers (2018) en automatisk och intelligent återkoppling som digitala verktyg erbjuder och även en personlig miljö där man kan göra misstag för att lära sig programmeringen.

7.8. Tips från lärarna om arbete med programmering

Ett tips som L1 bidrog med var att man, oavsett hur negativa eleverna är till programmeringen, som lärare fortsätter att få in programmeringen i undervisningen. Att våga stå på sig själv är viktigt för annars kan man ge upp och då är det lätt att man inte vågar testa nya saker i sitt klassrum. L1 påpekar även vikten av att dela sina tankar med andra lärare eftersom att det kan vara bra att få stöd av varandra i början. Det är oerhört viktigt att lärare delar med sig men det måste också ges möjlighet för det menar läraren. T.ex. så kan man ha en timme i veckan bara schemalagd för diskussioner mellan lärare med samma ämnen.

L2 tycker att det är viktigt att man själv blir duktig på att programmera innan man undervisar programmering, vilket gäller all undervisning egentligen. Dvs. om man är osäker så ska man inte gå in i ett klassrum och försöka lära eleverna att programmera om man inte

har förstått allt själv. L2 påstår även att man ska börja helt från början och inte förutsätta att eleverna kan någonting. Den didaktiska funktionaliteten gäller även för lärare, om de ska lära ut programmering krävs det även att de programmerar själva.

L3 nämner att man ska skapa en lektion som passar enligt en utvecklingsmodell som kallas för *The 5 E's* som står för e*ngage, explore, expand, enertiate* och *evaluate*. Denna modell är även bra att använda i andra sammanhang men just i programmeringen så blir en del av *explore*-fasen just att man programmerar och att man frågar sig: ”Vad leder detta till?” och ”Hur långt vi kommer vi?”. Sedan kan man förklara dem mönster man hittar i *explore*-fasen och utvidga det på något sätt för att sedan gå tillbaks till denna fas och utvärdera resultatet. Vid denna utvärdering så handlar det inte om att förstå vad eleverna tyckte om just denna lektion utan då handlar det mer om när man t.ex. har arbetat med primtalsfaktorisering, att ställa sig frågan: ”Kunde jag primtalsfaktorisera det här?”. Läraren menar då att utvärderingen är en form av kontroll av att eleverna har förstått syftet med lektionen. Om eleverna har gjort det har de förmodligen utvecklat någon färdighet eller utvecklat förståelse för koncept.

Ett tips av L4 var att googla på lösningar om man kör fast, det finns många hemsidor där bl.a. ett internetsamhälle som kallas för *stackflow* hjälper till med frågor om programmering. Även om man har skrivit ett program som inte fungerar så kan man få hjälp med att förstå vad det är som blir fel.

7.9. Vad lärare kan göra för att hjälpa eleverna att utveckla sin problemlösningsförmåga

L2 anser att läraren kan ge förutsättningar för eleverna kan utveckla problemlösningsförmågan genom att låta dem lösa problem i par eller grupp där de får möjlighet att redovisa, diskutera och jämföra lösningar.

L3 anser att problemlösning är en av dem få sakerna som vi inte kan lära ut till eleverna eftersom om man berättar hur de ska göra och tänka så har man löst problemet åt dem och då är det inte problemlösning längre. Men både L2 och L3 påstår att man som lärare kan hjälpa till genom att erbjuda eleverna öppna problem som kan lösas på flera sätt. Man kan även ge dem förutsättningar dvs. lära dem att såhär skulle man kunna tänka, säga att dem ska titta på vissa saker, visa på strategier så att dem ibland kan förenkla problemet till en tidigare variant. När man löser problem med programmering lär man sig också koncept och övar färdigheter inom programmering som kan förenkla lösningen av ett problem, loopar används t.ex. för att genomföra samma beräkning många gånger.

7.10. Möjligheter och utmaningar med programmeringen

En L2 påstår att de elever som redan är duktiga i matematik kommer att bli ännu duktigare om de får lösa svårare problem med hjälp av programmeringen. L3 säger att när man gör simuleringar så kan man med hjälp av programmeringen uppnå häftiga saker, saker som man inte har kunnat göra tidigare. Hen menar att programmeringen öppnar upp möjligheter för undervisningen genom att läraren kan välja ut relevanta problem att lösa.

L1 berättade om en kurs i programmeringsdidaktik där de fick massor med olika problemlösningsuppgifter som skulle tillämpas i praktiken men kursdeltagarna tyckte att uppgifterna som de fick var för svåra för deras elever. Lärarna fick se hur man kunde använda programmering i olika matematiska områden som: sannolikhetslära, algebra, grafer och funktioner. Det fanns en del intressanta uppgifter att arbeta med men trots detta så tyckte lärarna att det krävdes enorma kunskaper i både matematik och programmering för att lyckas lösa uppgifterna. Som lärare vill man inte verka okunnig och köra fast själv om man inte känner sig trygg med programmeringen berättade L1.

L1 tycker att det är mycket stoff som både lärarna och eleverna ska hinna igenom på en termin och då blir det svårt att få in programmeringen trots att det är tänkt att ett programmeringsmoment skulle kunna användas så att man inte behöver gå igenom ett matematikområde två gånger. Men en av de största utmaningarna var att eleverna är på olika nivåer, vilket ställde till det mest. Så då skulle man förlora den välbehövda tiden för att eleverna först skulle få grunderna i programmering för att sedan använda programmeringen som en strategi för problemlösning. Men enligt modellen är själva användandet av programmeringen en förutsättning för att eleverna ska lära sig det.

L1 påstod att det är svårt att välja en problembaserad uppgift som eleverna kan lösa med programmering, det kommer att kräva förståelse på hög nivå. Detta medför att lärarna måste sätta höga krav på elevernas matematiska kunskaper men även programmeringen och det är självklart att lärare ska ha höga förväntningar men verkligheten ser tyvärr inte ut så. Många av kollegorna till L1 av lärarna rädda att man förlorar mer än det lilla man vinner när man försöker få in programmeringen. L1 påstår att Skolverket alltid har goda tankar men det är svårt att genomföra det som står i kursplanerna med dem förutsättningar som nu finns.

L2 berättar att de elever som kunde programmera sedan tidigare har haft lite tråkigt emellan åt. ”*Om man hade haft mer tid och vetat att de kunde programmera sedan tidigare så hade man som lärare kunnat förbereda lite klurigare uppgifter och satt dessa elever i en grupp*” säger L2. Men eftersom den klassen inte fick mer än cirka sju programmerings-tillfällen så har det varit svårt att hitta uppgifter som passar just då. Det har även varit svårt att

sätta dem i en egen grupp eftersom att läraren har behövt dessa elever för att hjälpa dem andra eleverna för annars så hade det tagit helt stopp för dem.

Den största utmaningen som lärarna ser är att eleverna inte kan programmera. Men enligt L2 är den stora utmaningen även att få alla att uppnå en godtagbar nivå i matematiken eftersom att alla elever kommer från olika skolor där dem har gjort olika saker, som att ha använt olika programspråk kanske. Det är svårt att få alla att nå godkänt i matten eftersom att det finns en stor variation i klasserna, vissa elever når precis godkänt och andra får alla rätt på provet och då har alla elever haft A i grundskolebetyg. Läraren tror att precis som att det finns en stor variation i matematikkunskaperna så kommer det vara en stor variation i deras programmeringskunskaper när dem kommer från grundskolan om några år. Om man tittar på modellen för didaktisk funktionalitet så kan man tolka detta som att elever som inte kan programmera kommer att ha svårt för inlärning i programmering, det kommer att ta längre tid för dem att öva på färdigheter och utveckla koncept inom programmering.

7.11. Lärarnas tankar kring de olika programmeringsmiljöerna

Lärarna är eniga om att blockprogrammering är på lite lägre nivå och att det ska väcka elevernas lust och intresse. L3 hävdar att det är tillämpbart för alla när man vill introducera programmering och förstå strukturer i programmering, dessutom så gör arbetet i dessa att man undviker syntaxfel eftersom det är bara att klicka ihop blocken. Men blockprogrammering räddar inte från logiska fel som t.ex. vilken ordning man ska placera blocken vilket innebär att man kan fokusera på semantiken, dvs. innehållet istället för syntaxen. Nackdelen med blockprogrammering är att programmen blir ganska tungskrivna efter ett tag när nivån blir mer avancerad. T.ex. så är en kod i blockprogrammering som är en A4-sida lång mycket mindre om man skulle skriva den i text. Anledningen till att det tar plats och blir oöverskådligt är att man måste dra de förutbestämda blocken med en viss storlek och placera dem i en viss ordning. Ibland ska man ha vissa block till villkoren och då blir dem större i datorfönstret vilket blir osmidigt när man vill skriva större program.

Men när eleverna har kommit till gymnasiet så måste lärarna lägga programmeringen på en högre nivå. L1 påstår att man måste använda textbaserad programmering på gymnasiet och lägga programmering på en högre nivå för annars blir det som att lura eleverna. När eleverna ska studera vidare så kommer nivån vara mycket högre. Men självklart kan det vara bra att börja med en programmeringsaktivitet i t.ex. Scratch när programmeringen ska introduceras för eleverna. På hemsidan *www.code.org* erbjuds en timmes aktivitet med block-programmering som kallas ”*Timmen med kod”* där eleverna introduceras till de mest grundläggande koncepten inom programmering med dem vanligaste skrivsekvenserna. Detta

kan man göra i Ma1c eller på högstadiet eftersom att det är väldigt enkelt men också kul eftersom att det är utformat som ett spel. Efter varje övning, som i princip ser ut som ett spel, får de bekräftelse på att de gjort rätt. Även här används programmeringsvektyget som ett automatiserat och intelligent verktyg som ger eleverna respons med ledtrådar om vad som inte fungerar (Drijvers, 2018). Efter hela övningen får eleverna ett digitalt diplom för att ha utfört den och denna aktivitet kan vara bra att använda som introduktion för elever som aldrig programmerat tidigare. Just den aktivitet som L3 visade kallades för klassisk labyrint och handlar om att man på olika sätt ska få figuren från Angrybird att komma fram till grisen som har tagit dess ägg. Under aktiviteten så får man titta på olika videor vid olika tillfällen där mer eller mindre kända människor berättar om programmeringen och om konceptet som används i nästa nivå.

Om man vill att fågeln går ett steg framåt i så lägger man till ett ”gå framåt”-block och vill man se vad man egentligen har skrivit så kan klicka så att man kan se programmeringskoden. Nästa nivå innebär att man ska lägga till flera av samma block och i nästa nivå igen så får man använda en funktion som gör att ”gå framåt”-blocket upprepar sig ett antal gånger då det blir jobbigt att dra det blocken när man vill ha det 10 gånger kanske. I nivåerna efter så vill man vända fågeln i en viss riktning först innan den går eller att något ska upprepas tills man uppnår någonting, t.ex. att man går åt höger och sedan framåt, flera gånger kanske tills man når till grisen och inte ett bestämt antal gånger. Eller också kan man programmera att fågeln ska ”svänga till vänster om det finns en väg till vänster” eller helt enkelt ”sväng till det håll där det finns en väg”. Svårigheten i aktiviteten ökar för varje nivå och det finns totalt 20 nivåer.

När man gör dessa övningar så förstår man vikten av upprepning och man gör dessutom det avancerade steget som innebär att man har val, det enda som inte behandlas här är abstraktion vilket ingår i en modell i programmeringen. Modellen heter SARA och innebär att man har Sekvensen som ska utföras, Alternativ dvs. gå åt höger eller gå åt vänster, Repetition (for-loop eller while-loop t.ex.) och slutligen Abstraktion som innebär att man skriver en funktion som gör något och som kommer att göra det flera gånger. Det sista är det som inte finns med men det är sådant som kan göras när man går vidare i programmering berättar läraren. Aktiviteten är byggd så att eleverna efter varje nivå gör något som blir mer avancerat i nästa steg och därför blir det enkelt att utveckla nya koncept, som är en didaktisk funktionalitet av digitala verktyg. Dessutom så får eleverna stöd i form av text om dem behöver det, texterna kommer fram om eleverna kör programmet och de inte lyckas komma fram till rätt kod, återigen en egenskap som digitala verktyg erbjuder (Drijvers, 2018).

L2 hade även testat Scratch där eleverna fick programmera så att en liten mask eller annat objekt styrs i ett koordinatsystem så att den kan gå åt ett visst håll för att äta t.ex. Men L2

gillade inte det eftersom att det oftast slutade med att eleverna ville välja bakgrundsbild istället och leka i programmet istället för att göra det de skulle. Om fokus inte läggs på det som är viktigt riskerar lärare att bara använda vissa programspråk i undervisningen. Enligt L2 så är dem olika programspråken väldigt lika men det är småsaker som skiljer sig, t.ex. i vilken ordning variablerna ska sättas in i funktioner.

7.12. I vilka matematiska områden är programmeringen lämplig eller olämplig att användas

Lärarna har använt programmeringen i bl.a. taluppfattning. Tre lärare har använt programmering t.ex. när de behandlat primtal då det är mycket lättare att hitta t.ex. det 115:e primtalet med hjälp av ett program än att göra det för hand. Detta ställer inte så höga krav på matematiska kunskaper heller. Om man väljer i början att lösa uppgifter med programmeringen så är det lämpligt att det handlar om enklare matematik. Det kan bli ett problem om dem har svårigheter med både programmeringen och dessutom matematiken. Därför är det bra enligt modellen att först programmera och sedan när man lär sig programmeringen kan man öva på färdigheter och utveckla koncept. I statistiken tror L1,L2 och L4 att det är bra att lösa uppgifter med hjälp av programmering eftersom att det finns mycket data som är jobbig att hantera. L3 tycker att programmeringen kan användas på alla ställen där det finns problemlösning men att det kanske tar tid att hitta dem rätta problemen.

Lärarna tycker att det inte är lämpligt att använda programmeringen till att lösa uppgifter som utvecklar procedurförmågan. I algebra och ekvationslösning så vill L2 inte använda programmering. L3 anser inte att aritmetik passar i programmering men hen säger att under aritmetik så finns även moduloräkning och kryptering. Att skriva ett program som kan omvandla olika tal till en viss talbas är intressant och användbart i programmeringen i Ma1c.

8. Slutsats och diskussion

I tidigare forskning framhävs att arbete med programmering utvecklar problemlösnings-förmågan och att arbete med programmering syftar till att intressera elever samtidigt som man bör sträva efter att minska de svårigheter som elever upplevs få vid arbete med programmering. Studiens mest centrala frågeställning var att förstå hur verksamma lärare ute på fältet har börjat integrera programmering i undervisning för arbete med matematisk problemlösning. För att förstå deras arbete ansåg jag att det var viktigt att förstå hur de tolkade kursplanerna i matematik där programmeringen skulle ingå. Deras uppfattning var ganska lik den uppfattning jag hade. Lärarna har lagt ner tid på att lära eleverna grunderna eller åtminstone att förstå grunderna i programmering och de är medvetna om att det i framtiden ska fungera annorlunda eftersom eleverna ska ha lärt sig grunderna redan i grundskolan. Att fokus i programmeringsundervisning läggs på att undervisa grunder för att sedan applicera dessa kunskaper i programmeringsprocesser nämns även i tidigare forskning (Ala-Mutka, 2004). Lärarna är övertygade om att eleverna kommer kunna programmera när dem kommer till gymnasiet om några år. Men det kräver också att dem har lärt sig Python på högstadiet eller sett hur det fungerar i alla fall innan dem kommer till gymnasiet. Eleverna ska enligt Skolverket (2018b) fått testa olika programmeringsmiljöer vilket innebär visuell men även textbaserad programmering. När det gäller programspråk så verkar det som att Python används på många gymnasieskolor men Scratch har använts på högstadiet. Det blockbaserade programspråket Scratch innehåller däremot inte någon syntax utan fokuserar mer på semantik och logiskt tänkande och då blir nivån väldigt grundläggande och erbjuder en lekfull upplevelse (Resnick et al., 2009).

Det finns inte tillräckligt med tid för eleverna att lära sig programmera men eleverna ska lösa matematiska problem med hjälp av programmeringen som ett verktyg, precis som miniräknaren används. Därför har lärarna strategiskt valt att lägga upp undervisningen så att eleverna mer eller mindre först har fått en kort grundkurs i programspråket Python eller i Scratch för att sedan använda dessa kunskaper i matematisk problemlösning eller programmeringsprojekt. L1 hade inte gjort på samma sätt, utan visat eleverna en enkel kod och diskuterat denna utifrån deras matematiska kunskaper. Då har eleverna arbetat tillsammans med en gemensam skärm och diskuterat koden för att försöka förstå vad den gör. I stort sett så har lärarna testat olika undervisningsstrategier, t.ex. att låta eleverna programmera på sina egna datorer genom att i början få en påbörjad lösning men även få skriva hela lösningar till problem med hjälp av programmering. Både under och efter detta arbete har lärare samlat klassen för att antingen förklara något som eleverna har fått problem

med eller för att sammanfatta lektionen och gå igenom det som var viktigt så att eleverna får med sig detta. Eleverna har under dessa lektioner arbetat med enklare programmeringsaktiviteter men även löst genomtänkta uppgifter som är av mer komplex karaktär. Eftersom att Skolverket (2018c) är medvetna om att det är svårt att få in programmeringen i matematiken i detta skede så kan man arbeta med matematiska begrepp som t.ex. att undersöka primtal och detta har gjorts på de flesta lärares matematiklektioner. Skolverket understryker även att oavsett om man är nybörjare eller expert på programmering så handlar mycket av processen om att felsöka sitt program och rätta till fel (ibid.).

Resultatet har även visat på att lärare som har låtit elever programmera har stött på problem som att eleverna har svårigheter med notation och att förstå syftet med programmeringen, två svårigheter som Du Boulay (1986) också nämner i sin studie. Det är tydligt varför lärarna har gjort så som de gjort och trots att de flesta lärarna har försökt att lära eleverna programmering från början så har dem haft svårt med att förstå syntaxen. Detta kan bero på att eleverna behöver lära sig programmering under en längre tid innan det blir bra. Om L1 hade lagt några lektioner på att låta eleverna programmera och förstå syntaxen, alltså programspråket och grammatiken i språket så hade det kanske gynnat eleverna mer än att de bara får se koden eftersom att läraren upptäckte att eleverna hängde upp sig på själva koden.

Av studiens resultat kan man få inblick i hur man kan låta elever arbeta med programmering i matematikundervisning, genom att fortsätta hitta exempel på uppgifter och bli bättre på programspråket kan man erbjuda elever en bra grund att stå på inom programmering som ett verktyg för matematisk problemlösning. Eftersom att undersökningen har beskrivit hur fyra lärare på olika skolor och med olika erfarenheter har arbetat så kan det ge en liten bild av hur matematiklärare arbetar generellt så det krävs ännu mer studier kring hur lärare arbetar för att kunna generalisera. Alla lärare arbetar olika men man kan se liknelser i deras strävan efter att få eleverna att begripa programmeringen. L2 har låtit eleverna lösa några matematiska problem för hand och sedan med programmering och då kan man enligt Skolverket (2018c) förstå förtjänsterna med detta verktyg.

En tydlig och gemensam faktor i lärarnas arbete var att de alla fungerade som handledare för eleverna under lektionerna för att de skulle lära sig programmera. Att de gav utrymme för eleverna programmera och lära sig grunderna för att sedan lösa problem tyder på att lärarna ser själva användandet av programmeringsverktyget som en viktig byggsten i lärandet och därför ansåg jag att denna teoretiska vinkling passande bra som analysverktyg. Jag bedömde att den på ett bra sätt kan bidra till förståelsen av det problem som belyses i just denna uppsats eftersom den urskiljer de didaktiska funktionaliteterna av detta digitala verktyg. Både val av teori och undersökningsmetod men också hur man definierar olika begrepp bestämmer hur

examensarbetet har formas. Genom att ändra i dessa faktorer skulle ett annat perspektiv kunna fås. Eftersom det var svårt att veta hur lärarna arbetade i förväg och vilka teorier deras arbete baseras på valdes teorin för resultatanalys efter intervjuerna med lärarna. Så för att sammanfatta arbetet med programmeringen: elevernas bristande kunskaper i programmering medförde att det inte kunde implementeras på ett smidigt sätt men eleverna fick ändå mer förståelse för detta, speciellt de elever som hade programmeringsvana sedan tidigare. Resultatet pekar också på att det är svårt att hitta en lagom nivå på undervisningen för alla elever och att hitta bra problemlösningsuppgifter som passar. Men det finns hemsidor som erbjuder enkla programmeringsaktiviteter som är bra att inleda programmerings-undervisningen med. I kurser på universitetet som handlar om programmeringsdidaktik kan man även få inspiration till uppgifter och Skolverket har bidragit med uppgifter som kan användas i problemlösning. I tidigare forskning framkommer det att Scratch erbjuder, förutom kreativ problemlösning, logiskt resonerade (Calder, 2010) vilket stämmer överens med vad L4 och L3 påstår i resultatet.

I intervjuerna framkommer det även att lärarna tycker det är mycket viktigt att man som lärare kan programmera och vet vad eleverna kan stöta på för problem så att man kan hjälpa eleverna på ett bättre sätt vilket även påpekas i forskning (Clements, 2002).

Syftet med den andra frågeställningen var att få en inblick i hur de intervjuade lärarna arbetar just med att utveckla elevernas problemlösningsförmåga med hjälp av programmering. Tre lärare menar att de inte har sett att programmeringen har hjälpt till med att utveckla elevernas problemlösningsförmåga och att det kanske är tidigt att säga det nu med tanke på att de inte behärskar programmeringen fullt ut. Resultatet visar dock att en lärare upplever att de eleverna som hade programmeringsvana sedan tidigare använder strategier i problem-lösningen med programmeringen som visar på att de löser uppgifter mer generellt och inte testar sig fram. Här kommer Holyes (2018) tankar in som handlar om att frigöra eleverna från tekniska procedurer för att ge utrymme åt tolkning och problemlösning. Detta motsäger forskningen som pekar på att eleverna utvecklar problemlösningsförmågan genom att ”gissa och pröva” (Calder, 2010; Clements, 2002; Lang-Ree, 2016; Papert, 1980). Detta skulle vara intressant att undersöka ytterligare eftersom att både Skolverket och forskning (Papert, 1980) pekar på att programmering handlar om att testa sig fram och modifiera kod för att hitta lösningar. L2 märker också att eleverna tycker att det är smartare att lösa uppgifter där upprepning av beräkningar behövs. T.ex. kan det handla om att en funktion ska åstadkomma något genom att upprepa beräkningar tills ett önskat resultat har uppnåtts. Detta har även Calder (2010) uppmärksammat i sin studie.

En av lärarna tror inte att programmering kommer att utveckla elevernas problemlösningsförmåga på ett annat sätt än vad andra verktyg gör. En annan lärare hade däremot sett att många elever hade gjort en ”resa” i deras problemlösningsförmåga. Även om lärarna hade haft olika upplevelser kring om programmeringen bidrog till förbättring av problemlösningsförmågan var de överens om att programmeringen öppnar upp möjligheter för arbete med matematiska problem som är mer relevanta och intressanta att lösa. De ser också att det är möjligt att använda programmering i många olika matematiska områden, speciellt statistiken är ett mycket bra område eftersom att det blir lättare med just att hantera stora datamängder.

L4 anser att det faktiskt kan utveckla problemlösningsförmågan genom att eleverna får insikt om att de måste tänka stegvist och genomföra systematiska algoritmer. Att en sak måste hända innan en annan sak sker eller beräknas blir tydligt i en programmeringsprocess. Detta påminner om det som både Skolverket (2018c) beskriver men även Hromovic (2006), att programmeringsprocessen kan liknas vid ett matlagningsrecept där instruktioner ges om vad som ska göras. Även det stegvisa tänkandet kan kopplas till det som (Abelson & Sussman, 1996) påstår om att elever skulle ha nytta av att dela upp lösningen och lösa små delar av problemet för att sedan sätta ihop alla delar i en viss ordning. L4 anser att eleverna utvecklar problemlösningsförmågan i det avseende att eleverna förstår att de måste lösa problem i olika steg där problemet delas upp i olika delar precis som beskrivs i tidigare forskning (Rich, Bly & Leatham, 2014; Lang-Ree, 2016; Papert, 1980).

Två lärare anser att det är viktigt att intressera eleverna i arbetet med programmeringen för om problemen är intressanta så kommer eleverna att vilja lära sig programmera mer, något som nämns i forskning kring hur man bör arbeta med programmering (Saeli, Perrenet & Jochems, 2011). Lärarna säger gemensamt att de inte skulle vilja använda programmering i vissa matematiska områden men om det ingår problemlösning i programmeringen ser dem att det är bra. Även om inte just denna studie kan bekräfta ett arbetssätt som tydligt visar på att elever utvecklar problemlösningsförmågan så kan fortsatt undersökning av samma karaktär möjligtvis göra detta. Eller så är det precis som en lärare beskriver det, att programmerings-verktyget är ett av många verktyg som kan stärka elevers problemlösningsförmåga utveckla deras strategier vid problemlösning. Slutligen kan man säga att om eleverna inte ges tid till att arbeta med programmering så blir det svårare att lära sig detta vilket leder till att det bli en utmaning att öva sina färdigheter och utveckla koncept inom programmering.

8.1. Förslag till vidare forskning

Eftersom att programmeringens införande är i ett tidigt skede så krävs det fortsatt forskning kring hur lärare ska arbeta med detta i matematikundervisningen. Det skulle vara intressant att genomföra fler intervjuer med verksamma lärare i skolan så att man kan få en större bild av hur man kan arbeta med matematisk problemlösning med hjälp av programmering. Detta kan vara mer givande om man genomför studier om några år då lärarna förhoppningsvis har kommit igång ordentligt med detta arbete. Genom att undersöka vilka uppgifter elever kan arbeta med specifikt kan man få en djupare förståelse för hur dem kan utveckla sin problemlösningsförmåga. En annan strategi för att undersöka hur elever i både grundskolan och gymnasieskolan utvecklar sin problemlösningsförmåga med programmering skulle kunna vara att genomföra observationer i klasser där elever får lösa problem med just programmering. Konkreta strategier för hur de tar sig an problemet och hur de löser det skulle kunna frigöra fantastiska kunskaper och öka förståelsen för lärare.

9. Referenser

Abelson, H., Sussman, G.J. (1996). Structure and Interpretation of Computer Programs, 2nd edition. Series MIT Electrical Engineering and Computer Science.

Ala-Mutka, K. (2004). Problems in learning and teaching programming-a literature study for developing visualizations in the Codewitz-Minerva project. Codewitz needs analysis, 20.

Alvehus, Johan (2013). Skriva uppsats med kvalitativ metod. Stockholm: Liber.

Calder, N. (2010). Using scratch: an integrated problem-solving approach to mathematical thinking. Australian Primary Mathematics Classroom, 15(4), 9-14.

Brinkkjaer, U. & Høyen, M. (2013). Vetenskapsteori för lärarstudenter. (1. uppl.) Lund: Studentlitteratur.

Clements, D. H. (2002). Computers in Early Childhood Mathematics. Contemporary Issues in Early Childhood, 3(2), 160–181. https://doi.org/10.2304/ciec.2002.3.2.2

Clements, D. H., & Gullo, D. F. (1984). Effects of computer programming on young children's cognition. Journal of educational psychology, 76(6), 1051.

Clement, J., Lochhead, J., & Soloway, E. (1980, January). Positive effects of computer programming on students understanding of variables and equations. In Proceedings of the ACM 1980 annual conference (pp. 467-474). ACM.

Drijvers, P. (2015). Digital technology in mathematics education: Why it works (or doesn't). In Selected regular lectures from the 12th international congress on mathematical education (pp. 135-151). Springer, Cham.

Drijvers, P. (2018). Tools and taxonomies: a response to Hoyles. Research in Mathematics Education, 20(3), 229-235.

Drijvers, P., Boon, P., & Van Reeuwijk, M. (2011). Algebra and technology. In Secondary Algebra Education (pp. 179-202). SensePublishers.

Du Boulay, B. (1986). Some Difficulties of Learning to Program. Journal of Educational Computing Research, 2(1), 57–73. https://doi.org/10.2190/3LFX-9RRF-67T8-UVK9

Hoyles, C. (2018). Transforming the mathematical practices of learners and teachers through digital technology. Research in Mathematics Education, 20(3), 209-228.

Hromkovič, J. (2006, November). Contributing to general education by teaching informatics. In International Conference on Informatics in Secondary Schools-Evolution and Perspectives(pp. 25-37). Springer, Berlin, Heidelberg.

Iteration. (2018, 17 november). I *Wikipedia.* Hämtad 2019-05-14 från https://sv.wikipedia.org/wiki/Iteration

Lang-Ree, H. L. (2016). " Vi må tenke og ikke bare tegne": En kvalitativ studie om bruk av programmering som verktøy i arbeid med matematikk (Master's thesis).

Larsen, A.K. (2009). Metod helt enkelt: en introduktion till samhällsvetenskaplig metod. (1. uppl.) Malmö: Gleerup.

Larsson, Å. (2017, september). 8 av 10 lärare osäkra på att lära ut programmering. *Skolvärlden*, 9. Hämtad från https://skolvarlden.se/artiklar/8-av-10-larare-osakra-pa-att-lara-ut-programmering

Lester, F. (1985). *Teaching and learning mathematical problem solvning: Multiple Research Perspectives.* (Hillsdale, New Jersey: Lawrence Erlbaum Associates, Inc., Publishers.

Mannila, L., Peltomäki, M., & Salakoski, T. (2006). What about a simple language? Analyzing the difficulties in learning to program. Computer Science Education, 16(3), 211-227.

Mouwitz, L. (1999). *Dialoger om problemlösning*. *Nämnaren, 4*, 44-45.

Mouwitz, L. (2007*). Vad är problemlösning?*. *Nämnaren, 1*, 61-61.

Papert, S. (1980). Mindstorms: Children, computers, and powerful ideas. Basic Books, Inc..

Pólya, G. (1970). Problemlösning: en handbok i rationellt tänkande. Stockholm: Prisma.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., ... & Kafai, Y. B. (2009). Scratch: Programming for all. Commun. Acm, 52(11), 60-67.i

Rich, P. J., Bly, N., & Leatham, K. R. (2014). Beyond cognitive increase: investigating the influence of computer programming on perception and application of mathematical skills. Journal of Computers in Mathematics and Science Teaching, 33(1), 103-128.

Romeike, R. (2008, July). What's my challenge? The forgotten part of problem solving in computer science education. In International Conference on Informatics in Secondary Schools-Evolution and Perspectives (pp. 122-133). Springer, Berlin, Heidelberg.

Saeli, M., Perrenet, J., Jochems, W. M., & Zwaneveld, B. (2011). Teaching programming in Secondary school: A pedagogical content knowledge perspective. Informatics in Education, 10(1), 73-88.

Simulering. (2019, 3 maj). I *Wikipedia.* Hämtad 2019-05-21 från https://sv.wikipedia.org/wiki/Iteration

Skolverket. (2011). *Om ämnet Matematik*. Hämtad 2016-11-15 från:

https://www.skolverket.se/download/18.6011fe501629fd150a2893a/1530187438471/Kommentarmaterial_gymnasieskolan_matematik.pdf

Skolverket. (2017). *Kommentarmaterial till kursplanen i matematik*. Stockholm: Skolverket. Hämtad 2019-05-17 från https://www.skolverket.se/getFile?file=3794

Skolverket. (2018a). *Kursplan för gymnasieskolan*. Ämne - Matematik. Stockholm: Skolverket.

Skolverket. (2018b). *Kursplan för grundskolan. Ämne - Matematik*. Stockholm: Skolverket.

Skolverket. (2018c). *Om programmering i matematikundervisning*. Hämtad 2019-03-31 från 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

Skolverket. (2018d). *Att undervisa med programmering*. Hämtad 2019-03-31 från https://larportalen.skolverket.se/#/modul/1-matematik/Grundskola/438_matematikundervisningmeddigitalaverktygII_%C3%A5k7-9

Soloway, E. (1993). Should we teach students to program?. Communications of the ACM, 36(10), 21-25.

Szlávi, P., & Zsakó, L. (2006, November). Programming versus application. In International Conference on Informatics in Secondary Schools-Evolution and Perspectives (pp. 48-58). Springer, Berlin, Heidelberg.

Vetenskapsrådet (2002). *Forskningsetiska principer inom humanistisk-samhällsvetenskaplig forskning*. Stockholm: Vetenskapsrådet.

Vorderman, C. (2015). Hjälp ditt barn med programmering: en illustrerad guide som lär ut programmering steg för steg. Göteborg: Tukan.

Bilaga 1

NATURVETENSKAP-
MATEMATIK- SAMHÄLLE

På lärarutbildningen vid Malmö universitet skriver studenterna ett examensarbete på avancerad nivå. I detta arbete ingår att göra en egen vetenskaplig studie, utifrån en fråga som kommit att engagera studenterna under utbildningens gång. Till studien samlas ofta material in vid skolor, i form av t.ex. intervjuer och observationer. Examensarbetet motsvarar 15 högskolepoäng, och utförs under totalt 10 veckor. När examensarbetet blivit godkänt publiceras det i Malmö universitets databas MUEP (http://dspace.mah.se/handle/2043/599).

Datum

Samtycke till medverkan i studentprojekt

Mitt namn är Nour-Mariam Merhi och jag läser min sista termin på ämneslärarprogrammet på Malmö universitet, med planerad examen i juni. Jag har inhämtat skolans godkännande till att genomföra studien som beskrivs närmare nedan:

Jag skriver nu mitt examensarbete i matematik som är mitt fördjupningsämne där jag vill undersöka *hur* verksamma lärare arbetar med programmering som en strategi för problemlösning i matematikundervisningen genom intervjuer, eftersom att detta ingår nu i den reviderade kursplanen. Syftet med undersökningen är *att belysa hur lärare, nyexaminerade såväl som erfarna, kan arbeta med problemlösning genom att integrera programmering i matematikundervisningen.* Vid analys av intervjuerna kommer jag att fokusera på *hur* lärare väljer att arbeta med att utveckla elevers problemlösningsförmåga via programmering och *förstå* de didaktiska val som de gör. Ett exempel är hur de väljer att hantera elever som ännu inte lärt sig programmera eftersom att det är ett aktuellt problem som många lärare (inklusive mig själv) kommer att få ta itu med. Deltagarna i studien kommer att intervjuas under ca en 30-40 min och dokumentation av intervjuerna kommer att ske med ljudupptagning för att senare underlätta analysen.

Ljudinspelningen kommer att göras med hjälp av en diktafon som används vid Malmö Universitet med hänsyn till den nya GDPR-lagen, dvs. inga privata mobiltelefoner kommer att användas. Inga personuppgifter kommer att samlas in, utan det blir helt anonymt. Endast ljudupptagningen av lärarna kommer att samlas in och laddas upp på M-servern på Malmö Universitet för att förhindra spridningen och hur länge de har arbetat kommer att noteras. Allt insamlat material kommer att lagras på Malmö universitets server under arbetet med examensarbetet och samtyckesblanketterna kommer att förvaras oåtkomligt på Malmö universitet. Endast intervjuaren, handledaren och examinatorn kommer att ha tillgång till det insamlade materialet.

Studien kommer att genomföras i enlighet med Vetenskapsrådets forskningsetiska principer (2002) vilket innebär att jag innan intervjuerna informerar deltagarna i intervjun om undersökningens syfte och berättar hur användningen av ljudinspelningarna kommer att ske. De intervjuade informeras även om att deras deltagande kommer att vara helt frivilligt och anonymt i studien och att de när som helst kan avbryta sin medverkan i studien utan vidare motivering och deras kunskap som de bidragit med kommer inte att användas. Ljudinspelningarna kommer enbart att användas för den aktuella studien och sedan raderas efter att examensarbetet är godkänt.

……**Nour-Mariam Merhi**
Studentens underskrift och namnförtydligande

Kontaktuppgifter till student (tfn nr, e-mail):

……

Ansvarig handledare på Malmö universitet:

Peter Bengtsson

Kursansvarig på Malmö universitet:

Agneta Rehn

Kontaktuppgifter Malmö universitet:
www.mau.se
040-665 70 00

Information om Malmö universitets behandling av personuppgifter

Personuppgiftsansvarig	Malmö universitet
Dataskyddsombud	dataskyddsombud@mau.se
Typ av personuppgifter	Namn, anteckning av lärandesituation, bild och/eller filmklipp samt ditt samtycke till att Malmö universitet behandlar dessa personuppgifter.
Ändamål med behandlingen	För att möjliggöra undervisnings- och examinationssituationer i skolmiljö för studenter vid Malmö universitets lärarutbildning.
Rättslig grund för behandling	Ditt samtycke.
Mottagare	Personuppgifterna kommer endast användas i utbildningssyfte inom ramen för lärarutbildningen vid Malmö universitet och kommer inte att spridas vidare till någon annan mottagare.
Lagringstid	Malmö universitet kommer spara dina personuppgifter så länge de behövs för ovan angivet ändamål eller till dess att du återkallar ditt samtycke. Efter genomförd kurs/program kommer personuppgifterna att raderas. Malmö universitet kan dock i vissa fall bli skyldiga att arkivera och spara personuppgifter enligt Arkivlagen och Riksarkivets föreskrifter.
Dina rättigheter	Du har rätt att kontakta Malmö universitet för att 1) få information om vilka uppgifter Malmö universitet har om dig och 2) begära rättelse av dina uppgifter. Vidare, och under de förutsättningar som närmare anges i dataskyddslagstiftningen, har du rätt att 3) begära radering av dina uppgifter, 4) begära en överföring av dina uppgifter (dataportabilitet), eller 5) begära att Malmö universitet begränsar behandlingen av dina uppgifter. När Malmö universitet behandlar personuppgifter med stöd av ditt samtycke, har du rätt att när som helst återkalla ditt samtycke genom skriftligt meddelande till Malmö universitet. Du har rätt att inge klagomål om Malmö universitets behandling av dina personuppgifter genom att kontakta Datainspektionen, Box 8114, 104 20 Stockholm.

Samtycke

Härmed samtycker jag till att medverka i ovan beskrivna studentprojekt, samt bekräftar att jag har tagit del av informationen om Malmö universitets behandling av personuppgifter, och Vetenskapsrådets forskningsetiska principer, som säger att

- medverkan baseras på samtycke och detta samtycke kan när som helst återkallas. Alla som tillfrågas har alltså rätt att tacka nej till att delta, eller (om de först tackar ja) rätt att avbryta sin medverkan när som helst, utan några negativa konsekvenser.
- deltagarna kommer att avidentifieras i det färdiga arbetet.
- materialet kommer enbart att användas för aktuell studie och kommer att förstöras när denna är examinerad. [1]

Namn: ……

Namnförtydligande: ……

Dagens datum: ……

[1] De forskningsetiska principerna kan du läsa mer om i Vetenskapsrådets skrift *Forskningsetiska principer inom humanistisk-samhällsvetenskaplig forskning* (2002), som du kan finna här: http://www.codex.vr.se/texts/HSFR.pdf

Bilaga 2

Intervjuguide till en semistrukturerad intervju

Under intervjun skulle jag vilja få ta del av din tankar kring följande ämnen:

- *Hur länge har du arbetat som lärare?*
- *Har du genomfört en utbildning/kurs i programmering/programmeringsdidaktik för matematikundervisning?*
- *Vet du vad det innebär att man ska använda sig av programmering som en strategi för problemlösning?*
- *Hur tror du att programmering ska användas för att stärka/utveckla elevers problemlösningsförmåga?*
- *I vilka matematikkurser och i vilka matematiska områden har du använt programmering?*
- *Hur har du börjat lära ut/arbeta med programmering som en strategi för problemlösning i din undervisning? (Undervisningsmetoder, tidsåtgång, programmeringsverktyg/språk)*
- *Arbetar du projektinriktad eller instruktions inriktat? Varför?*
- *Hittar du eller gör du egna exempel/aktiviteter som du använder dig av i undervisningen? Om läraren ger exempel på aktivitet/uppgift i programmering, ställa vidare frågor som: Hur utvecklar denna elevers problemlösningsförmåga?*
- *Använder du dig av Skolverkets moduler eller läroböcker?*
- *Hur säkerställer du att eleverna har lärt sig det matematiska innehållet och inte bara programmeringen?*
- *Vilka undervisningsstrategier använder du för att utveckla elevernas problemlösningsförmåga med hjälp av programmering?*
- *Hur ser du en progression i elevernas problemlösningsförmåga? Hur dokumenterar du denna?*
- *Hur hanterar du elever som ännu inte lärt sig programmera?*
- *Brukar de ha svårigheter med något specifikt? Hur hanterar du detta?*
- *Har du några ”bra” tips till en nyexaminerad lärare på hur man kan arbeta med programmering som en strategi för problemlösning?*
- *Tar du stöd i forskning/moduler och hur gör du detta?*
- *Hur ger du stöd till svaga elever och stimulans till starka elever?*
- *Vilka möjligheter ser du med programmeringen? Vilka utmaningar finner du?*
- *Vilka tankar har du kring de olika programmeringsmiljöerna (t.ex. blockprogrammering, textbaseradprogrammering) som finns tillgängliga?*
- *Vilka sorts uppgifter tror du är lämpliga att jobba med i programmering?*
- *I vilka av de olika matematiska ämnesområdena tänker du att programmering i matematik skulle kunna användas, förutom de som du redan testat? Varför?*
- *Finns det något område som du inte skulle jobba med programmering i? Varför?*