GÖTEBORGS UNIVERSITET

Varför är programmering så svårt?

En kvalitativ studie om matematiklärares utmaningar i sin undervisning i programmering.

Julien Eskola
Ämneslärarprogrammet med inriktning mot gymnasiet inom matematik och biologi.

Examensarbete: 15hp
Kurs: LGMA2A
Nivå: Avancerad nivå
Termin/år: VT-2021
Handledare: Daniel Persson
Examinator: Johanna Pejlare

Nyckelord: Programmering. Matematiklärare. Datalogiskt tänkande. Skolverket.

Sammanfattning

Studien grundade sig i tidigare forskningsresultat som konstaterade att många svenska matematiklärare inte kände sig tillräckligt förberedda för att undervisa i programmering. Syftet med studien var lokalisera vilka delar som saknades för att matematiklärarna skulle känna sig tillräckligt förberedda för att undervisa i det. Efter avslutad studie var målet att kunna bygga vidare på resultatet och långsiktigt hjälpa lärare med sin undervisning. Studien fokuserade på att ta reda på vad lärare kände att de saknade och även vad som skiljde lärare som kände sig förberedda och de som inte gjorde det. För en tydligare analys av lärarnas skillnader i förståelse och uppfattning om olika fenomen utgick analysen utifrån begreppet datalogiskt tänkande. Studien som genomförts var en kvalitativ studie där målet var att få en förståelse för hur lärarna själva upplevde sin situation. För att göra detta undersöktes lärarna utifrån en kombination av två teoretiska ramverk, nämligen ”de tre dimensionerna av datalogiskt tänkande” och ”PGK-hierarkin”. Med utgångspunkt i dessa formades en intervjuguide för att besvara samtliga frågeställningar. Efter datainsamlingen genomfördes en analys, vilket gjordes enligt strukturen för en tematisk analys. Resultatet för studien visade att det fanns flera områden som lärarna behövde utveckla för att kunna känna sig tillräckligt förberedda för att undervisa i programmering. Exempel på dessa var förståelsen för ämnesplanen, didaktiska kompetenser och förståelse för programmeringens användning i matematiken. Studien visade även att det fanns skillnader mellan hur lärarna såg på programmering, både inom matematiken och även programmering i allmänhet, där flera lärare saknade förståelse för programmeringens möjligheter. Studien visar även indikationer på att många matematiklärare saknar förståelse för det datalogiska tänkandet, vilket också var en tydlig skillnad mellan lärarna som ansåg sig vara tillräckligt förberedda och de som inte kände sig det.

Förord

I mitt förord vill jag passa på att tacka flera personer som hjälpt mig under arbetets gång och som gjort det möjligt för mig att göra denna studie. Först och främst vill jag tacka samtliga som deltog i intervjuerna och visade engagemang och intresse för mitt arbete. Jag vill även tacka min handledare Daniel som har varit ett bra stöd under arbetet där han kommit med tips samtidigt som han uppmuntrat mig att använda den egna kreativiteten. Jag passar även på att tacka min examinator Johanna som kommit med bra tankar och feedback inför slutliga inlämningen. Slutligen vill jag även tacka mina fina vänner, Frida och Liv, för att de funnits som stöd under arbetet och gång på gång fått mig att förstå att det inte bara är jag som inte förstår och att det löser sig till slut.

Julien

Innehållsförteckning

1 Inledning

Som lärarstudent inom matematik har lärarstudenter under sin utbildning läst en mängd olika kurser inom olika matematiska områden för att uppnå en viss matematisk kompetens. Utöver dessa kurser ingår även kurser inom didaktik där lärarstudenter får ta del av olika teorier och strategier för att uppnå lärande. Dessa kunskaper får vi sedan möjlighet att testa och utveckla under de fyra praktikperioder som ingår i utbildningen. En del av matematiken som lärarstudenter oftast inte får träna speciellt mycket på, varken kunskapsmässigt eller didaktiskt, är användandet av programmering. Utifrån detta har det väckts känslor om programmering, dels nyfikenhet och dels rädsla hos mig och även flera av lärarstudenterna i min årgång, där vi under olika tillfällen diskuterat hur handledarna hanterat detta när vi varit ute på praktik. Oftast mynnar dessa diskussioner ut i att de flesta av oss inte fått tagit del av programmering under våra praktikperioder, vilket därmed får mig att fundera på "varför har jag knappt fått se inslag av programmering under mina praktikperioder?".

I den svenska gymnasieskolans ämnesplan för matematik kan man sedan juli 2018 läsa hur programmering ska vara en del av undervisningen. Under åren har ämnesplanen i matematik varit väldigt omdiskuterad och från och med juli 2021 kommer en ny revidering träda i kraft (Skolverket, 2021a). Men införandet av programmering har inte varit enkelt för vare sig lärare eller elever. När revideringen första gången presenterades 2016 blev både lärare och lärarstudenter nervösa inför utmaningen. Många lärare hade mycket negativt att säga om förändringen och det har minst sagt varit en turbulent period. Ungefär samtidigt som själva införandet gjordes skrevs också första examensarbete där vi lyfte olika svårigheter som tillkom med implementering av programmering. När det arbetet skrevs var implementeringen av programmering i matematiken relativt nystartad och ingen visste hur det skulle utvecklas. Med utgångspunkt i mitt föregående arbete ska jag i denna studie fördjupa mig i hur det ser ut utifrån lärarnas perspektiv ungefär tre år efter implementeringen. Tidigare forskningsresultat (Misfeldt, Szabo & Helenius, 2019) har visat att matematiklärare på gymnasiet oftast inte känner sig tillräckligt förberedda för att undervisa i programmering. Detta resultat tillsammans med min fundering om varför jag knappt fått se inslag av programmering under mina praktikperioder har skapat ett intresse för mig i att ta reda på varför det är på detta vis.

1.1 Syfte

Studien grundar sig i tidigare forskningsresultat om att matematiklärare på gymnasiet oftast inte känner sig tillräckligt förberedda för att undervisa i programmering. Utifrån detta resultat blir syftet således att försöka ta reda på vad lärarna känner att de saknar för att de ska känna att de är tillräckligt förberedda. Målet med studien blir därmed att kunna lokalisera specifika områden som aktiva lärare anser som problematiska för att vi i framtiden ska kunna hjälpa lärare i deras undervisning. För att göra detta kommer jag att analysera datan utifrån teorier för datalogiskt tänkande. Begreppet datalogiskt tänkande kommer från engelskans "computational thinking" och innefattar problemlösningsfärdigheter för att förstå, analysera och lösa problem med hjälp av en dator. Studien utgår från en huvudfrågeställning med grund i tidigare forskning, som sedan följs av två fördjupande frågor. Frågeställningarna för studien är följande:

- På vilka grunder anser matematiklärare att de inte är tillräckligt förberedda för att undervisa i programmering inom matematiken?
- Vad skiljer lärare som känner sig förberedda och lärare som inte känner sig förberedda?
- Hur skiljer sig det datalogiska tänkandet mellan lärare som känner sig förberedda och de som inte känner sig förberedda?

1.2 Strukturell översikt

Rapporten för denna studie börjar med en bakgrund som innefattar lite om programmering och dess utveckling inom gymnasieskolan, uppkomsten av begreppet datalogiskt tänkande och sedan en presentation av det teoretiska ramverket. I det avsnittet beskrivs både tidigare forskning och ramverken som använts för att analysera insamlade datan. Därefter följer ett metodavsnitt som innefattar en omfattande metodbeskrivning. I den ingår delar om metod för datainsamling, urval, analysmetod, forskningsetiska aspekter och även trovärdigheten för studien. Efter metodavsnittet presenteras resultatet, som dels innehåller presentation av respondenterna, dels de kategorier som använts för att besvara frågeställningarna. Rapporten avslutas med en diskussion. Den innefattar en resultatdiskussion, metoddiskussion, reflektion kring de didaktiska konsekvenserna och slutligen om möjliga forskningsområden. I slutet hittas även referenser och bilagor som använts i studien.

2 Bakgrund

I detta avsnitt ges en möjlighet för läsaren att skapa sig en förståelse för ämnet för att på ett enklare sätt förstå resten av studiens upplägg. Bakgrunden introduceras med ett avsnitt om programmering, som främst fokuserar på utvecklingen av programmeringen inom gymnasieskolan. Läsaren får både se tillbaka ur ett historiskt perspektiv och se hur programmering används idag. Därefter kommer ett avsnitt som gör en djupdykning i det breda begreppet "datalogiskt tänkande". Där presenteras dels uppkomsten av begreppet, dels hur det formats med tiden. Slutligen presenteras tidigare forskning och även de teoretiska ramverken använts för analysen.

2.1 Programmering

Begreppet programmering kan användas inom flera olika områden och har en ganska omfattande innebörd. En kort och enkel förklaring av vad begreppet programmering innebär är att skapa en instruktion för hur något teknologiskt, exempelvis en dator, ska utföra en viss aktivitet (Butterfield, Ngondi & Kerr, 2016). Programmering innefattar analys av krav, skapande av algoritm, koda ett program och sedan testa om den utför det uttänkta målet ("Programmering", 2020, december 30)

2.1.1 Programmering i den svenska gymnasieskola

Även fast vi pratar om programmering som något nytt och modernt är det absolut inget nytt i skolvärlden. Redan år 1961 pratade Alan Perlis om att den vanliga medborgaren bör ta del av programmering och förstå processen och hur den används. Han antydde mer specifikt att programmering kan hjälpa människor att förstå det teoretiska bakom olika beräkningar, där denna förståelse kan nyttjas för att bättre förstå sig på exempelvis matematik (Mannila, 2017). Under samma period som Perlis uttalar sig om detta har programmering på lite olika sätt börjat synas i skolor runt om i världen, bland annat i Sverige (Mannila, 2017; Rolandsson & Skogh, 2014). Programmeringens syfte i skolvärlden har varierat mycket under åren vilket har gjort att det inte alltid varit ett givet ämne i skolan. Under de senaste 50 åren har ämnesplanerna för gymnasiet gjorts om fyra gånger, där mindre revideringar av ämnesplaner inte räknats in. I dessa ämnesplaner skiljer det sig på hur skolan sett på digitaliseringen, vilket gjort att man korrigerat mängden programmering utifrån syfte. Detta avsnitt delas in i två delar. Först ska programmering presenteras utifrån ett historiskt perspektiv för att sedan titta på hur de nuvarande ämnesplanerna ser ut.

2.1.1.1 Programmering ur ett historiskt perspektiv

Under 60-talet var gymnasieskolan uppdelad i två olika delar, en akademisk- och en yrkesmässig utbildning. Under 70-talet slogs dessa två delar ihop till en gemensam skola och införde även en ny läroplan, "Lgy70". Programmering hade på olika sätt erbjudits redan sedan slutet av 60-talet, men det fanns inget övergripande fokus på att lära sig hantera datorer eller att programmera (Rolandsson & Skogh, 2014). Under 80-talet ökade diskussionen kring programmering och mer och mer forskning gjordes på området. Detta ledde till att fler och fler länder började inkludera programmering i läroplanerna, bland annat Sverige (Mannila, 2017). Politiker insåg även under tidigt 80-tal att Sverige skulle behöva duktiga problemlösare i framtiden för att Sverige skulle kunna fortsätta sin utveckling som ledande industriell nation (Rolandsson & Skogh, 2014). Som följd av detta började Skolöverstyrelsen omformulera läroplanerna där delar som datalära skulle få en större plats, främst inom natur och teknik spåren. I en uppdaterad version av Lgy70 (Skolöverstyrelsen, 1981) som var riktad mot de som

läste matematik på naturvetenskaplig eller teknisk linje kunde man läsa om programmering som en del av matematiken. Programmering var en del av "dataläran" som i sig var ett moment i matematiken, likt geometri eller algebra. Dataläran beskrevs som ett sätt för eleverna att utveckla förståelsen om databehandling och numeriska metoder där olika koncept som iteration, stegning och simulering lyftes. Skolöverstyrelsen tydliggjorde även att syftet dels var att eleverna skulle förstå att det är människan som styr datorer genom att ge instruktioner och dels att de tänkte att eleverna genom programmering stimulerade kreativiteten och självständiga tänkandet. Samtidigt som de antydde att det skulle stimulera eleverna stod det även tydligt att eleverna inte skulle utbildas i själva programmeringsspråket och dess detaljer (Skolöverstyrelsen, 1981). Denna typ av inkluderingar av programmering i matematiken har även funnits på flera andra program men inte i lika stor utsträckning. I slutet av 60-talet utvecklades programmeringsspråket BASIC som var avsedd för nybörjare (Mannila, 2017). När programmering började ta större plats i läroplanen beslutade Skolöverstyrelsen även att BASIC skulle vara det officiella språket som skulle användas på alla skolor vid användande av programmering (Skolöverstyrelsen, 1981).

Som följd av att Skolöverstyrelsen införde programmering och andra, vad som ansågs vara avancerade, moment på natur- och teknikprogrammen minskade även antalet ansökningar till dessa program. Anledningen var att många uppfattade dessa linjer som för teoretiska och svåra (Rolandsson & Skogh, 2014). Hebenstreit (citerad i Rolandsson & Skogh, 2014) skrev i slutet av 80-talet "to teach programming to everybody is like teaching everybody to pilot a private plane as a preparation for travelling by plane". Problematiken och diskussionerna följde med fram till början av 90-talet där Skolöverstyrelsen började överväga programmeringens roll i skolan, vilket ledde till att undervisningen med programmering minskade till följd av detta. Dessutom var det många lärare som inte tagit tag i att lära sig programmering, vilket resulterade i att de heller inte undervisade i programmering i den utsträckning det var tänkt (Kafai & Burke, refererad i Mannila, 2017). I takt med att utvecklingen av datorer förbättrades blev datorerna rustade med diverse olika program som gjorde datorerna alltmer användarvänliga. Det gjorde att intresset och argumenten för att programmera minskade ytterligare. Under mitten av 90-talet gjordes läroplanen om och fick det nya namnet "Lpf 94" (Rolandsson & Skogh, 2014). Fokus skiftade från att lära sig om datorer till att lära sig med datorer (Mannila, 2017). I Lpf 94 nämndes inte begreppet programmering någonstans, varken i delen om mål och riktlinjer, i någon av de nationella programmen eller i beskrivningen för ämnet matematik (Skolverket, 1994). Däremot beskrevs det betydligt mer övergripande om användning av datorer, oftast som ett verktyg för studier. Inom ämnet matematik skrevs exempelvis "Tillgången till nya tekniska hjälpmedel förändrar delvis matematikens innehåll och metoder. Många rutinoperationer, främst av numerisk och grafisk karaktär, kan nu utföras av miniräknare och datorer" (Skolverket, 1994, s.40).

Mot slutet av 90-talet och början på 2000-talet började teknikens utveckling verkligen ta fart, där fler och fler började använda "smarta telefoner" och dess appar. Denna utveckling gjorde att många återgick till tankesättet om att allmänheten behövde förstå hur all teknik i form av datorer, telefoner och så vidare fungerade (Mannila, 2017). Skolverket följde denna utveckling och valde att återigen omformulera läroplanen. Denna nya läroplan skulle få namnet "GY2000" (Rolandsson & Skogh, 2014), där programmering åter skulle få ta plats. Här gjordes flera insatser för att inkludera programmering, där man på naturvetenskapsprogrammet kunde läsa inriktningen "matematik och datavetenskap" som skulle innehålla flera kurser i programmering. Däremot fick inte programmeringen någon plats i matematiken, utan blev som ett eget ämne inom datavetenskap. Enda gången som begreppet programmering användes var under beskrivningen för kursen "matematik - diskret" där ett av målen var att elever efter avslutad

kurs skulle känna till grundläggande programmering. I GY2000 valde Skolverket istället att använda sig av "tekniska hjälpmedel" där de egentligen endast skriver att det ändrat förutsättningar för matematiken. Det stod väldigt lite om själva användning av tekniska hjälpmedel, där egentligen den enda delen som det nämndes var under rubriken "ämnets karaktär och uppbyggnad" (Skolverket, 2000). Där stod det "...numeriska, grafiska och algebraiska metoder utnyttjas och nya typer av problem av mer sammansatt karaktär kan studeras i ämnet. De tekniska hjälpmedlen har dock begränsat värde utan kunskaper om begrepp och metoder" (Skolverket, 2000, s.74–75). Även denna läroplan blev snabbt omdiskuterad och politiker ville fortsätta satsa på den digitala utvecklingen. 2011 släpper Skolverket ytterligare en ny läroplan, denna gång lite mer influerad av den digitala tekniken. Begreppet programmering nämndes fortfarande inte utan istället skrev Skolverket om "arbete med digitala hjälpmedel och med stöd av datorer" (Skolverket, 2011).

I samband med att Skolverket släppte den nya läroplanen 2011 började det på flera håll i världen komma uppdateringar av läroplaner med större fokus mot de tekniska hjälpmedel, där skolors fokus framförallt var på utveckling av förståelse och användningen av programmering (Mannila, 2017). 2016 beslutade Skolverket att de behövde göra omformuleringar i styrdokumenten för att satsa på utveckling av digitala kompetenser. Motiveringen till detta var att de vill stärka elevers förmåga att kunna använda, förstå och kritiskt förhålla sig till sin digitala omvärld. Skolverket skrev även tydligt att de vill arbeta för att utbildningen ska innehålla inslag av just programmering, speciellt inom matematik och teknik (Skolverket, 2016). Den reviderade ämnesplanen (Skolverket, 2018) tydliggjorde vikten av att använda digitala verktyg över lag i matematiken men specificerade endast programmering i vissa kurser. Det var främst kurser som ingick på natur- och teknikprogrammet som innefattade programmering, men även kursen matematik 3b som var valbar när man läste på andra program. Användning av programmering nämndes under rubriken "problemlösning" och har en övergripande beskrivning. Skolverket (2018) skrev följande "Strategier för matematisk problemlösning inklusive modellering av olika situationer, såväl med som utan digitala verktyg och programmering". I en rapport inför implementeringen av programmering i matematiken skrev Skolverket även:

> I vissa sammanhang kan programmering vara synonymt med "skrivande av kod" medan det i andra sammanhang avses ett vidare perspektiv på programmering där även problemformulering, val av lösning, att pröva och ompröva samt att dokumentera räknas in i programmeringsaktiviteten. För detta krävs förmåga till kreativ problemlösning, logiskt tänkande, ett strukturerat arbetssätt och förmåga att generalisera. Det senare perspektivet på programmering har varit utgångspunkt för förslagen till förändringar i styrdokumenten. (Skolverket, 2016, s. 7)

Det som Skolverket refererade till som "det senare perspektivet" är väldigt lik det som innefattas i de olika definitioner som finns för datalogiskt tänkande (begreppet datalogiskt tänkande kommer behandlas i större utsträckning i avsnitt 2.2). Skolverket skrev även i rapporten att de diskuterat begreppet och att det haft ett inflytande i hur de formulerat syftet och centrala innehållet, men Skolverket valde att inte nämna begreppet utan istället använda sig av begreppet programmering för att synliggöra detta (Skolverket, 2016).

2.1.1.2 Programmering i matematiken idag

Den reviderade ämnesplanen som presenterades 2018 är den som används än idag. Däremot har Skolverket gjort ytterligare en revidering som kommer träda i kraft juli 2021. I den nya revideringen har Skolverket omformulerat flera delar av ämnesplanen, där många av

ändringarna är intressanta för denna studie. Precis som tidigare nämner Skolverket i syftet, centrala innehållet och i kunskapskraven om hur digitala verktyg, på lite olika sätt, ska ta plats i matematikundervisningen (Skolverket, 2021a). Digitala verktyg nämns ofta och inkluderas i många olika moment och precis som tidigare nämns inte programmering i vare sig syfte eller i kunskapskraven. Programmering nämns bara i de centrala innehållet för vissa utvalda kurser, med fokus för de matematikkurser som ingår i natur- och teknikprogrammet, som även kallas "c-spåret". Programmering finns även med i det centrala innehållet för matematik 3b som är en fortsättningskurs för flera andra program än natur och teknik. I de lägre kurserna (matematik 1c och 2c) skrivs "Exempel på hur programmering kan användas som verktyg vid problemlösning, databearbetning eller tillämpning av numeriska metoder." (Skolverket, 2021a). När Skolverket skriver "exempel på" är syftet att läraren ska lägga fokus på den praktiska användningen, där elever ska få möta det matematiska innehållet och se hur det kan användas och inte nödvändigtvis använda det (Skolverket, 2021b). I de högre kurserna (3b, 3c, 4, 5) står det istället "Användning av programmering som verktyg vid problemlösning, databearbetning eller tillämpning av numeriska metoder." (Skolverket, 2021a). Den stora skillnaden mellan dessa två formuleringar är att i de lägre kurserna ska jobba med "exempel", medan de senare kurserna syftar till "användningen", alltså att eleverna ska använda sig av programmering (Skolverket, 2021b).

Skolverket menar att största anledningen till att de infört programmering i matematiken är att de skapar möjligheter för att göra större och fler beräkningar. Med hjälp av programmering kan eleverna utforska matematiken och utveckla kunskaper som inte hade varit möjliga att göra för hand eller med ett färdigt digitalt redskap (Skolverket, 2021b). Skolverket har valt att inte skriva ut något om vilket programmeringsspråk eller vilka programmeringsmiljöer som lärare ska arbeta i (Skolverket, 2021a). Däremot skriver de "även kalkylblad kan användas för iterativa och villkorsstyrda beräkningar, vilket är det som ofta används för att kännetecknа programmering" (Skolverket, 2021b, s. 25). I kommentarmaterialet kan man läsa om lite olika förslag på hur de tre olika områden (problemlösning, databearbetning och tillämpning av numeriska metoder) kan användas. Läraren har sedan friheten att välja hur och vilken del som eleverna ska fokusera på utifrån vad läraren anser passa bäst till kursens innehåll (Skolverket, 2021b).

Eftersom det inte står något om programmering i kunskapskraven kan man fundera på hur mycket lärarna kan förvänta sig av eleverna. För studiens syfte skickades det ett mejl (se Bilaga 2) till Skolverket för att besvara några frågor. I mejlet beskriver de bland annat att programmering inom matematik kan kräva avancerad kodning, vilket gör att fokus flyttas från matematiken till programmeringen. De föreslår istället att eleverna kan använda sig av färdiga koder som de ska försöka förstå och variera vissa delar för att få fram vissa värden. Däremot ska det, speciellt inte i de högre kurserna, endast bestå av färdiga koder. De skriver bland annat "Det är svårt att dra en gräns för hur långt in i dessa tankesätt och metoder man bör eller behöver gå. Att endast ge elever ett färdigt program för att (exempelvis) beräkna värden på integraler uppfyller knappast avsikten med programmering i det centrala innehållet". De skriver sedan att låta elever skriva och bearbeta egna koder ligger väldigt nära avsikten med programmering i matematiken, samtidigt som de också skriver att det oftast är för tidskrävande och att lärare då behöver prioritera bort detta (Skolverkets upplysningstjänst).

2.2 Datalogiskt tänkande

Som tidigare nämnt har vi haft programmering inom ämnet matematik sedan 2018 när Skolverket gjorde en revidering av ämnesplanerna. Däremot är inte detta något nytt. Precis som

med diskussionerna kring användandet av datorer i skolvärlden har forskare även länge diskuterat programmeringens roll i matematiken. En av de första personerna som var med och utvecklade tankar kring att kombinera programmering med matematiken för utveckling av matematiska förmågor var Seymour Papert (Mannila, 2017). Papert (1980) skriver i sin bok om hur han tillsammans med sin forskargrupp utvecklat programmet LOGO. LOGO är ett programmeringsspråk utvecklat för att även den inte så erfarna ska kunna använda sig av programmering för att kommunicera med en dator. Papert var väldigt inspirerad av Piagets konstruktivistiska synsätt för lärande, vilket innebär att vi lär oss genom att på ett spontant och kreativt sätt får arbeta i en naturlig miljö. Detta lärandesätt blir tydligt i LOGO som är utvecklat utifrån hur barn kommunicerar och lär sig utifrån nyfikenhet. Han skriver

> The goal of education research tends therefore to be focused on how to improve classroom teaching. But if, as I have stressed here, the model of successful learning is the way a child learns to talk, a process that takes place without deliberate and organized teaching, the goal set is very different. I see the classroom as an artificial and inefficient learning environment that society has been forced to invent because its informal environments fail in certain essential learning domains, such as writing or grammar or school math. (Papert, 1980, s.8)

Papert var inte bara med och skapade LOGO utan var också först med att uttala sig om ett begrepp som än idag är ytterst aktuellt, och inte minst inom denna studie, nämligen *computational thinking*. I den svenska översättningen av hans bok används begreppet *datametodologiskt* tänkande, vilket sällan verkar användas inom forskning. Idag används istället den förkortade varianten *datalogiskt tänkande* (Mannila, 2017, Gerestrand, 2017; Westerberg, 2018). Genomgående i detta arbete kommer jag därför använda mig av begreppet datalogiskt tänkande. Papert förklarar egentligen aldrig vad begreppet innebär utan använder det som en relation mellan att programmera och processen att tänka (Nouri m.fl., 2020). Det tar sedan ungefär 25 år innan begreppet blir aktuellt igen, denna gång av forskaren Jeannette Wing. 2006 skriver hon en artikel där hon uttrycker att datalogiskt tänkande inte endast ska tillämpas av datavetare utan något som bör tillämpas av gemene man (Mannila m.fl., 2014). Till skillnad från Papert beskrev Wing olika delar som involveras när människor arbetar med datavetenskap och programmering. Wings artikel var startskotten för många forskares arbeten med att försöka definiera datalogiskt tänkande (Nouri m.fl., 2020).

Det som gjorde att Wings artikel blev så pass uppmärksammad var att hon antydde att datalogiskt tänkande är en attityd och en färdighet som är fundamental för varje människa. (Mannila m. fl., 2014). Hon menade att precis som barn lär sig läsa, skriva och räkna aritmetik så bör vi även lägga till datalogiskt tänkande som en del av den analytiska förmågan. (Wing, 2006). Detta menade hon skulle bli viktigt i och med utvecklingen av tekniken för att förbereda unga inför ett arbetsliv som mer och mer kommer influeras av datalogiskt tänkande (Sands, Yadav & Good, 2018). Wing tydliggör även att datalogiskt tänkande inte ska ses som en likhet med programmering, utan snarare att kompetensen som används när man programmerar är användbar för att kunna lösa problem. Hon förklarar datalogiskt tänkande som ett sätt att lösa problem, designa system och förstå människans behov. Hon definierar datalogiska färdigheter som ett mer övergripande begrepp som innefattas av flera olika färdigheter, nämligen abstraktion, nedbrytning av problem, igenkänning av mönster, modellering, algoritmiskt tänkande och logiskt tänkande (Wing, 2006). Några år senare kom hon med en uppdatering av sin syn på datalogiskt tänkande, som tillsammans med den första kan sammanfattas med att hon ansåg att alla kan dra nytta av att lära sig principerna, metoderna och de olika koncepten som används inom datavetenskap (Nouri m.fl., 2020).

Efter Wing är det många forskare som försökt operationalisera datalogiskt tänkande (Nouri m.fl., 2020). Denning (2009) anser också att datalogiskt tänkande är viktigt och går så långt att han tror datalogiskt tänkande kommer vara oundvikligt för framtida forskning inom alla kategorier. Han refererar till det som "det tredje benet", där de två andra benen, teori och experiment, är grunden för nuvarande forskning. Denning (2009) tillsammans med sina kollegor arbetade fram ett ramverk för "computational science" som de kallade "The great principles framework". Den innefattar två delar, dels olika principer som innefattar exempelvis kommunikation, koordination och design och dels olika praxis som innefattar programmering, modellering och tillämpning. Denning (2009) förklarar datalogiskt tänkande som en integrerad tankegång för de olika delarna som innefattas inom praxis, som en förmåga att tolka världen som algoritmiska omvandlingar från indata till utdata. Denning och Wing skiljer sig i hur de förklarar datalogiskt tänkande, men båda argumenterar för att det är viktigt att studenter lär sig om detta (Sands, Yadav & Good, 2018). Selby (2015) nyttjade Wings definition och försökte koppla samman de med *Blooms taxonomi*, vilket är en är en modell för hur man kan se på lärande. Modellen består av sex olika nivåer som innefattar en viss typ av förståelse och kunskaper, som sedan ordnas i en viss hierarkisk ordning i formen av en pyramid. Basen består av de mest grundläggande kunskaperna, som sedan byggs på uppåt med mer avancerad förståelse ("Blooms taxonomi", 2020, maj 5). Med hjälp av Wings syn på datalogiskt tänkande kategoriserade Selby det som: utvärdera, design av algoritm, generalisering, abstraktion av funktionalitet, abstraktion av data och nedbrytning (Selby, 2015). Dessa olika sätt att försöka operationalisera datalogiskt tänkande fortsätter, där flera olika forskare försöker precisera innebörden av datalogiskt tänkande (Aho, 2012; Barr & Stephanson, 2011; Grover & Pea, 2013; Lee m.fl., 2011). De olika definitionerna framhäver olika förmågor och koncept som datalogiskt tänkande innefattar, där de flesta består av abstraktion, algoritmer, data, nedbrytning av problem, parallellism, felsökning, prövning och kontroll (Nouri m.fl., 2020).

Utöver alla försök att precisera vad datalogiskt tänkande innefattar finns det även en mängd olika övergripande förklaringar, som på ett förenklat sätt beskriver vad datalogiskt tänkande är. Exempelvis skriver Khine (2018) en kortfattad beskrivning av datalogiskt tänkande som en tankeprocess som behövs för att strukturera och formulera problem sådana att det på ett bra och effektivt sätt kan hanteras av någon typ av beräknande tekniskt hjälpmedel. The Royal Society definierar istället datalogiskt tänkande som "the process of recognising aspects of computation in the world that surrounds us, and applying tools and techniques from Computer Science to understand and reason about both natural and artificial systems and processes" (2012, s.29).

2.3 Tidigare forskning

Forskare har sedan länge studerat olika lärandemoment gällande programmering, allt från hur läraren ska undervisa till vad eleverna faktiskt lär sig under lektionerna. I och med implementeringen av programmering inom matematiken på gymnasiet öppnades många nya forskningsområden. Det finns sedan tidigare en del forskning kring undervisning med programmering, men eftersom programmering tidigare nästa enbart använts på universitet gör det att det finns väldigt lite forskning kring hur lärare ska använda det på grund- och gymnasieskolan (Heintz, Mannila & Färnqvist, 2016). Dessutom har tidigare forskning främst fokuserat på elevers missuppfattningar och svårigheter, mer sällan har forskare studerat utifrån lärares perspektiv (Nouri m.fl., 2020). Oftast fokuserar dessa studier på svårigheterna med lärandet av programmeringen istället för hur elever ska lära sig matematiska fenomen genom programmering (Kilhamn & Bråting, 2019). Många av de studier som gjorts på datalogiskt tänkande som har en utgångspunkt hos lärare har oftast fokuserat på lärarstudenter och deras utbildningar (Sands, Yadav & Good, 2018). Exempelvis visade Angeli och Jaipal-Jamani

(2018) att man med relativt små insatser, i form av kortare utbildningar, kan uppnå en relativt bra nivå av förståelse hos lärarstudenter. Givetvis kan liknande studier ge indikationer på hur mycket som krävs för att uppnå en viss förståelse, men det kommer inte kunna identifiera de svårigheter som finns för aktiva matematiklärare i att undervisa matematik med inslag av programmering. Genom att göra studier på aktiva lärare kan både forskare och politiker få en bättre förståelse för de svårigheter och utbildningar som krävs (Sands, Yadav & Good, 2018), vilket också är ett argument för att denna studie genomförs med fokus på just aktiva matematiklärare.

En av de mest uppenbara svårigheterna med att undervisa utifrån ett datalogiskt tänkande är att veta vad som ska ingå i undervisningen. Med tanke på den mängd olika varianter och förklaringar som finns för begreppet hos olika forskare är det inte konstigt att även lärare blir förvirrade när de ska försöka applicera detta på sin undervisning (Sands, Yadav & Good, 2018). Precis som Wing (2006) konstaterade är datalogiskt tänkande inte samma sak som programmering utan innefattar flera aspekter, däremot menar Lye och Koh (2014) att det krävs datalogiskt tänkande för att kunna lära ut programmering. Programmering, som också har en relativt bred innebörd, kan också ses som en aktivitet som utvecklar det datalogiska tänkandet (Kilhamn & Bråting, 2019). Denna invecklade begreppsdefinition speglas också i Skolverkets (2016) beskrivning av programmering, där de nämner att programmering kan ses som mer än att bara koda utan också innefattar problemformulering, val av lösning, prövning osv. Det nämner även "För detta krävs förmåga till kreativ problemlösning, logiskt tänkande, ett strukturerat arbetssätt och förmåga att generalisera" (2016, s.7), vilket är nära kopplat till det som innefattas av flera av de definitioner som tidigare nämnts av datalogiskt tänkande. I studien av Sands m.fl. (2018) rapporterades att många lärare missförstått vad som innefattas i det datalogiska tänkandet, där de menar att många lärare tror att de genom användandet av olika datorprogram, exempelvis Microsoft Office, utvecklar det datalogiska tänkandet. Generellt har lärare väldigt lite eller helt saknar kunskap och förståelse för hur datalogiskt tänkande ska användas och implementeras i undervisningen (Sands, Yadav & Good, 2018). I en studie av Kilhamn m.fl. (2021) undersökte de lärares argument till att ha programmering som en del av matematiken, där de kom fram till två olika inriktningar. Den första är mer allmänt om programmering, nämligen att programmering kan vara ett bra verktyg och att det skapar intresse och engagemang hos eleverna. Den andra inriktningen syftar till att programmering kan hjälpa lärandet i matematiken (Kilhamn, Bråting & Rolandsson, 2021). Den första inriktningen av argument är egentligen ingen motivering till varför programmering just ska vara i matematiken, utan mer att programmering kan vara ett bra hjälpmedel i skolvärlden. Lärarna i Kilhamn m.fl. (2021) studie hade alla erfarenhet av programmering, ändå fanns det flera som ställde sig kritiska till implementeringen. Även Misfeldt m.fl. (2019) konstaterar att det finns matematiklärare som överhuvudtaget inte ser sambandet mellan programmering och matematiken, där några lärare i deras studie menade att det hade varit bättre att implementera programmering i andra ämnen. Ett av argumenten som lärarna använder refererar tillbaka till när datorer introducerades som ett hjälpmedel i skolan. De menar att när elever skulle börja lära sig använda skrivbord, sökfunktioner och liknande på datorer var detta inte något som placerades i matematiken, utan då införde Skolverket ett nytt ämne som behandlade detta (Kilhamn, Bråting & Rolandsson, 2021). Däremot finns det även många argument för att programmering faktiskt ska vara en del av matematiken. En studie visar att matematiklärare generellt anser att det finns ett samband mellan matematik och programmering, där många lärare vill kunna ta detta samband med in på lektionerna. Problemet som många lärare uttrycker är att de känner sig begränsade i kunskapen om hur de faktiskt ska göra det (Misfeldt, Szabo & Helenius, 2019). Många lärare menar att på samma sätt som man tänker i programmeringen, med logiska algoritmer som man arbetar med steg för steg att bryta ner det i mindre delar, tänker

man även i matematiken. Flera lärare menar även att programmering kan vara ett alternativt sätt att lära sig olika matematiska fenomen, dels genom visualisering och djupdykning av ett matematiskt innehåll och dels genom att programmering i sig själv kan klassas som matematik (Kilhamn, Bråting & Rolandsson, 2021).

Ett första steg i att lärare ska använda programmering i matematik är att de överhuvudtaget ska kunna programmera. Problemet börjar redan med att dagens matematiklärare har väldigt begränsade eller inga programmeringskunskaper alls (Bråting, Kilhamn & Rolandsson, 2021; Nouri m.fl., 2020). För att kunna motivera och engagera elever till att nyttja programmering som ett hjälpmedel inom matematiken måste läraren både kunna hantera program i vilket de programmerar och också omsätta det i tydliga matematiska situationer. För att göra detta krävs goda kunskaper i både programmering och matematik (Bråting, Kilhamn & Rolandsson, 2021). Sedan måste lärare även kunna förstå hur de faktiskt ska använda programmeringen tillsammans med matematiken i sin undervisning. Många lärare verkar förstå att de med hjälp av programmering kan träna på logiskt tänkande och problemlösning, vilket är två delar som oftast ingår i definitionen av datalogiskt tänkande (Sands, Yadav & Good, 2018). Däremot noterades det i en studie av Nouri m.fl. (2020) att det fanns flera andra delar av det datalogiska tänkandet som lärare i deras studie inte nämner. Nouri m.fl. menar att en av anledningarna är att lärare saknar tillräcklig utbildning inom programmering och därmed saknar förståelse för grundläggande koncept. Bristen på goda kunskaper inom programmering begränsar sedan även den kreativa förmågan i skapandet av övningar, vilket gör att lärare inte kommer åt de matematiska lärandemålen som är satt för just det momentet. Oftast hamnar fokus på proceduren i användningen av algoritmerna istället för fokus på matematiken och förståelsen av det eleverna jobbat med (Bråting, Kilhamn & Rolandsson, 2021). Dessutom kommer den låga förståelsen för datalogiskt tänkande bland lärare i kombination med de vaga formuleringarna i ämnesplanerna om hur programmering ska användas skapa ännu sämre förutsättningar för lärande (Nouri m.fl., 2020).

I en undersökning om svenska matematiklärares uppfattning om programmering svarade en stor del av lärarna att de inte kände sig tillräckligt förberedda för att undervisa i programmering (Misfeldt, Szabo & Helenius, 2019). Liknande resultatet har även konstaterats av andra forskare som även lyfter att man behöver utbilda lärare i datalogiskt tänkande, både aktiva och blivande (Sands, Yadav & Good, 2018; Nouri m.fl., 2020). Dessutom är en del av de studier som presenterats gjorda på lärare som delvis använder programmering i sin undervisning, där det med största sannolikhet finns en stor skara av lärare som inte har lika bra kunskaper som i studierna som gjort (Kilhamn, Bråting & Rolandsson, 2021).

2.4 Teoretiska ramverk

Hittills har både olika definitioner och förklaringar för datalogiskt tänkande presenterats utifrån olika forskares arbeten och sedan även vad tidigare forskning kommit fram till gällande lärares uppfattningar och utmaningar med införandet av programmering i matematiken. Denna studie kommer att bygga vidare på resultat från tidigare studier och analysera matematiklärares utmaningar utifrån det datalogiska tänkandet. En utmaning i detta är att begreppet datalogiskt tänkande hanteras på lite olika sätt beroende på vem som skriver om det. I grunden finns det en viss samhörighet mellan olika definitioner, samtidigt som det finns väldigt mycket åsikter om vad som inte ska ingå. Problematiken ligger i att begreppet är av sådan karaktär att det med bara två ord, datalogiskt tänkande, ska förklara en väldigt bred och komplex innebörd. För att försöka tydliggöra och underlätta detta utmanande begrepp har forskare försökt skapa modeller

för att på ett enkelt sätt kunna kategorisera och ge en övergripande bild. I avsnitt 2.2 beskrivs hur Wing (2006), Denning (2009) och flera andra forskare försökt definiera datalogiskt tänkande. Denna stora variation i användandet av begreppet menar Weintrop m.fl. (refererad i Nouri m.fl,. 2020) skapar en förvirring och därmed svårigheter när det ska integreras med undervisningen. I samband med att diskussionen kring datalogiskt tänkande växte kom allt fler definitioner fram, vilket gjorde att förvirringen blev allt större. I takt med den ökade förvirringen började forskare fundera på hur de skulle kunna applicera detta på undervisningen. Många forskare började därmed skapa olika verktyg och ramverk för att stötta utvecklingen av datalogiskt tänkande i undervisningen.

Nedan presenteras två ramverk som använts vid utformandet av denna studie. Teorierna grundar sig båda i det datalogiska tänkandet men är av olika karaktär och därmed kommer åt olika aspekter.

I avsnitt 2.3.1 beskrivs bland annat om att matematiklärare inte känner sig tillräckligt förberedda för att undervisa i matematik med användande av programmering. För att programmering ska kunna vara en del av matematiken måste lärarna först och främst kunna hantera ett programmeringsspråk men även förstå dess användning inom matematiken. Lye och Koh (2014) sträcker sig så långt att de menar att det krävs datalogiskt tänkande för att kunna lära ut programmering. För att ta reda på om hur stor förståelse lärarna har för datalogiska tänkandet i relation till programmering kommer analysen utgå från två ramverk. Båda teorierna grundar sig i det datalogiska tänkandet men är av olika karaktär. Det första är mer av ett sätt att se vilka aspekter av datalogiska tänkandet som lärarna reflekterar kring, medan de andra mer fokuserar på förståelsen för det datalogiska tänkandet.

2.4.1 Det tre dimensionerna av datalogiskt tänkande

Det första ramverket som presenteras är den av Brennan och Resnick (2012). Ramverket har aldrig fått något namn men kommer i denna rapport att refereras som “de tre dimensionerna av datalogiskt tänkande”. Deras ramverk ses som både en bra riktlinje för vad undervisning i datalogiskt tänkande ska innehålla men fungerar också som ett övergripande sätt att förklara vad datalogiskt tänkande faktiskt innefattar. Ramverket har även använts som en bas för forskare i deras studier, både empiriska och teoretiska (Nouri m.fl., 2020). De tre dimensionerna av datalogiskt tänkande grundar sig i programmeringsspråket Scratch, som i grunden är ett barnanpassat programmeringsspråk där barn och unga med hjälp av block instruerar programmet, likt man gör när man kodar, hur exempelvis en karaktär ska röra sig (Brennan & Resnick, 2012). Programmet har blivit väldigt populärt på skolor i de yngre åldrarna då det är en bra första ingång till att förstå hur de ska instruera ett program (Lye & Koh, 2014). Utifrån studier av barn och ungas användande av Scratch föreslog sedan Brennan och Resnick (2012) sitt ramverk för datalogiskt tänkande. Ramverket har senare även nyttjats för att studera hur datalogiskt tänkande synliggörs och används vid svårare programmering som innefattar kodning (Nouri m.fl., 2020).

Det ramverk som Brennan och Resnick (2012) föreslog summerar det datalogiska tänkandet i tre olika dimensioner, där varje dimension innefattar olika delar (se Tabell 1). Den första dimensionen kallar de för *datalogiska koncept* (från engelskans “computational koncepts”; egen översättning) och är de olika koncept som används när någon exempelvis programmerar. Begreppen är viktiga då de finns en gemenskap för dessa inom i princip alla program som går att programmera i. Begreppen som ingår är sekvenser, loopar, händelser, parallellism, villkor, operatorer och data. En *sekvens* är, likt ett recept, ett stycke med instruktioner, exempelvis koder, som ska vara skrivna i en viss ordning för att datorn ska kunna tolka indata och utföra

aktiviteten. *Loopar* används för att utföra en aktivitet flera gånger. Exempelvis om någon vill testa sannolikheten för att få krona när den singlar slant och vill testa detta en miljon gånger, kan en använda en loop, istället för att skriva ut koden en miljon gånger. *Händelser* är i detta fall ett sätt att förklara ett händelseförlopp som beror på någon viss information. Exempelvis skulle det kunna vara "om man trycker på knappen så börjar klockan ticka". Begreppet *parallellism* innebär egentligen bara att flera sekvenser utförs samtidigt. *Villkor* är ett väldigt viktigt begrepp inom programmering. Det används för att styra koden beroende på vad som sker, där en oftast använder exempelvis "if-satser" som enklast förklaras som "Om X, gör Y". Begreppet *operatorer* används mellan olika koder för att instruera programmet vad den ska göra. Det skulle exempelvis kunna vara tecknet "+" mellan två textsnuttar, som då indikerar att dessa två ska sättas ihop till en mening. Sista konceptet som ingår i den första dimensionen av datalogiskt tänkande är *data* och syftar då till att programmet kan lagra, återkalla och uppdatera olika värden på variabler.

Den andra dimensionen kallar Brennan och Resnick (2012) för *datalogiska tillvägagångssätt* (från engelskans "computational practices"; använt översättning i Mannila, 2017, s. 82). Denna dimension fokuserar på hur en lär sig och inte vad. Beroende på hur mycket kunskaper och erfarenheter en person har inom programmering bemöter den problem på lite olika sätt. Denna dimension består av fyra olika delar, att arbeta inkrementellt och iterativt, att testa och felsöka, att återanvända och omformulera och slutligen att abstrahera och modellera. Att arbeta *inkrementellt* och *iterativt* förklaras enklast med att en arbetar utifrån en grund, exempelvis en påbörjad algoritm, där en stegvis försöker lägga till nya eller omarbetade delar i algoritmen för att förbättra den utifrån givet syfte. En annan väldigt viktig del inom programmering är att testa sina koder och felsöka, då det sällan är att koden skrivs på rätt sätt och utför precis det som en vill. Att behärska detta arbetssätt blir viktigare ju bättre man blir på att programmera, då algoritmerna oftast tenderar att bli väldigt långa och det blir då väldigt svårt att finna småfel. Att arbeta med *återanvändning* och *omformulering* innebär att en nyttjar algoritmer eller mindre sekvenser som en själv eller någon annan skrivit för ett visst ändamål. Denna är kanske inte perfekt anpassad för ens egen kod och då gäller det att göra om den på ett sådant sätt att det går att använda den. De sista delarna i den andra dimensionen är att *abstrahera* och *modellera*. Att abstrahera innebär att försöka minska komplexiteten genom att gå från helheten och fokusera på det som är lite mindre och som är viktigt utifrån givet problem. Med hjälp av detta kan man sedan modellera de små viktiga delarna och skapa något större.

Den tredje dimensionen kallar Brennan och Resnick (2012) för *datalogiskt perspektiv* (från engelskans "computational perspectives"; egen översättning) där ordet perspektiv syftar till att fokuset breddar sig från programmeringen och ser aktiviteten mer översiktligt. Här vill man komma åt det som programmeringen ger mer än att en kan skriva koder, där de valt att sammanfatta dimensionen med begreppen att uttrycka, interagera och ifrågasätta. Den första delen, *uttrycka*, beskriver de som motsatsen till att bara se appar och program som konsumtion utan istället ett sätt att ta inspiration för sitt egna kreativa skapande. "A computational thinker sees computation as a medium and thinks, "I can create." and "I can express my ideas through this new medium"." (Brennan & Resnick, 2012, s. 10). Den andra delen i tredje dimensionen, *interagera*, innebär att man ska arbeta tillsammans med andra. Dels för att få hjälp och att hjälpa andra och dels för att utveckla själva samarbetet. I detta avseende menar man att ens förståelse för interaktionen mellan människor kan göra att de tillsammans utvecklar större och bättre algoritmer, vilket därmed även ökar förståelsen för det datalogiska tänkandet. Den sista delen, *ifrågasättande*, handlar om att inte känna sig frånkopplad från tekniken utan våga ifrågasätta och ställa sig kritisk till sin situation. Det handlar även om att testa gränserna och ifrågasätta det som kan ses som något givet.

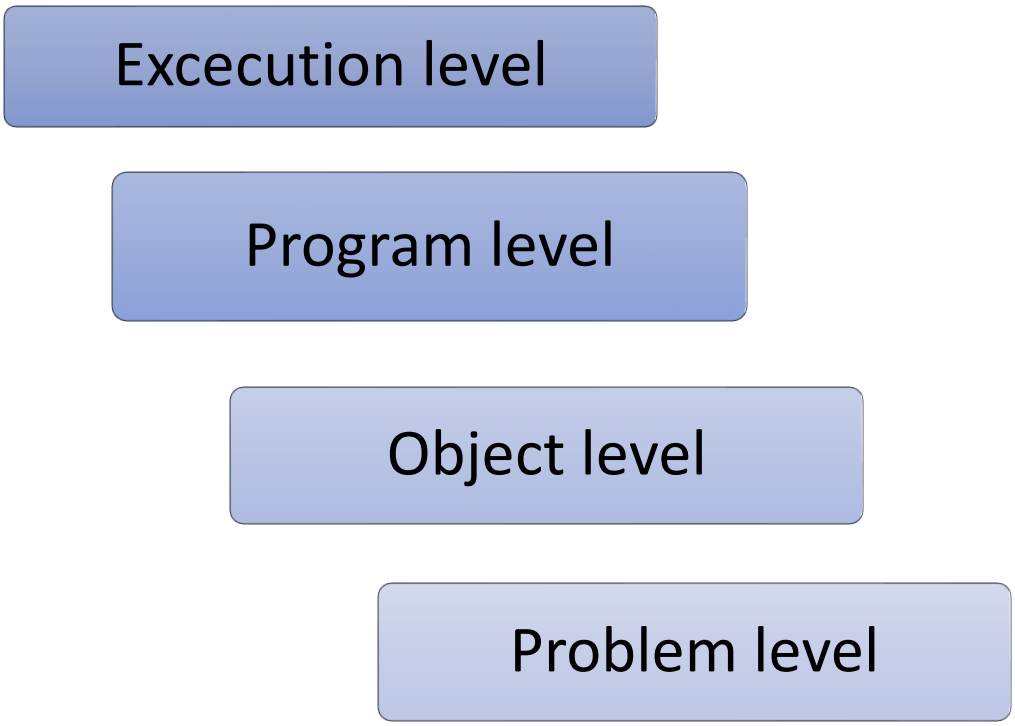
Tabell 1. En tabell som dels sammanfattar de olika delarnas innebörd och dels illustrerar tillhörigheten av de olika begrepp som presenterats ovan. Tabellen är skapad utifrån Brennan och Resnicks ramverk (2012) med inspiration från en tabell skapad av Nouri m.fl. (2019).

Dimensioner	Definitioner
Datalogiska koncept:	Koncepten som används, exempelvis när en programmerar. Innefattar: • Sekvenser • Loopar • Händelse • Parallellism • Villkor • Operatorer • Data
Datalogiska tillvägagångssätt	Syftar till arbetssättet i tänkandet och lärandet, fokus på hur du lär istället för vilka förmågor som utvecklas, när du exempelvis programmerar. Innefattar: • Vara inkrementell och iterativ • Testa och felsöka • Återanvända och omformulera • Abstrahera och modellera
Datalogiska perspektiv	Perspektiv som formas både om sin omvärld och en själv, innefattar: • Uttrycka • Interagera • Ifrågasätta

2.4.2 PGK-hierarkin

I avsnitt 2.3.1 nämndes Selbys (2015) definition som utgick ifrån blooms taxonomi och som kategoriserade olika kunskapsnivåer i sex olika steg. Perrenet, Groote och Kaasenbrood (2005) gjorde en liknande modell för utvecklandet av datalogiskt tänkande som de kallade PGK hierarkin (från engelskans "PGK-hierarchy", egen översättning). Den består istället av fyra nivåer som, på samma sätt som Selbys definition, rangordnar olika förståelsenivåer (se Bild 1). Den första och lägsta nivån kallar de "execution level". På denna nivå har man en grundläggande förståelse för algoritmer och kan felsöka och förstå att något blir fel i en kodning. Nivå två kallar de "program level", vilket är när en förstår algoritmer som instruktioner och kan dela upp en algoritm i mindre delar på sådant sätt att personen kan ta ett steg i taget i sin lösning. Det innefattar att personen vet i vilken ordning saker ska skrivas och även kan skriva koder i ett specifikt programmeringsspråk. Nivå tre kallar de för "object level", och här tänker personen datalogiskt och är inte beroende av en viss programmeringsmiljö. Här känner personen igen vanliga mönster och kan generalisera abstrakta händelser genom att parametrisera dem för en bredare användning. Den sista och högsta nivån kallar de för "problem level" där personen ska se algoritmen som en svart box med inställningen "givet ett problem, vilken typ av algoritm är anpassningsbar?". Personen har alltså olika kategorier för givna problem som är kopplade till vissa algoritmer. På denna nivå gör personen även koder som anses vara "finare" där all onödig information skalats ner och förenklats.

Bild 1: En bild som illustrerar de fyra olika nivåerna av förståelse för datalogiskt tänkande som beskrivs för PGK-hierarkin. Bilden är utformad utifrån Perrenet, Groote och Kaasenbrood (2005) förklaring av de olika nivåerna.



3 Metod

Nedan kommer ett metodavsnitt presenteras som förklarar hur studien genomförts. Metodavsnittet börjar med en grundlig förklaring om hur insamlingen av data gjort och övergår sedan till hur urvalet valts ut. Metodavsnittet fortsätter sedan med en beskrivning av analysmetoden där det genomgående beskrivs hur den tematiska analysen använts i förhållande till denna studie. Avsnittet avslutas med hur arbetet förhållit sig till forskningsaspekter och trovärdighet.

3.1 Metod för datainsamling

Studiens syfte bygger på tidigare forskningsresultat om att matematiklärare inte känner sig tillräckligt förberedda för att undervisa i programmering. Målet med studien är att försöka förstå vad som gör att många matematiklärare känner sig otillräckliga i undervisningen med programmering. Därav faller det sig naturligt att göra en kvalitativ studie som används för att studera olika uppfattningar och fenomen på ett djupare plan (Jacobsson & Skansholm, 2019). Inom kvalitativa studier brukar forskare använda begreppet *livsvärld* för att försöka förklara hur en person uppfattar sig själv i förhållande till sin omgivning (Dalen, 2015). Kvale och Brinkmann (citerad i Dalen, 2015, s. 15) skriver:

> Den kvalitativa forskningsintervjun försöker förstå världen från intervjupersonens synpunkt, formulera meningen i människors upplevelser, ta fram deras livsvärld, innan man ger sig in på det vetenskapliga förklaringarna.

För att kunna göra denna djupdykning i lärarnas erfarenheter och uppfattning om deras situation var det lämpligt att använda intervjuer som datainsamlingsmetod (Jacobsson & Skansholm, 2019). För att säkerställa att de olika teman med dess underliggande frågor skulle besvaras av respondenterna användes en intervjuguide som en grundläggande struktur för genomförandet av intervjuerna. Utarbetandet av intervjuguiden presenteras i avsnitt 3.1.2. Efter samtliga intervjuer transkriberades intervjuerna för att sedan övergå i en analysprocess. Metoden för analysen beskrivs i avsnitt 3.3.

3.1.1 Val av intervjuform

En intervju kan se ut på väldigt många olika sätt beroende på vad forskaren vill undersöka. Generellt är forskare som utför intervjustudier relativt öppna inför sina studier, vilket innebär att den som intervjuar ska gå in i samtalet med en förutsättningslös inställning och vara beredd att uppfatta och bemöta uttalanden och situationer som man kanske inte förväntade sig. Det är däremot viktigt att poängtera att en förutsättningslös inställning inte innebär att intervjuaren kommer dit oförberedd, utan att hen ska vara påläst och förberedd att leda samtalet vidare utifrån de olika svaren som respondenten kan ge (Dalen, 2015). I denna studie var en av frågeställningarna att undersöka på vilka grunder matematiklärare anser att de inte är tillräckligt förberedda för att undervisa i programmering. Då kan inte intervjuaren gå in med inställningen att läraren känner på detta sätt, utan måste forma intervjun utefter hur respondenten uppfattar sin livsvärld.

3.1.2 Utarbetande av intervjuguide

Alla studier som innefattar intervjuer som datainsamlingsmetod kommer bestå av någon form av intervjuguide (Dalen, 2015). En intervjuguide är till för att fungera som ett stöd för att alla viktiga teman och dess underliggande frågor besvaras under intervjutillfället. Dessa teman och frågor grundar sig oftast i den valda teorin för studien (Jacobsson & Skansholm, 2019), för att

senare kunna svara på studiens frågeställningar. Som tidigare nämnt i avsnitt 3.1.1 är intervjuer generellt formade utifrån ett öppet synsätt. Däremot skiljer man på olika intervjuformer utifrån dess öppenhet. Eftersom studien är riktad att svara på specifika frågeställningar utifrån något teoretisk som redan har kategoriserade nivåer och koncept passade det bättre att utforma en intervju med lite tydligare och strukturerade frågor (Dalen, 2015). Intervjun hamnar någonstans mellan det som kallas *strukturerad-* och *semistrukturerad intervju*. En strukturerad intervju följer en vi en viss ordning på noga utvalda frågor, vilket gör att alla respondenter svarar på samma frågor i en viss ordning. Denna intervjuform används i kvantitativa undersökningar för att det oftast finns ett begränsat antal sätt att svara på frågan på. En semistrukturerad intervju består istället av olika teman med ett antal frågor, där frågorna inte är lika begränsade och inte heller ställs i någon specifik ordning, utan kommer naturligt beroende på hur samtalet utvecklas (Jacobsson & Skansholm, 2019). Min intervjuguide (se Bilaga 1) är en blandning av dessa, då huvudfrågorna i intervjuguiden är ganska fasta, har en viss mängd av svar och bör komma i en viss ordning, medan följdfrågorna är mer av öppen karaktär och egentligen inte behöver ställas till alla respondenter, därav mer en semistrukturerad intervju.

Intervjuguiden konstruerades enligt *områdesprincipen*, vilket innebär att intervjun börjar med frågor i periferin och sedan tar sig in mot de mer centrala och utmanande frågorna som mer är av relevans för studien. Mot slutet av intervjun återgår intervjun sedan till lite öppnare frågor som inte är lika känsliga för respondenterna (Dalen, 2015). Intervjuguiden kan delas in i fem olika teman. Det första temat fungera som en introduktion där bland annat intervjuarens bakgrund, studiens syfte och annan information angavs. Det var även under den första delen som intervjuaren efterfrågade godkännande om inspelning för vidare analys. Det andra temat fick namnet “grundläggande fakta” där frågor om ålder, yrkeserfarenhet och liknande ingick. Tema tre rörde lärares kunskaper inom programmering, där frågor om utbildning, intresse och försök till förklaring av programmering ställdes. Tema fyra är den mer djupgående delen och innefattar flest frågor som är relevanta för studien. Temat fick namnet “lärarens undervisning” och bygger på frågor som besvarar lärarens kunskaper och brister i undervisning av programmering. Temat startade med ett påstående, som är ett utdrag från en studie på svenska matematiklärare, att lärare inte känner sig tillräckligt förberedda för undervisning i programmering. Respondenten förväntas ta ställning till påståendet och därefter också motivera sitt svar. Att lyfta forskningsresultat och be respondenten ta ställning och berätta vad den anser är ett väldigt bra sätt att få respondenter att öppna upp sig och med egna ord förklara hur den uppfattar en situation utifrån dennes åsikter (Dalen, 2015). Sista temat är av mer öppen karaktär och fick namnet “övergripande om programmering i matematiken”. Här är frågorna egentligen lite av samma karaktär som föregående grupp, men här förväntas inte respondenten “avslöja” sina egna brister utan består av frågor som är av mer generell karaktär, exempelvis “Vilka är de viktigaste egenskaperna en behöver för att kunna undervisa i programmering?” (Se Bilaga 1)

Genomgående i konstruerandet av intervjuguiden övervägdes hur frågorna skulle ställas för att de skulle förstås på rätt sätt och även ge svar på det som efterfrågades. Dalen (2015, s. 36) har sammanfattat fem kriterier för detta:

- Är frågan klar och otvetydig?
- Är frågan ledande?
- Kräver frågan speciell kunskap och information som informanten kanske inte har?
- Innehåller frågan alltför känsliga saker som informanten kommer att vägra uttala sig om?
- Ger frågeställningen utrymme för att informanten kan ha egna och måhända och otraditionella uppfattningar?

Alla dessa kriterier gick inte att bemöta för alla frågor, då vissa av frågorna var av sådan karaktär att man exempelvis vill veta hur lärare förklarar vad programmering innebär. En person med begränsade kunskaper sätts lite på plats, men det blir också intressant utifrån intervjun att se dels vad personen säger och dels hur personen reagerar på en sådan fråga.

3.2 Urval

En viktig del för läsaren i att kunna bedöma giltigheten i resultatet för studien är att läsaren kan ta del av urvalet, där hen ska få reda på hur många som deltagit, vilka dessa är och varför just dessa varit med i studien (Jacobsson & Skansholm, 2019). Urval kan göras på lite olika sätt beroende på förutsättningar. Urvalet i denna studie bestämdes genom en blandning av två urvalsstrategier, främst genom det som Dalen (2015) kallar *kriterieurval* men även med inslag av det som Jacobsson och Skansholm (2019) kallar *bekvämlighetsurval*. Kortfattat kan kriterieurvalet förklaras som att forskaren väljer vissa kriterier för sin undersökning, där mängden kriterier sedan begränsar urvalet på ett sådant sätt att bara en viss grupp kan delta. Det kriteriet som först bestämdes var att deltagare i studien måste ha lärarlegitimation och vara behörig lärare i ämnet matematik, då studien inte fokuserar på programmering i allmänhet utan med fokus inom ämnet matematik. Det krävdes även att respondenterna skulle vara aktiva lärare, främst med anledningen att programmering inte funnit speciellt länge inom matematiken och därmed säkerställdes det att lärarna arbetat med den uppdaterade ämnesplanen. Det tredje kriteriet begränsade på vilken nivå respondenterna undervisade, då studien fokuserar på gymnasielärares uppfattningar. Bekvämlighetsurval innebär att forskaren antingen väljer enheter som är nära, lättillgängliga eller där forskaren redan har en kontakt (Jacobsson & Skansholm, 2019).

Respondenterna som var med i studien kommer från fyra olika skolor, varav fyra var män och två var kvinnor. Lärarnas ålder och deras arbetserfarenhet som lärare skiljde sig mycket. Lärarna var mellan 25 och 47 år och hade mellan 1-21 års erfarenhet. Lärarna var spridda över fem olika program. Tidsaspekten påverkade omfattningen, vilket gör att antalet deltagare inte kunde uppnå den mängd som skulle önskats i en “riktig” forskningsstudie.

3.3 Genomförande

Vid sökandet av respondenter försökte jag få med personer med olika egenskaper. Exempel på dessa egenskaper som togs i beaktning var ålder, kön, erfarenhet och program läraren undervisar på. De flesta i studien har jag inte haft någon kontakt med tidigare, däremot har nästan alla deltagare kontaktats genom bekanta. Totalt kontaktades fem lärare via bekanta, varav två tackade nej. Det skickades även ett mejl till två rektorer på två stora skolor i Göteborg, detta förtydligas i nästa stycke. Genom rektorerna kom ytterligare en person med i studien. När respondenter tillfrågades gjordes detta delvis utifrån att kunna besvara frågeställningarna. Första frågeställningen togs inte i beaktning, då tanken var att slumpen skulle avgöra. Däremot valdes en av lärarna utifrån att kunna besvara den andra frågeställningen, eftersom den handlade om att jämföra skillnader mellan lärare som kände sig förberedda och lärare som inte gjorde det. För att kunna göra denna jämförelse behövde jag säkerställa att jag hade åtminstone en lärare som kände sig förberedd för att undervisa i programmering. Denna person var den som kontaktades via tips från rektorn där mejlet specificerade att jag sökte en lärare som känner sig bekväm med programmering och var legitimerad matematiklärare.

Samtliga sex personer som godkände att vara med i studien kontaktades via ytterligare ett mejl, där alla fick liknande meddelande. Det två personerna som hade en personlig kontakt med mig

sedan tidigare fick ett lite annorlunda mejl. Informationen som skilde var främst i presentationen om vem ansvarig för studien var och liknande. Resten av mejlet innehöll samma information till samtliga deltagare. Informationen som fanns i mejlet innefattade studiens syfte, vilka kriterier som behövde uppfyllas, vad som önskades av deltagarna och även ett visst förslag på dagar som personerna själva kunde välja tider på.

Eftersom studien gjordes under en pandemi rekommenderades det inte att ha fysiska träffar med respondenterna, istället gjordes intervjuerna genom kommunikationsprogrammet "Zoom". Intervjuerna utfördes med både ljud och bildupptagning för att få med både vad respondenterna sa och även deras visuella uttryck för att göra det så likt en "verklig" situation som möjligt. För att se hur all teknik fungerar och även ifall intervjuguiden ge svar utifrån det som önskas gjordes en pilotstudie. Planen var att testa detta på den första intervjun och att den inte skulle räknas med i resultatet. Efter genomförd intervju var det inget som var oklart och allt gick bra med tekniken och därmed tog detta ändå med som en del i resultatet. Eftersom personen inte var medveten om att det endast var ett test antogs det inte påverka resultatet.

Eftersom alla intervjuer gjordes via Zoom spelades både ljud och bild in, vilket låg till grund i beslutet att inte föra anteckningar utan istället försöka hålla ett naturligt och avslappnat samtal. Efter avslutade intervjuer transkriberades samtliga inom tre dagar, utöver en intervju som transkriberades ungefär en vecka efter. Transkriberingarna är gjorda på ett sådant sätt att även en utomstående person skulle kunna ta över studien, då texterna som skrivits speglar precis det som respondenten angav. Detta är en viktig del i transkriberingen för att kunna göra en så giltig tolkning som möjligt (Dalen, 2015). Däremot utelämnades utfyllnadsljud som exempelvis "eh..." och även början av meningar som på något sätt blev felaktiga eller tomma på information. Genomförandet av själva analysen presenteras inte exakt i denna studie, däremot presenteras analysmetodens upplägg i avsnitt 3.4.

3.4 Analysmetod

För denna studie kommer analysmetoden tematisk analys att användas. Den används för att analysera, förklara, hitta mönster och gruppera den insamlade datan i olika teman, därav namnet "tematisk". Denna analysmetod valdes dels på grund av dess relativt enkla upplägg vilket är fördelaktigt för oerfarna forskare och dels att den är applicerbar utifrån många olika teoretiska utgångspunkter och därmed även utifrån denna studies syfte. Analysmetoden utvecklades av Virginia Braun och Victoria Clarke (2006) och kommer också följa de sex steg i analysprocessen som de beskriver. Innan stegen förklaras och motiveras utifrån studiens syfte ska upplägget inför analysen presenteras. Analysen utgår från en deduktiv ansats, vilket innebär att studien utgår från något teoretiskt för att sedan analysera datan och kategorisera den utifrån teorins givna uppdelning (Braun & Clarke, 2006). Denna studie kommer, precis som det presenterades i 2.3.2, utgå utifrån ett teoretiskt ramverk av Brennan och Resnick (2012) för att placera in vilka delar av det datalogiska tänkandet som lärarna lyfter och sedan placera dessa delar utifrån deras förståelse i de olika kategorierna som PGK-hierarkin består av. Samtidigt förväntas inte allt kunna placeras in enligt teorin, då studien teorin främst syftar på det kunskapsmässiga. Analysarbetet kommer göras utifrån vad respondenterna säger och inget annat. Detta är ett semantiskt tillvägagångssätt, vilket innebär att det alltså inte kommer ske någon tolkning i underliggande förståelse eller uppfattningar om olika fenomen (Braun & Clarke, 2006).

Det första steget i Braun och Clarke (2006) analytiska process är att lära känna sin data. Detta görs genom att bland annat lyssna igenom på inspelningen, transkribera och sedan läsa sina

transkriberingar. Analysen startar dock redan vid det tillfället som intervjuerna genomförs eftersom intervjuaren ska vara uppmärksam på hur personen svarar på frågorna, hur den reagerar och hur respondenten upplevs genom bland annat kroppsspråket (Dalen. 2015). Utifrån det en ser och hör vid intervjutillfällena börjar forskaren redan då fundera på eventuella mönster och teman som kan vara aktuella, vilket starten av analysprocessen. Dalen (2015) menar att det oftast är bäst att skriva anteckningar på allt som forskaren noterar utöver just det som sägs och spelas in.

Steg två i processen är att ska skapa koder som representerar den insamlade datan (Braun & Clarke, 2006). Kodningen är ett sätt beskriva den stora datamängden i en mindre omfattning i form av nyckelord, som forskaren tror den kan använda för att besvara studiens frågeställningar (Jacobsson & Skansholm, 2019). Kodningen gjordes enligt det som Corbin och Strauss (refererad i Dalen, 2015) kallar för öppen kodning, med syftet att på ett systematiskt sätt beskriva datan utan att fokusera för mycket på teoretiska ramverket. För att göra detta tydligt och även underlätta för resten av analysen användes färgkodning, där varje färg representerade en viss kod. Färgkodningen gjordes i Word-dokument då det flertalet gånger hände att flera koder gjordes om till en kod, eller tvärtom, vilket underlättade tydligheten eftersom färger enkelt ändras i Word-dokument jämfört med att använda penna och papper. Det tredje steget i den tematiska analysen är enligt Braun och Clarke (2006) att börja identifiera olika teman. Detta görs genom att ta ett steg tillbaka och översiktligt arbeta med koderna som tagit fram för att kategorisera de under olika teman. Braun and Clarke (2006) rekommendera att använda sig av exempelvis en tematisk karta, som fungerar likt en tankekarta, för att finna kopplingar mellan olika koder. Det skulle exempelvis kunna vara några koder kan sammanfattas under ett visst tema, som i sig är en underkategori till ett annat tema.

Det fjärde steget kan ses som själva analysen av framtagna teman. Här går forskaren igenom alla teman och dess koder för att se om de kan göras om eller slopas. Exempelvis kan teman som är lika slås ihop, teman med för olik eller för lite innehåll slopas, stora teman bryts ner till två eller fler teman och så vidare. Braun och Clarke (2006) skriver att det fjärde steget innehåller två olika faser av analysen. Den första fasen handlar om att gå tillbaka till insamlade datan för att se om de koder och teman som tagits fram verkligen visar på liknande mönster och därmed kan kategoriseras på det sätt som en gjort i steg 3. När forskaren gått igenom alla teman och "godkänt" dessa har hen troligen skapat en ny tematiska karta med nya teman och kopplingar. När detta är gjort går analysen över till fas två, vilket innebär att forskaren ska säkerställa att den nya uppsättningen av teman i sin tematiska karta verkligen representerar den insamlade datan. Braun och Clarke (2006) förklarar detta som en kontroll av validiteten för analysarbetet.

Det femte steget går ut på att definiera och förfina de teman som tagits fram i föregående steg. Här handlar det om att precisera namnen på de teman som tagits fram och även vilka aspekter som ska ingå för att läsare av studien ska kunna förstå precis vad som ingår inom temat. Det sjätte steget är själva skrivandet av rapporten. Här ska den tematiska kartan översättas och presenteras i skrift med motivering till varje tema med utgångspunkt i insamlade datan.

3.5 Forskningsetiska aspekter

Studien har följt de fyra forskningsetiska krav som Vetenskapsrådet (2002) tagit fram. Dessa krav gäller forskning inom humanistiska och samhällsvetenskapliga ämnen och är applicerbar på denna studie. Vetenskapsrådets fyra krav är informationskravet, samtyckeskravet, konfidentialitetskravet och nyttjandekravet. Informationskravet innebär att forskaren ska informera deltagarna om syftet med studien och även om att det är deltagarens roll att bestämma

om hen vill vara med och att personen även får avbryta om det önskas. Samtyckeskravet syftar till att forskaren behöver få personerna godkännande om deltagande i studien. Det innebär även att personen själv bestämmer om den anser att det är rimliga förutsättningar och under hur länge den vill vara delaktig utan att det följer några negativa följder för hen. Konfidentialitetskravet innebär att alla personuppgifter och andra känslig etiska uppgifter kommer behållas inom studien och att forskaren har tystnadsplikt. Nyttjandekravet innebär att informationen som samlas in endast får användas till det syfte som forskaren förmedlat i studien och inget annat (Vetenskapsrådet, 2002).

Gällande informationskravet har samtliga deltagare har fått ta del av studiens syfte och även vad som kan förväntas om de deltar i studien. Alla informerades om hur lång tid som det kan förväntas ta och även själva bestämma när de ville ha intervjun. Deltagarna fick även inför intervjutillfället ta del av förutsättningarna och förväntad längd, vilket de även fick godkänna som en del av samtyckeskravet. En viktig del i samtyckeskravet var att de skulle godkänna att samtalet spelades in, både ljud och bild. Deltagarna blev även informerade om att personuppgifter och annan känslig information endast kommer hanteras av mig och i vissa fall möjligen av min handledare, vilket är en del av konfidentialitetskravet. De fick även informationen om att varje enskild intervju varken skulle spridas eller sparas utan raderas direkt efter avslutad studie. Slutligen tydliggjorde även användningen av materialet, där respondenterna blev informerade om att det endast kommer nyttjas till just det syftet som de fått ta del av, vilket svarar för nyttjandekravet.

3.6 Trovärdighet

I vetenskapliga arbeten likt denna är det viktigt att ha en hög trovärdighet. Med detta menar man oftast att forskningsbaserade studien ska ha hög validitet och reliabilitet (Jacobsson & Skansholm, 2019). En viktig del i trovärdigheten är att använda sig av teorier och metoder som tidigare påvisats fungera. Nedan följer en beskrivning av hur dessa har tagits i beaktning när studien planerats och utförts.

3.6.1 Validitet

Validitet är ett sätt att beskriva mätningens relevans, Nationalencyklopedin definierar validitet som “den utsträckningen i vilken ett mätinstrument mäter det som man avser att mäta” (citerad i Jacobsson & Skansholm, 2019). Studien avser att undersöka vad som gör att matematiklärare inte känner sig förberedda på att inkludera programmering i sin matematikundervisning. Dalen (2015) lyfter tre delar av validiteten som i synnerhet har präglat studiens utformning och genomförande. Det första är validiteten i urvalet. Inom kvalitativa studier väljer forskare generellt ut sitt urval utifrån ett specifikt ändamål, vilket ibland kan göra det svårt att dra generella slutsatser. Denna studien görs i sådan liten omfattning att det blir ännu svårare att dra generella slutsatser på större populationer. Studien görs på sex lärare från fyra olika skolor i anslutning till Göteborg, därav är det heller inte rimligt att dra slutsatser hur det ser ut i Sverige eller ens i Göteborg. Däremot beskrivs tydliga kriterier för att de som deltar ska vara aktuella för studien och det är därav inte heller helt meningslöst att fortsätta undersöka de mönster som presenteras i denna studien. Det andra som Dalen (2015) lyfter som tagits i beaktning är val av metod. I denna studie används teorier som använts i flera tidigare studier och som inom forskningsvärlden anses vara valida teorier. Även tematisk analys är en väl beprövad metod och anses vara valid. Undersökningens frågeställningar och problemformulering är av sådan karaktär att det ska vara möjligt att genomföra enligt metoden och teorierna som tidigare presenterats, där även intervjun är utformad på sådant sätt att det ska kunna svara på studiens

frågeställningar. Det tredje som Dalen (2015) lyfter är tolkningsvaliditet, som istället påverkar validiteten utifrån hur man som forskare försöker förstå respondenterna i sin studie. Tolkningen av datan har försökts göras utifrån ett så neutralt synsätt som möjligt, samtidigt har tolkningen troligen påverkats delvis utifrån ens uppfattningar från intervjutillfällena.

3.6.2 Reliabilitet

Reliabilitet beskriver studiens tillförlitlighet, alltså ett mått på hur väl studien mäter det som är avsett att mätas. För att en studie ska anses ha hög reliabilitet ska studien innehålla så få slumpmässiga mätfel som möjligt (Jacobsson & Skansholm, 2019). Jacobsson och Skansholm (2019) menar att det huvudsakligen kan ske mätfel i tre delar av undersökningen. Den första delen är mätinstrumentet, alltså själva metoden. Denna studie bygger på flera välkända vetenskapliga artiklar och nyttjar en analysmetod som anses vara väl beprövad, därmed kan studiens metod anses vara reliabel. Den andra delen avser genomförandet av studie, exempelvis om respondenterna känner sig trygga, om det finns tillräckligt med tid och liknande. Här har jag gjort flera insatser för att göra detta så reliabelt som möjligt genom att i förväg informera respondenterna om syftet och ungefärligt innehåll, avsätta längre tid än vad intervjun förväntades ta, forma en intervjuguide på ett sådan sätt att frågorna skulle besvara frågeställningarna och att respondenterna skulle känna sig bekväma med frågorna. Däremot kunde reliabiliteten kunnat vara högre på denna del då undersökningen gjordes under en period som flertalet lärare menade var “som stressigast” och att flertalet inte var vana vid att använda kommunikationsprogrammet Zoom. Båda dessa kunde inte göras annorlunda på grund av att kursen är placerad där den är och dessutom under rådande pandemi. Den tredje delen innefattar deltagarna i studien, det vill säga både forskaren och respondenterna. Forskaren måste vara väl förberedd och säker i sin roll, samtidigt som respondenterna måste kunna representera en grupp utifrån givet syfte och dessutom inte känna sig tvingade. Exempelvis skulle det kunna vara om frågan ställdes till en rektor som sedan bett en lärare vara med i studien. För att förbereda inför intervjutillfällena lästes intervjuguiden igenom flertalet gånger och potentiella följdfrågor dokumenterades. Däremot kan en ifråga reliabiliteten eftersom intervjuaren i denna studie inte var erfaren. Vissa deltagare i studien visade smått motstånd i att vara med på grund av att de ansåg att de inte hade tillräckliga programmeringskunskaper. Detta bemöttes genom att motivera att även “mindre bra” kunskaper var ett intressant resultat och att studien inte gick ut på att testa deras kunskaper och heller inte att personlig information skulle ingå. Ingen av deltagarna har visat motstånd efter denna motivation och flertalet har visat intresse för studiens resultat.

4 Resultat

Resultatet börjar med att presentera respondenternas uppfattning om deras förmågor i förhållande till programmering i sin undervisning och även lite om deras bakgrund. Resultatet följer därmed av två avsnitt som är slutprodukten av den tematiska analysen som genomförts. Dessa används för att svara på samtliga frågeställningar, där det kommer vara en mer direkt koppling till de två första frågeställningarna. Den tredje frågeställningen kommer istället behandlas på ett djupare plan i diskussionen i avsnitt 5.1.3. För att tydligt försöka presentera resultatet har därmed de två första frågeställningarna delats upp i två olika avsnitt i resultatet.

4.1 Presentation av lärarna

För att behålla personerna så anonyma som möjligt kommer kandidaterna presenteras som L1, L2, L3, L4, L5 och L6. Siffrorna representerar i vilken ordning transkriberingen av intervjuerna gjordes. Samtlig information i detta avsnitt är uppbyggt direkt utifrån svaren från intervjuerna och är inte en del av den tematiska analysen. Svaren är bygger på lärarnas egna uppfattningar och har inte satts i relation till något eller någon annan.

L1 är en relativt ny lärare och har endast arbetat i ett år. Personen har läst ämneslärarprogrammet och är behörig i matematik. Personen anser att hen är relativt förberedd för att undervisa i programmering men poängterar samtidigt att hen aldrig har undervisat i programmering.

L2 har arbetat som gymnasielärare i nio år och är legitimerad matematiklärare. Personen anser att hen definitivt inte är tillräckligt förberedd för att undervisa i programmering. Personen har aldrig inkluderat programmering i sin matematikundervisning

L3 har på olika sätt arbetat som lärare i elva år men har endast varit gymnasielärare i fyra år av dessa. Personen har tidigare arbetat på bland annat grundskola men även på olika vuxenutbildningar. L3 är legitimerad matematiklärare. L3 anser att hen inte är tillräckligt förberedd för att undervisa i programmering i matematikundervisningen. Har använt programmering enstaka gånger i sin undervisning

L4 är legitimerad gymnasielärare i matematik och har undervisat i 21 år. L4 anser att hen delvis är förberedd för att undervisa i programmering. Å ena sidan har hen grundläggande kunskaper å andra sidan anser hen inte att det är tillräckligt mycket och känner sig väldigt begränsad. Hen inkluderar programmering i sin matematikundervisning vid enstaka tillfällen per år.

L5 har varit lärare i 18 år och är legitimerad matematiklärare. Studerade först två år som civilingenjör innan hen valde att bli lärare istället. L5 anser att utifrån premisserna som Skolverket anger så är hen tillräckligt förberedd för att undervisa i matematik med inslag av programmering och är bekväm med det. Däremot anger även personen att hen skulle önskat att kunna mer för att utveckla sina lektioner. L5 använder inte programmering speciellt ofta i sin undervisning men det händer.

L6 har varit gymnasielärare i matematik i ungefär 13 år. L6 har en magisterexamen i matematik och fysik och har sedan läst ett kortare program för att bli behörig matematiklärare. L6 anser absolut att hen är tillräckligt förberedd för att undervisa i programmering i sin matematikundervisning. L6 använder programmering återkommande i sin undervisning och anpassar mängden beroende på innehåll.

För att kortfattat sammanfatta ovan ska lärarna grupperas i mindre grupper. L2 och L3 är de som anser att de är minst förberedda, L1 och L4 som att de har någorlunda förståelse för programmering men inte bekväma med att undervisa i programmering. L5 kategoriseras ensam och menar att hen är tillräckligt förberedd för att undervisa och även bekväm med det, däremot hade hen önskat att hen kunde mer. L6 är tillräckligt förberedd och känner sig bekväm och nöjd med sin undervisning i programmering. Notera att dessa grupper inte är något som tagits fram i efterhand utan grupperingen är gjord på lärarnas egna uppfattning. Det innebär att resultatet kan likna mellan grupper och att personer inom samma grupp kan få olika resultat. Grupperingen används för att enklare tydliggöra de mönster som hittas.

4.2 På vilka grunder anser matematiklärare att de inte är tillräckligt förberedda för att undervisa i programmering inom matematik?

Denna frågeställning fokuserar på de lärarna L1 till L5 som anser att de inte kan tillräckligt eller på något sätt känner att de saknar något sin undervisning i programmering. L6 som ansågs vara väl förberedd och bekväm i sin undervisning har också tagits med i denna del, där hen främst förklarar vad hen tror är anledningen till att många känner sig osäkra och även med inslag av eventuella lösningar. Samtliga teman som är framtagna under denna frågeställning är olika svårigheter som lärarna själv lyft eller antytt. Det har alltså inte skett någon tolkning av deras intervjuer för att komma åt svårigheter. Den tematiska analysen resulterade i tre olika teman som svarar på första frågeställningen, *Otydlig ämnesplan*, *Didaktiska aspekter* och *Tidsbrist i matematiken*.

4.2.1 Otydlig ämnesplan

Den mest förekommande anledningen till att lärarna inte kände sig tillräckligt förberedda var att de inte lyckats tolka vad de ska programmera i matematiken. Flera lärare lyfter detta som den främsta anledningen, där många brottats med hur de ska koppla ihop programmering med matematiken. L6 säger exempelvis:

> Det största problemet är att man förklarar programmering som något så stort och att det tar jättelång tid att "bemästra" programmering i alla dess former, exempelvis mjukvaruutveckling och utveckling av spelappar. Hade man (skolverket) skrivit "konstruera algoritmer för beräkning med programmering" och preciserat lite mer inom vad hade det varit en helt annan sak, en liten procentandel bara.

Temat delas in i två olika underkategorier, *Vad lärare ska undervisa* och *Vad förväntas av eleverna*.

4.2.1.1 Vad lärare ska undervisa

Som tidigare nämnts hade samtliga lärare åsikter om hur Skolverket valt att presentera programmeringen som en del av matematiken. De finns flera olika delar som lärarna inte förstår men samtliga kan sammanfattas under denna underkategori. Det första handlar om en saknad förståelse för vad Skolverket menar med programmering.

> Jag inte vad programmering betyder i ämnesplanen som den står just nu. Innebär det att man ska använda sig av ett programmeringsspråk eller betyder det kanske något annat? - L5

> Jag vet inte vad man ska jobba med, alltså vad ingår i att programmera? [...] har jag främst programmerat i Geogebra - L1

> [...] exempelvis att man istället för programmering hade satt krav på att kunna använda grafiska hjälpmedel, exempelvis Geogebra, där det finns en direkt koppling till matematiken. - L4

Det blir här tydligt att lärarna har svårt att tolka vad som ingår i begreppet programmering och att det blir ett hinder i att förstå vad de ska använda. De sista två citaten är dessutom motsägelsefulla, där L1 anser att Geogebra är ett programmeringsverktyg och L4 önskar att lärare istället för programmering skulle använda Geogebra. I dessa uttalanden kan man även ana en saknad förståelse i hur lärarna ska koppla samman programmering med matematiken. Detta synliggörs på lite olika sätt beroende på lärarnas erfarenheter men samtliga indikerar samma problem. L2 och L3 som tydligast antydde att de inte var tillräckligt förberedd visade även störst saknad förståelse för hur detta skulle gå till.

> För det första har jag extrem dålig koll av vad som krävs… vad är det jag ska lära ut? Den änden är det största problemet. Om det bara är att programmera jo men visst då hade det varit enkelt men när det är just programmering och matematiken vet jag inte vad slutprodukten ska vara. Om tanken inte är att lära ut hur man programmerar förstår jag varken hur jag ska göra eller varför programmering finns med i matematiken. - L2

Däremot var detta inte något som bara L2 och L3 lyfte. Samtliga lärare utryckte en förvirring för vad Skolverket faktiskt ville att de skulle programmera och vad programmering betyder. L3 fick frågan om hen använt programmering i sin undervisning svarar hen “Ja, fast bara simpel programmering” och fortsätter:

> Alltså egentligen om saker som inte har med matematikkursen att göra. Det är mer att visa programmering men jag har tyvärr inte lyckats koppla det till innehållet i kursen. – L3

Även de lärare som ansågs vara förberedda visade indikationer på samma tankar. L5 berättar bland annat att hen främst använder programmering för att kunna bocka av att de jobbat med det, för att sedan fortsätta arbeta med matematiken. Flera av lärarna förklarar också deras förvirring genom att de inte tycker det är tydligt hur matematiska kopplingen görs i de exempeluppgifter och liknande som Skolverket publicerat. En del av lärarna har bland annat gått kurser som Skolverket anordnat som en del av fortbildningen inför införandet av programmering. L3, som också deltagit i en av dessa kurser som Skolverket anordnat, berättar följande:

> På utbildningar och även på Skolverkets hemsida fokuserar oftast på exempelvis det här med roboten (diskuterades precis innan om en exempeluppgift för gymnasiematematiken där de inkluderade styrning av en robot) och att man första kan skriva instruktioner på papper och sedan ska elever styra varandra och sen användning av scratch. Jag hade velat ha, som man kan få ibland från Skolverket, konkreta exempel på vad man ska göra, lite hur koppling är till matematiken. - L3

L6 anser att det är den bristande tydligheten av Skolverket i hur lärare ska använda sig av programmering i matematiken som är ett av de största problem. L6 har utbildat lärare på sin skola och berättar om lärares uppfattningar om programmering i matematiken, där en del av dessa deltagit på Skolverkets fortbildningskurser:

> [...] och alla tänkte direkt på mjukvaruutveckling typ “aha ska vi göra appar och robotar”. Det ska inte vara en del av matematikundervisningen, det sabbar ju det hela. I teknik är det relevant att styra en robot men det är väldigt “long shots” att göra det i exempelvis matematik 2c. - L6

Flera av lärarna uttrycker att det är just otydligheten som är svårigheten och egentligen inte det kunskapsmässiga. Samtliga lärare fick svara på frågan om de tror att de hade varit ett problem med programmeringen rent kunskapsmässigt om Skolverket hade gett tydliga instruktioner på precis vad programmeringen förväntas användas till i matematiken och i vilket syfte. Ingen lärare svarar egentligen nej på den frågan, utan de menar allihopa att bara det hade varit tydligt skulle de kunna läsa till sig kunskapen. Däremot när det blir så brett och otydligt vet de inte vart de ska börja.

> Nej hade jag vetat precis vad som förväntas hade jag med lite förberedelsetid nog löst det utan problem. - L1

> Nej det tror jag inte, med lite youtube så hade jag löst det på egen hand. Det är inte det kunskapsmässiga utan det är tydligheten som är problemet, att jag inte vet vart jag ska börja, exempelvis i vilket program? - L2

4.2.1.2 Vad förväntas av eleverna

Utöver att lärarna verkar vara överens om att det är otydligt vad de ska lära ut är det också otydligt vad eleverna ska förväntas göra. Några lärare menar även att eftersom det inte står något i kunskapskraven om programmering blir det svårt att vet hur de ska utforma lektionerna i programmering.

> Ska eleverna lära sig programmera eller ska de redan kunna det när de kommer till gymnasiet? Eller är tanken kanske bara att man ska visa? - L1

> Det som ändå är grundproblemet med programmering i matematiken är att det inte ska prövas. När det inte står med i kunskapskraven blir det också svårt att förstå om eleverna ska förväntas använda sig av programmering eller om det är läraren som ska göra det. - L4

Man kan även utläsa att lärarna inte vet hur de ska hantera det stora glappet mellan hur lärarna tolkar att Skolverket vill att eleverna ska använda programmering kontra hur mycket eleverna faktiskt kan. Alla utom L6 pratar även om en saknad motivation i att faktiskt ta tag i programmeringen. De menar att eftersom det inte finns med i kunskapskraven känns det inte heller som något viktigt utan mer att eleverna bara ska få se det. Otydligheten av vad som förväntas av eleverna blir ännu värre menar L3 och L5 eftersom Skolverket inte förtydligar något gemensamt språk. L5 uttalande nedan sammanfattar på ett bra sätt flera av lärarnas tankar.

> Man vill ju att eleverna ska förstå “ah! Vad bra det är att kunna programmera när vi exempelvis inte kan lösa något algebraiskt och istället kan lösa det numeriskt med programmering” men den tröskeln är alldeles för hög. Dels för att många kommer helt oförberedda (i programmering) till gymnasiet och dels för de som har erfarenheter i programmering oftast har det i olika programmeringsspråk. Hur gör man då? Det hade varit bättre om det fanns en röd tråd genom där de (Skolverket) skrev ut vad man skulle programmera i. Exempelvis “från årskurs 7 ska samtliga programmera i python”. Eller förväntas de kanske använda flera språk? - L5

Samtliga lärare visar på en ovisshet i hur de som lärare ska bemöta programmering. Enligt lärarna kommer ovissheten från en svag styrning i vad som förväntas. Detta leder även till att lärarna hellre väljer att fokusera på annat då de inte förstå hur tanken är att de ska använda programmering och därmed inte heller kan motivera varför eleverna ska använda det.

4.2.2 Didaktiska aspekter

Det andra temat som den tematiska analysen resulterade i var "didaktiska aspekter". Flera lärare lyfte utmaningar som har med det didaktiska att göra, men utmaningarna uttrycktes inte lika tydligt som i det första temat. Detta tema är mer som ett "steg två problem" i lärares undervisning. L6 fick frågan om hen tror att det didaktiska är en svårighet för lärare, varpå L6 svarade "Ja absolut, men inte än". Både L2 och L3, som inte ansåg sig själva vara förberedda för programmering, lyfter knappt några didaktiska svårigheter. Däremot är denna potentiella ovetskap om didaktiska svårigheter något som L2 ändå poängterar. L2 säger såhär:

> Jag känner nog att det pedagogiska är lugnt, jag vet hur man lär ut. Det är mest att jag inte vet vad jag ska göra med programmeringen. Samtidigt så vet jag ju faktiskt inte eftersom jag aldrig undervisat i det, känns enkelt att säga såhär på förhand. - L2

Temat har delats upp i två underkategorier för att belysa två olika aspekter som lärarna lyfter som problematiska. De två kategorierna är *skilda fokusområden* och *programmeringsdidaktik*.

4.2.2.1 Skilda fokusområden

Denna underkategori syftar till lärares svårigheter att få undervisning att fokusera på rätt saker, nämligen matematik. Med det menar respondenterna att det finns en utmaning i att läraren försöker planera och utföra en lektion utifrån ett matematiskt mål och att fokus istället landar på att kunna använda sitt tekniska hjälpmedel. Alla lärare utom L3 lyfter detta som en svårighet. L3 säger inte heller inte emot detta utan frågan diskuterades inte.

> [...] och det blir lätt ett metakognitivt skifte då. Lärarens syfte med lektionen är det matematiska innehållet medan eleverna istället fokuserar på att få programmeringen att fungera. - L1

Även L6 som ansåg sig vara tillräckligt förberedd för att undervisa i programmering menar att problemet finns.

> Mitt mål är att det bara ska vara fokus på matematiken och inte programmeringen, programmering för matematikens skull. Men såklart möter man lite motstånd från eleverna i början och det blir ett hinder att man glömmer ett kolon och så står det stilla för man gjort ett litet tekniskt fel. Just detta är ett stort problem med att eleverna tycker programmering är svårt. - L6

Flera av lärarna lyfter detta som ett problem men har egentligen ingen potentiell lösning på hur de ska komma runt det. Flera kopplar detta problem till det som presenterades i det föregående temat om vad lärare egentligen kan förvänta av eleverna. Samtidigt lyfter bland annat L4 och L6 att det kanske bara är en svårighet som finns i början eftersom eleverna är ovana vid att programmera i matematiken. L6, som använder programmering relativt ofta i sin undervisning, menar att det är viktigt att de får lära sig grunderna först och sedan gå in på det matematiska.

> De måste först lära sig det mest grundläggande, typ att sätta värden på variabler och sätta ihop en loop utefter ett visst mönster. När de fått göra det några gånger märker man att

> det blir ett annat sätt att arbeta med programmering, just det här "delta-tänket", att göra något i små steg för att få ihop en helhet. - L6

4.2.2.2 Programmeringsdidaktik

Från att i föregående kategori fokuserat på kopplingen mellan programmering och matematiken syftar denna kategori till det ämnesdidaktiska inom programmering. På samma sätt som tidigare lyfter egentligen varken L2 eller L3 något om de didaktiska svårigheterna utan det är de resterande som gör det mest. L1 säger exempelvis:

> Det hade nog behövts didaktiska kunskaper, precis som det behövs i alla andra ämnen. Jag kanske kan lösa en kod och få den att fungera men det betyder inte att jag kan förklara stegen för eleverna. - L1

Lärarna skiljer sig lite hur de ser på de ämnesdidaktiska svårigheterna. I citatet ovan av L1 är det fokus på vad som krävs för att eleverna överhuvudtaget ska förstå. L5 förklarar det istället som att det inte är ett problem, men öppnar sedan upp för att det kanske bara är att hens okunskap gör att hen inte vet om det. L5 beskriver det enligt följande:

> Jag tänker att har man varit lärare länge så har man en pedagogisk grund och behärskar lärandet. Eller sen kan det vara så att jag slår mig för bröstet och att man skulle behöva lite ämnesdidaktiska kunskaper för att utveckla sitt lärande, jag kanske missar något. - L5

L4 pratar om sin undervisning i programmering ur ett mer ambitiöst perspektiv och ser det som ett problem att hen bara "klarar av" att undervisa i programmering och egentligen hade velat ha de ämnesdidaktiska knepen för att kunna vara en bra lärare.

> Jag som är rätt bra i engelska tänker att jag hade klarat av att undervisa i engelska, men undra hur bra skulle det gå? Didaktiskt har jag ingen aning hur jag ska komma runt svårigheter för att nå kursmål och utveckla eleverna. På samma sätt är det med programmering. Jag har ingen didaktisk fantasi och kreativitet utan jag bara gör. - L4

Avsaknaden av didaktiska kunskaper menar L4 skapar en osäkerhet hos lärare som också gör att de hellre avstår från programmering. Detta som en följd av att lärarna känner att de många gånger inte kan hjälpa eleverna och därmed känner sig otillräcklig. Denna osäkerhet kring att inte kunna hjälpa eleverna är något som även L3 även antyder. L6 lyfter också problematiken med att inte kunna det didaktiska och poängterar att det hade kanske varit bra att gå en kurs i exempelvis "computational science" (liknande begrepp av datalogiskt tänkande). Hen förklarar att skillnaden mellan programmering och matematiken är själva arbetsprocessen, där övergången kräver didaktiska kunskaper från lärarens sida.

> [...] medan att tänka i små små steg är eleverna väldigt ovana vid. Exempelvis ränteberäkning. Elever är bra på att räkna ut hur mycket totalsumman blir efter 4 år, däremot är de mindre bra på att förstå hur man kan se varje månad som en utveckling, att räkna ut varje månad i en loop. Det blir då en didaktisk utmaning att få eleverna att förstå och tänka iterativt och inkrementellt för att bilda en helhet. - L6

4.2.3 Tidsbrist i matematiken

Det tredje och sista temat för den första frågeställningen är tidsaspekten i matematiken. Nästan alla lärare i studien menar att programmering är ett tidskrävande arbete och att de inte känner att de vet hur de ska göra för att det ska få en naturlig plats i matematiken utan att det ska vara

ett längre projekt. Några av lärarna förklarar detta väldigt kort och utan någon större fördjupning i problemet.

> Det finns för lite tid för att programmering skulle kunna ta en större plats än vad den gör just nu. - L1

> Man hinner inte sitta och lära ut programmering under lektionstid. -L2

Både L1 och L2 pratar om programmering som en tidskrävande aktivitet. Uttalandet av L1 uppfattas mer som att matematiken redan innehåller mycket och att programmering därmed inte kan får en större plats, medan L2 mer specifikt syftar till att det är just lärandet av att programmera som är tidskrävande. L5 menar att utifrån premisserna som Skolverket anger, att eleverna ska lära sig programmering i grundskolan, så finns det en plats för programmering i matematikundervisningen. Hen fortsätter sedan:

> Det känns dock som att man inte riktigt hunnit dit än och att eleverna inte har programmeringsvanan, då blir som sagt tröskeln för stor för att hinna med. - L5

L2 och L5 är inne på lite samma spår, alltså att tiden som kommer läggas på programmering främst kommer läggas på att få fungerande koder. L2, L3 och L4 nämner även att de hade velat ha ett snabbt sätt att kunna använda programmering så att de kan komma åt det matematiska, vilket i dagsläget saknas hos lärarna. L5 är mer inne på att eleverna måste kunna mer när de kommer till gymnasiet, medan exempelvis L3 anser att det är bristen på förståelsen hos lärare för hur man på ett tidseffektivt sätt ska kunna koppla innehållet till programmering.

> [...] och man måste ju koppla det till matematiken, men jag känner att tiden inte räcker till. [...] Visst jag hade kunnat lära ut programmering och att eleverna ska bygga ett program för att lösa en andragradsekvation, men då måste vi kanske spendera 10 lektioner för en så liten grej. Jag saknar hur man kan få in det på ett snabbt sätt för att lösa olika problem eller bygga en förståelse. - L3

Detta problem är även något L6 har stött på tidigare men som hen tycker fungerar bättre nu. På lektioner används oftast korta algoritmer och eleverna arbetar alltid med ungefär samma syntax. Däremot säger L6 också att detta varit ett större problem tidigare, men att man på skolan har försökt komma runt detta problem genom att lägga in 20 timmar extra programmering utanför matematikundervisningen under första läsåret för eleverna. Detta är något som beslutats på skolan och är inget hen själv bestämt.

4.3 Vad skiljer lärare som känner sig förberedda och lärare som inte känner sig förberedda?

Denna frågeställning syftar till att förstå skillnaderna mellan lärarna som ansåg sig själva vara förberedda och de som inte ansåg detta. Syftet här var att försöka hitta gemensamma faktorer mellan olika lärare som har gemensam syn på sin undervisning i programmering. För läsarens förståelse av nedanstående resultat rekommenderas en repetition av avsnitt 4.1 där lärarna presenteras och även kategoriseras in i olika grupper. Den tematiska analysen resulterade i två teman som innefattar underkategorier. De två teman som tagits fram är *Förståelse för programmering* och *Relation till programmering.*

4.3.1 Förståelse för programmering

Under analysarbetet blev det tydligt att det fanns skillnader i hur lärarna såg och uppfattade programmering, dels genom hur de diskuterade kring vad programmering används till och dels om deras uppfattning om programmeringens roll i skolan och i matematiken. För att skilja på de olika områden har temat delats in i tre kategorier, *programmeringens användningsområden, programmering som ett verktyg i matematiken* och *programmeringsfenomen*.

4.3.1.1 Programmeringens användningsområden

Denna underkategori uppstod nästan enbart utifrån en av frågorna där respondenterna förväntades förklara vad programmering är. Det preciserades inte att det skulle vara utifrån en matematisk synpunkt utan tanken var att se hur lärarna förstod programmering mer generellt. L2 och L3 hade ganska lika svar. Båda beskriver på en väldigt basal nivå om hur man styr en dator för att utföra ett arbete.

> Att skriva en kod med siffror, bokstäver och olika tecken som i slutändan gör att en dator utför ett arbete. [...] Jag tänker också på hur en dator ska funka, typ hur Word är uppbyggt. - L2

L2 fyller även i med åsikter om programmering, där hen menar att programmering egentligen främst är nyttigt inom teknik och naturvetenskap. Även L1 uttalar sig på liknande sätt och menar att det exempelvis elever som läser samhällskunskapsprogrammet inte kommer komma i kontakt med programmering.

> ”Samhällare” behöver ju ej det (programmering). I natur och teknik är det mer anpassningsbart då man kanske blir ingenjör eller liknande. Men en samhällsvetare eller beteendevetare kommer aldrig använda det. – L1

När det kommer till förståelsen för programmering påminner L1 och L4s uttalanden om varandra. Båda är nämligen inne på att kunna koda datorer för ett visst matematiskt utförande. Samtidigt visar också L4 förståelse för att det även finns ett annat användningsområde och nämner att hen behöver begränsa sig om hen ska kunna besvara på frågan.

> Jag tror jag behöver begränsa det lite då, för man kan ju egentligen programmera hela mitt liv. I matematiken används programmering till uträkningar som jag inte hade klarat av själv, exempelvis stora datainsamlingar. - L4

L5 breddar också sina tankar likt L4 genom att prata om simuleringar, samtidigt som hen också landar i det matematiska, att exempelvis approximera eller numeriskt beräkna något. L5 säger däremot inget om att koda dator utan pratar lite bredare om programmering. L6 konstaterar att programmering är svårt att sammanfatta kort då det är väldigt stort område. L6 säger:

> Jag tänker på programmering som två delar. Det ena som mjukvaruutveckling i att framställa en produkt, med en viss kreativ frihet i att skapa typ ett spel. Den andra delen handlar mer om algoritmer, där man använder programmering för beräkning, sortering av data, beräkningar som upprepas i ett visst mönster… Egentligen sådant som är opraktiskt att göra för hand. - L6

4.3.1.2 Programmering som ett verktyg i matematiken

Denna kategori uppkom främst från en följdfråga till föregående underkategori, där frågan för denna underkategori var att förklara programmeringens roll i matematiken. Däremot har flera uttalanden plockats från andra delar av intervjun då flera hoppade tillbaka till detta tema. Även här är L2 och L3 inne på lite samma spår, nämligen att programmering troligen är jättebra för matematiken, men att de egentligen inte riktigt vet för vad. L3 fick frågan om hen har förståelse för vad man kan använda programmering till inom matematiken där hen svarar:

> Nej, mer att jag vet att man kan än att jag vet vad man kan. Egentligen kan man nog göra hur mycket som helst, kan jag tänka mig i alla fall. Man kanske kan lösa ekvationer, titta på olika mönster och en normalfördelningskurva eller något. - L3

Både L1 och L4 beskrev den matematiska användningen redan när de skulle beskriva programmering mer generellt. Däremot fördjupar sig L4 under intervjuns gång och nämner fler möjligheter med programmering, vilket även L5 gör.

> Huvudsyftet med programmering i matematiken är att använda datorns kapacitet till det som min hjärna och min räknare inte kan. Att kunna simulera fram världen, sannolikheter och göra större datainsamlingar. - L4

> [...] för att ändra i indatan för att se hur det påverkar loopar i processen med iterationer. [...] Exempelvis att approximera något som inte går att lösa algebraiskt och istället använda programmering för att se numeriska lösningar.[...] genom att testa sig fram och felsöka sina koder, vilket också delvis ingår i det matematiska tänket, alltså just det här logiska tänkandet. - L5

L1, L4 och L5 diskuterar även programmering som ett verktyg i matematiken med fokus vad det ger för eleverna. L1 och L4 säger båda att programmering saknar den direkta kopplingen och att man som lärare måste hitta det matematiska.

> [...] exempelvis att man istället för programmering hade satt krav på att kunna använda grafiska hjälpmedel, exempelvis Geogebra, där det finns en direkt koppling till matematiken som man inte visuellt kan visa på samma sätt. - L4

Både L1 och L4 pratar om en direkt koppling, där de syftar till att man i geogebra direkt får svar från programmet genom att bara ändra på parametrar, medan inom programmering måste du först lägga ner tid och kraft på att få en fungerande kod innan du kan se något. En tydlig skillnad från exempelvis L5 blir att båda pratar om det som att man ska titta på något, exempelvis tabeller eller grafer, där L1 nämner “se samband” och “visuellt visa”. L4 tillägger även att hen använt det för beräkning av riemannsummor men att det kan vara avsaknaden av förståelse för programmering som gör att hen inte vet hur det ska användas på ett bra sätt. L5 ser istället programmering som ett sätt för eleverna att arbeta med matematiken och skapa förståelse genom att testa sig fram. Hen pratar om elevers tro om att matematiken alltid är något exakt och missar den mer realistiska matematiken.

> Att förstå det här med att pröva sig fram till en lösning och att förstå vad en numerisk lösning är och att det blir så himla nära ett exakt värde med upprepade beräkningar. Eleverna ser matematiken som något exakt för vi arbetar bara med algebraiska lösningar, ett exakt svar och ett exakt värde. Hela tanken med numerisk matematik och att approximera något väldigt bra, det lämpar sig med programmering och det blir nog väldigt bra om man lyckas få eleverna att förstå det. - L5

L6 använder sig lite av båda, med fokus på att få en bättre förståelse för matematiken. Hen lyfter att fokus med programmering alltid handlar om att bygga en algoritm med små steg som sedan ger en helhet, där såväl de små stegen som helheten skapar en ökad förståelse för matematiska fenomen. Utifrån "små steg som ger en helhet" ser L6 många användningsområden för programmering, hen nämner exempelvis ränteberäkningar, differentialekvationer, integraler och flera andra. Samtidigt säger L6 också att det inte är lämpligt att programmera inom allt i matematiken, utan att vissa delar är bättre än andra.

> Exempelvis integraler där man ska räkna ut arean av en liten stapel, och sedan ändrar x-värdet så att det blir en ny höjd och så vidare. Just det här med att lösa stora problem med många små steg. Beräkning av integraler blir väldigt abstrakt om man inte använder programmering, för här kan man se hur det fylls på av olika stora areor. I ren matematisk teori är det bara massa symboler som blir en slutsumma efter en viss procedur. -L6

4.3.1.3 Programmeringsfenomen

Den tredje kategorin under temat "förståelse av programmering" har fått namnet "programmeringsfenomen". Denna kategori innefattar hur respondenterna resonerade kring begrepp, förståelse eller aktiviteter som kan relateras till programmering. Temat uppkom genom att det fanns en skillnad mellan lärare i hur de pratade om programmering. Här är inte fokus på programmeringens användningsområde utan istället hur de uttrycker sig när de pratar om aktiviteter som innefattar programmering.

L1 och L2 var de som använde minst programmeringsbegrepp och uttryckte sig på ett väldigt vardagligt sätt. Exempel på vardagliga formuleringar var "att koda i olika delar", "långa stycken med instruktioner" och "att skriva kod med siffror, bokstäver och tecken". Dessa hade lärarna istället kunna beskriva med programmeringsbegrepp som "sekvenser", "algoritmer" och "kommandon". Däremot visar lärarna ändå att de förstår att just dessa delar ingår i programmering. Till skillnad från dem så använde L3 fler begrepp som "kommando", "program" och "loopar". Däremot uttrycker både L2 och L3, till skillnad från L1, att de saknar förståelse för "grundtänket" i programmering och menar att de hade haft svårt att titta igenom en färdig kod i syfte att felsöka eller bara förstå vad den gör. L1 menar däremot att hen har koll på grunderna och förstår logiken i programmeringsspråk och därmed skulle kunna testa sig fram för att förstå en algoritm. L4 är delvis inne på samma spår som L1 angående grundförståelse, samtidigt som hen menar att hen endast har grundförståelse för ett språk.

> Jag är tvungen att läsa på inför programmeringsundervisningen. Jag kan inte dra en programmeringslektion bara sådär som jag kan med matematiken. - L4

Denna saknade grundförståelse, menar L4, gör att hen måste söka sig till hur en skriver algoritmer och även att det skulle vara helt omöjligt att sätta sig med ett annat programmeringsspråk. L4 har också en tendens till att använda sig av vardagligt språk. Dock visar det ändå att hen har en förståelse av att de olika fenomen hen beskriver används inom programmering. Samtidigt använder L4 också en hel del begrepp som anses vara programmeringsbegrepp, exempelvis "villkor", "data" och "simuleringar". L5 använder liknande begrepp när hen ska beskriva liknande fenomen som L4, exempelvis "loopar", "villkor", "felsökning" och "data". Däremot vidareutvecklar hen ofta begreppen ett steg längre. Exempel på detta är när hen delar in loopar i for- och while-loopar och förklarar hur de används i olika situationer och använder begreppet iterationer när hen förklarar hur man kan se utvecklingen av en händelse steg för steg. L5 menar även att hen har en grundläggande förståelse

programmering, vilket synliggjordes när hen pratade om skillnaden mellan två programmeringsspråk.

> Själva programmeringstänket hade jag, bara att jag inte kunde omsätta eftersom jag inte hade jobbat med Pascal (ett programmeringsspråk) på väldigt länge. Språken skiljer sig men grunden är ju samma, behövde bara en uppfräschning i syntax. - L5

L6 är den som använder flest programmeringsbegrepp, bland annat olika looper, villkor, data och datainsamling, iterativt och felsökning. L6 pratar om två olika delar av programmeringen i matematiken som är viktiga. Den ena är att kunna använda syntax för att skapa algoritmer som gör det en vill och den andra mer som ett logiskt tänkande.

> Sen tänker jag att man såklart måste kunna använda olika delar av programmeringen för att kunna göra algoritmer och även kunna felsöka och förbättra de, vilket inte har med själva matematikkunskaperna att göra. Det andra är att man ska kunna tänka logiskt, lite likt det man gör i matematiken, med att kunna använda sina tidigare eller andras lösningsmetoder för att lösa sina problem. [...] exempelvis ifrågasätta "varför blir svaret alltid blir större än jag förväntar mig", är det fel i algoritmen eller gör den inte det jag tror den gör. - L6

4.3.2 Relation till programmering

Det sista temat som tagits fram i den tematiska analysen är *relation till programmering*. Lärarna i studien hade lite olika bakgrunder och därmed lite olika erfarenheter inom programmering. Dessa visade sig också skilja mellan de olika grupperna av lärare, vilket resulterade i två underkategorier, *Uppdaterad utbildning* och *Egna intresset.*

4.3.2.1 Uppdaterad utbildning

En av de tydligaste skillnaderna mellan lärarna som kände sig förberedda och de som inte gjorde det var utbildningen. Både L2 och L3 har ett vagt minne om att de möjligen haft programmering under lärarutbildningen, men var inte säkra på detta. L2 hade inte varit med på någon utbildning alls sedan dess, medan L3 hade varit på en utbildning av Skolverket inom programmeringsspråket scratch, vilket hen inte tyckte var speciellt givande för gymnasiematematiken. L4 kommer ihåg att hen sysslat med programmering under lärarutbildningen, men kommer inte ihåg något från den då det var länge sedan. L4 menar att hen definitivt inte var redo för programmering när förslaget lanserades, däremot har L4 gått en kurs som anordnades av skolan som hen jobbar på, vilket gjort att hen delvis känner sig redo. Även L5 har gått en kurs anordnad av skolan hen jobbar på.

> Jag gick en kurs anordnad av skolan året innan det infördes. Innan jag hade läst kursen och kunde behärska Python (ett programmeringsspråk) kände jag definitivt att jag inte var tillräckligt förberedd. Jag hade med min en bra grund från civilingenjörsprogrammet, men det var ett tag sedan så jag behövde en uppfräschning. - L5

L1 har inte gått någon extern kurs utan bara haft sin utbildning inom programmering på lärarutbildningen. Däremot tog L1 lärarexamen för ett år sedan och anser därmed att de hen lärt sig från den är tillräcklig för att känna sig någorlunda förberedd för programmering.

> Vi hade programmering i flera kurser vilket jag känner var tillräckligt. Dessutom var den uppdaterad utifrån vad som står i ämnesplanen så man fick med sig det man behöver. - L1

L6 är den som har betydligt mest utbildning inom programmering, då hen är legitimerad lärare i programmering. L6 avslutade den utbildningen för ungefär ett år sedan.

4.3.2.2 Egna intresset

Ytterligare en skillnad mellan lärarna som kände sig förberedda och de som inte kände sig förberedda var hur stort intresse de hade för programmering. L6 som genomgått kurser på universitet värda 95 hp har uppenbarligen ett intresse för programmering, vilket hen också konstaterar då hen sysslat med programmering sedan slutet av högstadiet. Även L5 förklarade att hen hade ett visst intresse för programmering, där hen förklarar att hen efter avslutad utbildningen som skolan anordnat återfick ett intresse.

> På en hemsida som de gav fanns det säg 100 uppgifter som blev som ett spel. Jag fastnade direkt, likt ett pussel. Man fick ett uppdrag att skriva en sekvens och sen kom nya uppgifter som delvis byggde på tidigare uppgifter. Jag tyckte det var rätt kul, blev nästan som en hobby som jag sysslade med utanför jobbet. - L5

L3 och L4 anser båda att de har ett lite intresse för programmering men inget de sysslar med på fritiden.

> Jag skulle säga att det är ungefär som bowling, har inget emot det men sysslar inte med det vanligtvis. Jag sympatiserar med det men inte mer än så. - L4

L2 säger att hen inte har något intresse alls, medan L3 skulle säga att hen definitivt har ett intresse det för att kunna använda det i sin undervisning, men inte mer än så. L3 menar att hen tror det krävs ett eget intresse för att man ska kunna bli tillräckligt bra på det.

> Men min kollega som är väldigt duktig jobbar mycket med det hemma, mer som en hobby. Det är ju dem som lägger ner tid utanför arbetstid som blir bra på det. - L3

5 Diskussion

Diskussionen innefattar reflektioner utifrån studiens metod, resultat och framtida forskningsområden. Diskussionen inleds med en resultatdiskussion utifrån frågeställningarna för studien i relation till bakgrunden, tidigare forskning, teoretiska ramverken och resultatet. Därefter diskuteras metoden där datainsamling, urval och tematiska analysen diskuteras utifrån studiens syfte. Sedan presenteras lite olika didaktiska konsekvenser, dels för mig, dels för lärare generellt. Avslutningsvis diskuteras möjliga områden som forskare bör fortsätta studera inom för den fortsatta utvecklingen av programmering.

5.1 Resultatdiskussion

Resultatdiskussionen kommer utgå från frågeställningarna för studien och diskutera dessa i förhållande till bakgrund, tidigare forskning, teoretiska ramverken och resultatet. I resultatet presenterades fem olika teman, varav första tre svarade på första frågeställningen och de andra två på den andra frågeställningen. I resultatdiskussionen kommer även tredje frågeställningen att behandlas, där målet är att reflektera över hur lärares datalogiska tänkande skiljer sig beroende på hur förberedda de känner sig för att undervisa i programmering.

5.1.1 På vilka grunder anser matematiklärare att de inte är tillräckligt förberedda för att undervisa i programmering inom matematik?

I resultatet presenterades tre olika problemområden i vilka matematiklärare känner att de inte är tillräckligt förberedda inom för att undervisa i programmering. Dessa kategoriserades i tre olika teman, otydliga ämnesplaner, didaktiska aspekter och tidsbrist i matematiken. Målet med studien var att lokalisera specifika områden som aktiva lärare anser vara problematiska för att man skulle kunna ta till åtgärder. Precis som det beskrivs i bakgrunden så har tidigare forskning oftast fokuserat på lärarstudenten, där Sands, Yadav och Good (2018) menar att det måste göras studier på aktiva lärare för att vi ska få en bättre förståelse för deras svårigheter.

Att ämnesplanerna anses vara otydliga för lärare kommer troligen inte som en överraskning, speciellt när det kommer till programmering. I bakgrunden kan läsaren läsa ett citat utdraget från Skolverket, där de kombinerat begreppet programmering, och dess breda innebörd, med det ännu bredare och luddiga begreppet datalogiskt tänkande (Skolverket, 2016). Dessa två sammanslagna begrepp skapar därmed en stor förvirring hos matematiklärarna, där flera av lärarna i denna studien anser att det är just detta som gör att de känner sig oförberedda för att använda programmering i sin undervisning. Både de som ansåg sig vara förberedda och de som inte gjorde tyckte att Skolverkets formuleringar var för svåra att tyda, där flera lärare tolkat helt olika, allt från att inte behöva koda till att en ska styra robotar. Dessa svårigheter leder därmed till att flera av lärarna känner sig otillräckliga då de redan i grunden har en ganska låg förståelse för programmering. Detta resultat stämmer även överens med det som Nouri m.fl. (2020) konstaterat i deras studie. Värt att lyfta är att flera lärare menar att trots att de kanske inte känner sig helt säkra på sina programmeringskunskaper är det främst otydligheten i ämnesplanen som är problemet. Detta problem kanske får ett slut redan inför höstterminen 2021 då revideringen med förtydligande om hur programmering ska användas ska införas.

De didaktiska aspekterna var något som de lärarna som kände sig mer förberedda för undervisning i programmering lyfte. Det som lärarna främst påpekade var svårigheterna med att få fokus riktat mot matematiken. Precis som Bråting m.fl. (2021) konstaterat tidigare hamnar fokus oftast inte på utveckling av förståelsen för ett matematiskt fenomen, utan eleverna fastnar

istället i exempelvis hur ett visst syntax används. Det som var intressant med resultatet för det didaktiska aspekterna var att de lärare som knappt pratade om didaktiska svårigheter var de lärare som också inte ansåg sig vara tillräckligt förberedda. De menade istället att det skulle räcka med att vara utbildad pedagog. Även en av de mer förberedda lärarna uttryckte liknande, däremot lyfte hen flera svårigheter som kan klassas som didaktiska svårigheter. Bråting m.fl. (2021) diskuterade hur den bristande förmågan inom programmering påverkar den kreativa förmågan i undervisningen, vilket de menar påverkar huruvida läraren tillsammans med eleverna når lärandemålen eller inte. Liknande presenteras i denna studie, där en av lärarna menar att bristen på det didaktiska färdigheterna påverkar lärandet vilket i sig gör att hen har svårare att komma åt lärandemålen. Även L6, som är läraren med mest programmeringserfarenhet, säger att lärare förmodligen behöver en kurs i exempelvis datalogiskt tänkande, men att detta är ett problem för senare. Sands m.fl. (2018) och Kilhamn m.fl. (2021) uttrycker liknande, då oftast när forskare gjort liknande studier använder de antingen helt eller delvis lärare som har relativt goda förmågor inom programmering. Troligen leder till att man lämnar en skara av lärare med sämre programmeringskunskaper utanför dessa studier, som förmodligen är i minst lika stort behov av att få ämnesdidaktiska kunskaper inom programmering.

Den tredje aspekten som lärarna lyfter är tidsaspekten, där de känner att programmering oftast blir ett tidskrävande projekt som ibland kräver flera lektioner för att få en fungerande algoritm. Däremot har L6 som använder programmering regelbundet inte detta problem längre, där hen oftast använder sig av algoritmer som är relativt korta och endast använder ett fåtal syntax med sina elever. Samtidigt konstaterade L6 också att skolan beslutat att lägga in 20 timmar programmering första läsåret för ettorna, vilket gör att totala timmarna i skolan ökar. Detta är givetvis inget den enskilda läraren kan göra utan måste beslutas på högre nivå. Skillnaden menar jag dock delvis beror på hur förberedda eleverna är, men det beror också till stor del av hur läraren förstår hur uppgifter kan lösas med programmering. Precis som L6 säger bör inte programmering användas till allt utan det finns vissa saker som passar bättre. Dessutom tror jag att många lärare inte vet hur de ska göra, utan de tänker att exempelvis en algoritm för att beräkna integraler består av 50 rader kod och då tänker att det är omöjligt att hinna med i matematiken. Återigen kanske den reviderade ämnesplanen dels kommer tydliggöra vad som ska ingå och dels även innehålla lite lärarmaterial på hur lärare ska kunna, med korta algoritmer, beräkna relativt utmanande uppgifter.

5.1.2 Vad skiljer lärare som känner sig förberedda och lärare som inte känner sig förberedda?

På förhand kan denna fråga kännas väldigt bred och svår att besvara, då det givetvis är väldigt mycket som skiljer lärarna. De olika kategorierna som tagits fram har i resultatet motsvarat de skillnader som för just denna studie gick att finna och som visade något typ av mönster. De två teman som analysen resulterade i är förståelse för programmering och relation till programmering. Dessa innehöll lite olika delar vilket kommer diskuteras nedan.

Det första som lyftes i resultatet var programmeringens användningsområden. Resultatet visade att flera av lärarna inte verkade ha en förståelse för att programmering inte endast används inom naturvetenskapliga studier, där lärarna menade att elever som inte läser natur och teknik aldrig kommer komma i kontakt med programmering. Troligen menar lärarna att de som inte blir civilingenjörer kommer aldrig programmera en app eller en robot. Här saknas en grundläggande förståelse för programmeringens möjligheter, eftersom både ekonomer och samhällsvetare kan i framtiden behöva använda sig av programmering för att simulera olika händelser eller bara

bilda en förståelse för hur olika produkter fungerar. Det som också blev tydligt i resultatet var hur lärarna såg på programmering. Tittar man exempelvis på L3 och L6s sätt att förklara programmering skiljer det sig markant, där L3 beskriver det som att be datorn göra något och L6 breddar begreppet från mjukvaruutveckling till matematiska beräkningar. Här ser man tydligt hur oförberedda lärare inte riktigt har förståelse för programmeringens möjligheter i stort. Även i Kilhamn m.fl. (2021) studie visade att lärarna främst fokuserar på programmering som "ett bra verktyg", "skapar intresse" och "hjälpa lärandet" men nämner inte hur det kan påverka förståelsen av programmeringens roll i samhället. Detta synliggörs ytterligare när lärarna ska förklara hur programmering kan användas som ett verktyg i matematiken. De lärare som kände sig minst förberedda kunde knappt uttala sig om vad det kan användas till i matematiken. Misfeldt m.fl. (2019) konstaterade också att det finns en del lärare som inte ser sambandet. Däremot är skillnaden här att samtliga lärare förstod att det fanns en koppling, de visste bara inte exakt vad den kopplingen är, vilket mer stämmer överens med Szabo och Helenius resultat (2019).

Flera av de som delvis kände sig förberedda motiverade programmering som ett sätt att lösa det de inte själv kan lösa, exempelvis att göra tusentals återupprepningar, stora datainsamlingar och att approximera något som inte går att göra algebraiskt. Fördelningen mellan vilka som hade förståelse för detta vilka som inte hade följde inte ett lika tydligt mönster för varje gruppering av lärare, däremot syntes en tydlig skillnad mellan de som kände sig minst förberedda och de som kände sig mest förberedda. Ytterligare ett problem som lyfts, även av de som delvis kände sig förberedda, är den saknade "direkta kopplingen" till matematiken. Här skiljer sig de som delvis är förberedda och de som känner sig helt förberedda, där de som är delvis förberedda pratar om att "se samband" och "visuellt visa", vilket kanske oftast inte är meningen med programmering. De som däremot kände sig helt förberedda för programmering menar istället att programmering används för att utveckla en förståelse, exempelvis hur en med looper kan se hur arean under en integral blir större ju fler staplar som tas med. Till skillnad från Misfeldt m.fl. (2019) studie som visade att en del lärare anser att programmering borde implementeras i ett annat ämne istället, var det ingen i denna studie som uttryckte sig så. Däremot uttalade sig en av lärarna som inte känner sig förberedd att hen inte förstår varför de skulle behöva koda program i matematiken, men förstår att användandet av färdiga program kunde vara aktuellt.

En av de tydligaste skillnaderna mellan personerna i hur förberedd de kände sig var deras tidigare relation till programmering, där resultatet delades in i två delar. Den ena delen som verkar ha ett tydligt samband med om lärarna kände sig förberedda var om de hade genomgått en relevant utbildning den senaste tiden. Här fungerade de olika kategorierna nästan exakt utefter hur uppdaterad och omfattande deras utbildning hade varit. Det kommer förmodligen inte som en överraskning att en person efter avslutad utbildning kan mer än någon som inte utbildats, däremot visar resultatet att även mindre omfattande utbildningar kan ge lärarna tillräckligt för att de ska känna att de delvis är förberedda. Angeli och Jaipal-Jamani (2018) konstaterade i sin studie att man med ganska små insatser kan uppnå en relativt bra nivå av förståelse hos lärarstudenter. Resultatet för denna studie visar att detta även verkar gälla aktiva lärare, även om det är svårt att dra generella slutsatser utifrån en såpass liten studie.

Studien visar även att det fanns en korrelation i hur lärarna uppfattade olika programmeringsfenomen och hur förberedda de känner sig för att undervisa i programmering. Resultatet presenterar bland annat olika skillnader i begreppsanvändning och grundförståelse för programmering. Resultatet visar att lärare som kände sig mer förberedda använde sig i mindre utsträckning av vardagliga uttryck när de pratade om programmering, till skillnad från lärare som inte kände sig förberedda som istället använde mycket av vardagliga formuleringar.

Detta kan vara ett tecken på att lärarna inte heller kan tillräckligt mycket programmering, även om de inte själva anser att det är ett problem. Enligt Bråting m.fl. (2021) krävs det goda kunskaper i programmering för att lärare ska kunna motivera och engagera elever till att nyttja programmering som ett verktyg i matematiken. Lärarna som anser att de inte är tillräckligt förberedda antyder även att de saknar en grundförståelse för programmering, där exempelvis L4 säger att hen skulle varit chanslös i ett annat programmeringsspråk än hen använder idag.

Slutligen kunde man även se att lärarnas intresse för programmering verkade ha en betydande roll. De lärare som ansåg att programmering var mer än "okej" och faktiskt kunde tänka sig syssla med det utanför arbetstid var de som också kände sig mest förberedda. Det som däremot är ett negativt resultat är att en av lärarna menade att det i princip krävs att de arbetar med programmering utanför sin arbetstid för att kunna bli tillräckligt bra. Om detta stämmer betyder det att skolor potentiellt skulle behöva dra ner på antalet lektionstimmar för att lärarna ska få tid att lära sig programmera, vilket givetvis inte är optimalt utifrån skolans perspektiv. Däremot kan man fråga sig om det faktiskt är sant. Precis som L6 säger är det många lärare som ser programmering som att kunna koda och styra en robot, vilket medför att lärare ser programmering som något gigantiskt och därmed också är alldeles för svår och tidskrävande för att kunna bemästra. Genom avsmalningen som Skolverket gjort med den nya revideringen ser jag en möjlighet i att lärare kommer få en tydligare bild av hur det ska användas, och därmed också kan lära sig tillräckligt för att kunna undervisa i det.

5.1.3 Hur skiljer sig det datalogiska tänkandet mellan lärare som känner sig förberedda och de som inte känner sig förberedda?

I detta avsnitt är tanken att undersöka hur lärarnas datalogiska tänkande skiljer sig beroende på om de känner sig förberedda eller ej. Detta kommer endast göras utifrån hur de uttalat sig i intervjuerna och hur det kan förstås. Lye och Koh (2014) menar att det krävs ett datalogiskt tänkande för att kunna lära ut programmering, vilket därför blir intressant att se om lärarna i denna studien är införstådda i det datalogiska tänkandet. För att undersöka detta kommer de två teoretiska ramverken som presenterats i bakgrunden att användas. För att undersöka detta utgår diskussionen från resultatet i avsnitt 4.3.1.3.

Lärarna som ansågs vara minst förberedda, alltså L2 och L3, uttryckte få begrepp som ingår i Brennan och Resnicks ramverk (2012). De begrepp som de få gånger antingen nämner eller pratar vardagligt om är sekvenser, loopar och operatorer. Det är även svårt att avgöra om de förstår begreppen eller mest har hört om de.. Båda menar också att de saknar den grundläggande förståelsen för programmering, vilket kan tolkas som det som ingår i den första nivån i PGK hierarkin. Därmed anses de varken ha förståelse för någon av de datalogiska dimensionerna och uppnår inte ens den lägsta förståelsenivån i PGK hierarkin.

L1 och L4 som båda kände sig delvis förberedda skiljer sig en del och kommer därför analyseras var för sig. Utifrån det datalogiska tänkandet så påminner L1 väldigt mycket om de två som ansåg att de inte var tillräckligt förberedda. Hen använder en hel del vardagliga begrepp för att beskriva olika situationer, samtidigt som hen visar förståelse för sekvenser och algoritmer. Hen pratar om felsökning och att testa sig fram, men påstår även att hen hade haft svårt att felsöka en kod utan att få sätta sig in i det ordentligt. L1 påstår att hen har en grundläggande förståelse, vilket blir svårt att placera in i de tre dimensionerna. Utifrån hur hen uttryckt sig visar L1 inte att hen har tillräcklig god förståelse för att kunna klassas inom någon av de tre dimensionerna för datalogiskt tänkande. Hen passar delvis in i den lägsta nivån av PGK hierarkin men uppfyller

samtidigt inte alla kriterier då en även bör kunna felsöka och förstå att något blir fel i sin kodning. L4 däremot visar fler förmågor och förståelser för de datalogiska koncepten där hen dels använder begrepp som data och villkor men även visar förståelse för att hen behärskar begreppen sekvenser och loopar. L4 visar även delvis förståelse för begreppet iterativ då hen pratar om att arbeta med upprepningar men uppfyller inte ens majoriteten av begreppen som ingår i det datalogiska utövandet. L4 menar att hen har en grundläggande förståelse för algoritmer och programmering och anser att hen delvis behärskar ett programmeringsspråk. Dessa egenskaper gör att hen uppnår nivå två i PGK hierarkin och därmed visar på någorlunda kunskaper inom datalogiskt tänkande.

L5 visar på en bredare begreppsförmåga där hen både använder och förstår även begreppen iterationer, villkor, loopar, datainsamling. Hen lyfter även att elever ska testa sig fram och försöka förstå de små delarna stegvis. Hen visar på tydlig förståelse av både det som ingår i första och andra dimensionerna. L5 ger däremot inga intryck att hen befinner sig i den tredje dimensioner, datalogiska perspektiv. L5 förklarar att hen hade enkelt för att lära sig ett nytt programmeringsspråk på grund av liknande tänk. Det är svårt att veta om detta beror på att språken liknade varandra eller om hen helt enkelt inte är beroende av en viss programmeringsmiljö. L5 ger intrycket av att befinna sig på nivå 3 i PGK hierarkin men det är samtidigt svårt att konstatera.

När det kommer till de tre dimensionerna av datalogiskt tänkande uppnår L6 samtliga. Hen visar bra förståelse för de flesta datalogiska koncepten och ger uttryck för förståelse av flera av de datalogiska utövanden, exempelvis att vara inkrementell, iterativ, testa, återanvända och att abstrahera. L6 lyfter även vikten av att vara ifrågasättande, dels till sin algoritm men även till hur något matematisk är uppbyggt. L6 är även inne på att elever ska använda “tidigare och andras lösningar”, vilket ger intryck av att hen även har en förståelse för samarbetet mellan personer. När det kommer till PGK är L6 svårare att placera in, då man egentligen hade behövt utföra ett test för att ta reda på vad hen kan. L6 ger intrycket av att hen befinner sig på den högsta nivån eftersom hen har en väldigt bred kunskapsbas och kan anpassa sig utefter uppgift. L6 säger även att koderna som hen lär eleverna oftast bara är några rader trots relativt avancerad matematik, vilket innebär att hen har förmågan att smala ner algoritmerna tills att endast nödvändigheter finns kvar.

Utifrån ovan kan en anta att L1, L2 och L3 inte har tillräckligt god förståelse av det datalogiska tänkandet, L4 har en grundläggande förståelse, L5 en bättre förståelse och L6 besitter en god förståelse för det datalogiska tänkandet.

5.1.4 Sammanfattande resultatdiskussion

Utifrån resultatet i denna studie kan man se att det finns många aspekter som är utmanande för matematiklärare när det kommer till programmering. Målet med studien var att ta reda på vad lärare känner att de saknar och även lokalisera vilka skillnader det finns mellan de som känner att de inte är tillräckligt förberedda och de som känner att de är det. Genom den tematiska analysen har olika teman presenterats och därefter diskuterats i denna resultatdiskussion. Bland annat konstateras det att lärarna i denna studie saknar tydlighet från Skolverket i hur de ska undervisa och även vad som kan förväntas av eleverna. Detta är förhoppningsvis något som kommer lösa sig i och med den nya revideringen av ämnesplanen. Lärarna lyfter även didaktiska svårigheter, där det egentligen behöver tillsättas utbildningar i hur lärare ska undervisa i programmering. Detta tydliggörs även mot slutet av resultatdiskussionen där det visades ett möjligt mönster mellan hur förberedd lärare känner sig och hur bra förståelse de har för det datalogiska tänkandet. Lärarna i denna studie som kände sig mest osäkra var även de

som hade lägst förståelse för det datalogiska tänkandet. Resultatet i denna studie stämmer överens med Nouri m.fl. (2020) som menar att lärare saknar flera delar av det datalogiska tänkandet, där de menar att lärarna bland annat saknar tillräcklig utbildning. Även denna studie visar att det finns ett samband mellan dessa faktorer.

5.2 Metoddiskussion

Syftet med studien var att ta reda på vilka delar som lärarna uppfattade saknades för att de skulle känna sig tillräckligt förberedda för att undervisa i programmering. Kvalitativa studier används för att studera olika fenomen utifrån olika personers egna uppfattningar, vilket gjorde att det blev ett naturligt val av datainsamlingsmetod. Under studiens gång övergick fokuset från lärarnas uppfattning till att faktiskt hitta vad som skiljer lärare som är förberedda och inte förberedda. Med facit i hand hade detta även kunnat göras genom en kvantitativ studie, där fokus istället hade varit att undersöka skillnaderna.

Intervjuguiden anpassades för att undersöka lärarnas uppfattning om deras situation och hur de uppfattar programmering i stort. För att göra detta användes en semistrukturerad intervju med drag åt det mer strukturerade. Grunden var en semistrukturerad intervjuguide, där det lämnades ett stort utrymme för följdfrågor utifrån lärarens svar. Detta gjorde att intervjuerna nådde ett djup som fick lärarna att fundera kring deras situation och hur de upplever den. Samtidigt fanns det en tydlig struktur i grundteman vilket gjorde att samtliga respondenter på något sätt fick uttala sig i ungefär samma frågor, vilket underlättade när analysarbetet gjordes.

Personerna som medverkade i studien försökte jag att välja utifrån syftet att ha en stor variation mellan deltagarna som möjligt när det kommer till ålder, program de undervisa på, kön, år som lärare och så vidare. Däremot handplockades vissa delvis eftersom studien även gick ut på att jämföra mellan de som kände sig förberedda och de som inte kände sig förberedda. Detta kan givetvis ha påverkat resultatet, då antalet personer i varje kategori var väldigt låg. Det beror såklart även på att studiens omfattning är liten, vilket gör att det blir svårt att dra generella slutsatser om större populationer. Dessutom tackade två personer nej till intervju, varav ena först svarade att hen inte var duktig på programmering. Med tanke på lärares osäkerhet inom programmering kan det även vara så att de som valde att vara med hade en lite högre kunskapsnivå.

De teoretiska ramverken valdes utifrån att undersöka lärares förståelse för det datalogiska tänkandet och hur denna skiljer sig mellan lärare som känner sig olika förberedda för att undervisa i programmering. De tre dimensionerna av datalogiskt tänkande är en bra teori för att kapsla in ett så stort begrepp samtidigt som de visar på olika sidor av det datalogiska tänkandet. Till skillnad från de tre dimensionerna av datalogiskt tänkande är PGK hierarkin är en teori som bättre anpassar sig för att ta reda på hur mycket av det datalogiska tänkandet som de kan omsätta när de programmerar. Utifrån syftet var PGK en bra teori för detta, däremot befann sig lärarna på en lägre nivå än förväntat vilket gjorde att flera av lärarna till och med hamnade utanför teorin. Dessutom var det flera delar i resultatet som inte hade med de teoretiska ramverken att göra, exempelvis otydlig ämnesplan. Möjligen är bristen på förståelse av det datalogiska tänkandet ursprungsproblemet till många av delproblemen som hittats, samtidigt kan inte denna slutsats tas utifrån denna studie. Precis som det beskrivs i början av detta avsnitt övergick fokus från lärares uppfattning till skillnader mellan lärare under studiens gång.

Som analysmetod användes tematisk analys, där stegen i analysprocessen gjordes enligt de Braun och Clarke (2006) beskrivit. Bristen av erfarenhet påverkas givetvis analysen, vilket

gjorde att arbetet krävde mer tid än förväntat. För att undvika eventuella missar följdes stegen noggrant och omarbetades flera gånger för att minska på felmarginalen.

5.3 Didaktiska konsekvenser

Studien visar att det fortfarande finns en sanning i att matematiklärare inte känner sig tillräckligt förberedda för att undervisa i programmering. Studien visar även att det finns många fler svårigheter än "att kunna programmera". En del av svårigheterna är av sådan karaktär att lärare själva inte kan påverka, exempelvis otydlig ämnesplan och brist på tid i matematiken. Utöver dessa framkommer exempelvis didaktiska svårigheter, där en tydlig svårighet för lärare var att få lektionerna att fokusera på matematiken framför själva programmeringen. Många lärare såg problematiken att eleverna skulle behöva sitta upp mot tio lektioner för att förstå förstå vad all syntax gör och sedan få själva algoritmen att fungera. Till skillnad från dessa konstaterade den mest förberedda läraren att hen sällan använder mer än loopar och variabler, där vissa algoritmer kan vara fyra till fem rader långa. Det lärare kan ta med sig från detta är att använda programmering när det är lämpligt, där de reviderade ämnesplanerna kommer vara till stor hjälp i detta förtydligande. Samtidigt är det även upp till lärare att lägga lite tid på att förstå hur de exempelvis kan skriva en algoritm för att beräkna integraler.

En svårighet som är betydligt mer komplext är lärares förståelse för programmering och datalogiskt tänkande. För att lärare ska kunna utveckla detta måste man i princip ha specifika utbildningar riktade mot gymnasielärare i matematik. Samtidigt visar studien att relativt små insatser kan göra skillnad, där skolor exempelvis hade kunnat klämma in fortbildningskurser på något av loven eller liknande.

För min egna del har denna studie gett mig en bredare förståelse för undervisningen av programmering. Många av de vetenskapliga artiklar som lästs har gett mig kunskap och förståelse för datalogiskt tänkande och hur detta kan omsättas i min undervisning. Även intervjuerna, i synnerhet den som var med den mest erfaren inom programmering, gav mig inspiration till flera användningsområden med idéer på hur jag kan använda mig av programmering i min undervisning. Studien visar även att jag som nyexaminerad förmodligen kommer med en viss fördel gentemot många andra lärare, vilket skapar en trygghet i min framtida yrkesroll.

5.4 Fortsatt forskning

En viktig del av resultatet i denna studie grunda sig i att lärarna ansåg att styrdokumenten var alldeles för otydliga och att det skulle underlätta med en mer precis förklaring av hur lärare ska använda sig av programmering i undervisningen. Eftersom revideringen av ämnesplanen som träder i kraft i sommar kommer att tydliggöra vad programmering ska användas till är det av intresse för forskning att undersöka hur det går för lärare framöver. Samtidigt visar studien även att lärarna saknar vissa andra delar och att det finns en del som skiljer de lärare som är förberedda och de som inte är. Bland annat verkar en del av lärarna sakna en viss förståelse för programmering och för det datalogiska tänkandet. Utifrån Lye och Kohs (2014) uttalande om att det krävs datalogiskt tänkande för att kunna lära ut och förstå programmering bör detta vara en av de första sakerna som man fokuserar på att utveckla hos lärarna. Studien visar utifrån de teoretiska ramverken att lärarna uppnår olika dimensioner och olika nivåer av förståelse för det datalogiska tänkandet. Denna studien visar främst möjliga mönster då studien gjorts i sån liten omfattning. Det hade varit intressant att se om dessa mönster stämmer överens även hos större grupper. Dels hade det varit intressant att se hur mycket förståelse matematiklärare i Sverige

har för datalogiskt tänkande och dels göra en studie av vad som faktiskt är att kunna "tillräckligt" mycket. I denna studie anger lärarna själva om de är tillräckligt förberedda eller inte. Det hade varit av intresse för forskning att bilda en uppfattning av hur mycket lärarna behöver förstå av det datalogiska tänkandet för att kunna använda programmering i matematiken på ett gynnsamt sätt.

Referenslista

Angeli, C., & Jaipal-Jamani, K. (2018). Preparing pre-service teachers to promote computational thinking in school classrooms. I *Khine M. (Red.). Computational thinking in the STEM disciplines*, s.127–150. Springer: Cham.

Aho, A. A. (2012). Computation and computational thinking. *The Computer Journal*, *55*(7), s.832–835.

Barr, V., & Stephenson, C. (2011). Bringing computational thinking to K-12: what is Involved and what is the role of the computer science education community?. *Acm Inroads*, *2*(1), s.48–54.

Blooms taxonomi. (2020, maj 5). *Wikipedia*, . Hämtad 09.06, april 27, 2021 från //sv.wikipedia.org/w/index.php?title=Blooms_taxonomi&oldid=47603856.

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

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. I *Annual American Educational Research Association meeting*. Vancouver, Kanada.

Bråting, K., Kilhamn, C., & Rolandsson, L. (2021). Integrating programming in Swedish school mathematics: Description of a research project. I Y. Liljekvist, L. Björklund Boistrup, J. Häggström, L. Mattsson, O. Olande, H. Palmér (Eds.), *Sustainable mathematics education in a digitalized world. Proceedings of MADIF12. SMDF*. s.101–110.

Butterfield, A., Ngondi, G., & Kerr, A. (2016). *Programming*. A Dictionary of Computer Science.

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

Denning, P. J. (2009). Beyond computational thinking. *Communications of the ACM*, *52*(6), s.28–30. Gerestrand, Anders. (2017). *Programmering Som Ett Verktyg För Lärande - Lärares Uppfattningar Om Programmeringens Bidrag till Andra ämnen; Programming as a Learning Tool - Teachers' View of the Contribution of Programming to Other Subjects* (Magisteruppsats). Göteborg: Institutionen för tillämpad informationsteknik, Göteborgs universitet. Hämtad från https://gupea.ub.gu.se/handle/2077/53469

Grover, S., & Pea, R. (2013). Computational thinking in K–12: A review of the state of the field. *Educational researcher*, *42*(1), s.38–43.

Heintz, F., Mannila, L., & Färnqvist, T. (2016). A review of models for introducing computational thinking, computer science and computing in K-12 education. I *Frontiers in Education Conference (FIE) 2016*, s.1–9.

Helenius, O. & Misfledt, M. (2018). *Programmering och matematisk modellering*. Hämtad från Skolverket: https://larportalen.skolverket.se/#/modul/0-digitalisering/Gymnasieskola/448_matematikundervisningmeddigitalaverktygII_GY

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

Khine, M. S. (Ed.) (2018). *Computational Thinking in the STEM Disciplines - Foundations and Research Highlights.* Cham: Springer Nature Switzerland.

Kilhamn, C., & Bråting, K. (2019). Algebraic thinking in the shadow of programming. *Proceedings of the Eleventh Congress of the European Society for Research in Mathematics Education*,s.566–573.

Kilhamn, C., Bråting, K., & Rolandsson, L.. (2021). Teachers' Arguments for including Programming in Mathematics Education. *Bringing Nordic Mathematics Education Into The Future. Papers From NORMA 20. Preceedings Of The Ninth Nordic Conference On Mathematics Education, 2021.* s.169-176.

Lee, I., Martin, F., Denner, J., Coulter, B., Allan, W., Erickson, J., Malyn-Smith, J., & Werner, L. (2011). Computational thinking for youth in practice. *ACM Inroads, 2*(1), s.33–37.

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? Computers in Human Behavior, 41, s.51-61. Hämtad från https://www.sciencedirect.com/science/article/abs/pii/S0747563214004634

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

Mannila, L., Dagiene, V., Demo, B., Grgurina, N., Mirolo, C., Rolandsson, L., & Settle, A. (2014). Computational thinking in K-9 education. I A. Clear & R. Lister (Red.), *Procedings of working group reports of the 2014 on innovation & Technology in Computer Science Education Conference*, s.1-29. Hämtad från https://www.researchgate.net/publication/273772180_Computational_Thinking_in_K-9_Education

Misfeldt, M., Szabo, A., & Helenius, O. (2019). Surveying teachers' conception of programming as a mathematical topic following the implementation of a new mathematics curriculum. I U. Jankvist, M. Van den Heuvel-Panhuizen, & M. Veldhuis, M. (Eds.). *Proceedings of CERME11* (s.2713–2720). Utrecht: Freudenthal Institute, Utrecht University and ERME

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Enquiry, 11*(1), s.1-17.

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

Perrenet, J., Groote, J. F., & Kaasenbrood, E. (2005). Exploring students' understanding of the concept of the algorithm: Levels of abstraction. *ACM SIGCSE Bulletin, 37*(3), s.64–68 ACM

Programmering. (2020, december 30). Wikipedia, . Hämtad 09.47, april 20, 2021 från //sv.wikipedia.org/w/index.php?title=Programmering&oldid=48655181.

Rolandsson, L., & Skogh, I. (2014). Programming in School: Look Back to Move Forward. *ACM Transactions on Computing Education (TOCE), 14*(2), s.1-25.

Rolandsson, L. (2015). *Programmed or Not : A study about programming teachers' beliefs and intentions in relation to curriculum*. Hämtad från http://urn.kb.se/resolve?urn=urn:nbn:se:kth:diva-160724

Sands, P., Yadav, A., & Good, J. (2018). Computational thinking in K-12: In-service teacher perceptions of computational thinking. I *Khine M. (Red.). Computational thinking in the STEM disciplines*, s.151–164. Cham: Springer.

Selby, C. C. (2015, November). Relationships: computational thinking, pedagogy of programming, and Bloom's Taxonomy. I *Proceedings of the workshop in primary and secondary computing education* s.80-87.

Skolverket. (1994). *1994 år läroplan för de frivilliga skolformerna, Lpf 94: Särskilda programmål för gymnasieskolans nationella program; Kursplaner i kärnämnen för gymnasieskolan och den gymnasiala vuxenutbildningen.* Hämtad från: https://gupea.ub.gu.se/handle/2077/30998

Skolverket. (2000). *Naturvetenskapsprogrammet : programmål, kursplaner, betygskriterier och kommentarer*. Hämtad från: https://gupea.ub.gu.se/handle/2077/30942

Skolverket. (2011). *Läroplan för gymnasieskolan 2011*. Hämtad från: https://gupea.ub.gu.se/handle/2077/30836

Skolverket. (2016). *Redovisning av uppdraget om att föreslå nationella it-strategier för skolväsendet – förändringar i läroplaner, kursplaner, ämnesplaner och examensmål*. Hämtad 29 april, 2021, från https://www.skolverket.se/publikationsserier/regeringsuppdrag/2016/uppdrag-om-nationella-it-strategier-for-skolvasendet

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

Skolverket. (2021a). *Ämnesplan för matematik på gymnasiet 2011: reviderad: 2021*. Hämtad från: https://www.skolverket.se/undervisning/gymnasieskolan/aktuella-forandringar-pa-gymnasial-niva/forandringar-pa-gymnasial-niva

Skolverket. (2021b). *Kommentarmaterial till ämnesplanen i matematik 2011: reviderad: 2021*. Hämtad från: https://www.skolverket.se/publikationer?id=7841

Skolöverstyrelsen. (1981). *Läroplan för gymnasieskolan. Supplement, 75, Matematik för treårig naturvetenskaplig linje och fyraårig teknisk linje*. Hämtad från: https://gupea.ub.gu.se/handle/2077/31154

Vetenskapsrådet. (2002). *Forskningsetiska principer inom humanistisk- samhällsvetenskaplig forskning*. Hämtad från https://www.vr.se/analys/rapporter/vara-rapporter/2002-01-08-forskningsetiska-principer-inom-humanistisk-samhallsvetenskaplig-forskning.html

Westerberg, Josefine. (2018). *Programmering – Ett Nytt Redskap I Den Pedagogiska Verktygslådan.; Programming – A New Utility for the Pedagogical Toolbox* (Magisteruppsats). Göteborg: Institutionen för didaktik och pedagogisk profession, Göteborgs universitet. Hämtad från https://gupea.ub.gu.se/handle/2077/54908

Wing, J. M. (2006). Viewpoint. Computational thinking. *Communications of the ACM*, *49*(3), s.33-35

Bilaga 1 – Intervjuguide

Introduktion
Kort presentation av mig

Presentera studien

Förklara och få godkänt om inspelning och användning (raderas efter avslutad studie)

Grundläggande fakta
Ålder

Ämneskombination

Program person undervisar på

Utbildning
    När avsluta du den?

Hur länge har du arbetat som lärare

Kunskaper om programmering

Erfarenhet
- Ingick programmering i din lärarutbildning?
- Har du gått annan kurs inom programmering? (Vart, när, nivå, utsträckning)
- Vart skulle du säga att du fått din största del av dina programmeringskunskaper

Har du ett intresse för programmering?

Kan du förklara vad programmering är. (få de att utveckla)

- Hur kan det användas inom matematik?

Lärarens undervisning

I en studie på svenska matematiklärare visade det sig att lärarna generellt inte kände sig tillräckligt förberedda för att undervisa i programmering inom ämnet matematik. Håller du med om detta påstående utifrån din egna situation?

Om JA, förklara varför:
- Kunskaper i att programmera, vilka?
- Förståelsen av bryggan mellan matematik och programmering?
- Annat?

Hur påverkar det ditt lärande?
Har du försökt göra något åt att du inte känner dig förbered? I så fall vad?

Om NEJ, förklara vad du känner dig förberedd inför.
- Har du haft känslan tidigare?
  Om JA:
  - Vad har du gjort för att bli förberedd?
  - Hur påverkar det dig som lärare?

  Om NEJ:
  - Varför tror du att du inte haft den?
  - Vad tror du krävs för att andra ska känna som dig?

Har du hållit i lektioner med programmering?
Om Ja:

- Vilken form av programmering?
- Hur ofta förekommer det?
- Vilka aspekter vill du komma åt?
- Vilka svårigheter finns det för dig som lärare utifrån syftet med att utveckla dessa aspekter?

Om Nej:

- Varför?
- Skulle du vilja ha lektioner i programmering?
- Vilka aspekter skulle du då vilja komma åt?

Övergripande om programmering
Vad tror du är främsta anledningen till att matematiklärare inte känner sig förberedda på undervisning i programmering?

Varför tror du att detta är den främsta anledningen?

Vilka är de viktigaste egenskaperna en behöver för att kunna undervisa i programmering?

Har du något du funderar på som du skulle vilja tillägga?

Bilaga 2 – Mejl från Skolverket

Hej!

Tack för dina frågor om programmering i ämnesplanen i matematik.

Det finns flera olika definitioner eller avgränsningar för vad programmering är eller innebär, även bland de som arbetar med eller forskar på programmering. Inte heller Skolverket har någon fastslagen definition av programmering, men när det gäller matematik på gymnasial nivå är det användbart att åtminstone tänka att ”programmering” omfattar att kunna göra beräkningar i upprepade sekvenser/loopar och med villkor. I kommentarmaterialet för 2021 års ämnesplan står det bland annat så här:

”Anledningen till att programmering nämns i ämnesplanen är framför allt de möjligheter som programmering ger för att genomföra stora mängder beräkningar. Detta gör det möjligt att utforska och använda matematik på fler sätt än vad som är möjligt för hand eller med enklare digitala verktyg och färdiga applikationer.”
Med utgångspunkt i det bör ämnesplanen tolkas som att eleverna ska få möjlighet att inte bara använda färdiga program, där parametrar ändras, utan också att få se de tankesätt och metoder som gör programmering användbar i matematik. Det är svårt att dra en gräns för hur långt in i dessa tankesätt och metoder man bör eller behöver gå. Att endast ge elever ett färdigt program för att (exempelvis) beräkna värden på integraler uppfyller knappast avsikten med programmering i det centrala innehållet. Att visa elever på en begreppsmässig nivå – utan kod – hur ett sådant program fungerar, och låta dem själva använda programmet, ligger närmare avsikten. Att ge elever möjlighet att se och utforska programkod (som formler i kalkylblad eller mer konventionell kod) ligger ännu närmare avsikten, och att låta elever själva skapa eller bearbeta sådan kod ännu närmare (särskilt i kurs 3 och senare).
Att låta elever själva skapa eller bearbeta kod kräver dock att elever har ganska god teknikvana för att det ska vara meningsfullt, och i många lägen saknas ännu den vanan. Att lägga den tid som krävs för att få programmeringsvana är sällan möjligt i matematikkurserna, och läraren kommer att behöva prioritera det mot annat innehåll i kursen (precis som med många andra saker).
Programmering bör tolkas som att inte endast använda färdiga applikationer, utan att eleverna också ska få möjlighet att se hur principerna i programmering kan användas för att arbeta med matematiska frågeställningar. Det går däremot inte att säga hur djupt detta ”bör” behandlas, utan det måste balanseras utifrån de förutsättningar som den specifika läraren har med den specifika elevgruppen.
Du kan läsa mer i kommentarsmaterialet till ämnesplanen i matematik. Se länk nedan.

Länk:
Kommentarmaterial till ämnesplanen i matematik: www.skolverket.se/publikationer?id=7841