GÖTEBORGS UNIVERSITET

Användningen av programmering för att lära ut ett matematiskt innehåll i gymnasieskolan

Jeremia Mörling

LAU690

Handledare: Samuel Bengmark
Sanela Mehanovic

Examinator: Tommy Gustafsson

Rapportnummer:

GÖTEBORGS UNIVERSITET

Abstract

Examensarbete inom lärarutbildningen

Titel: Användningen av programmering för att lära ut ett matematiskt innehåll på gymnasieskolan

Författare: Jeremia Mörling

Termin och år: HT 2011

Kursansvarig institution: Sociologiska institutionen

Handledare: Samuel Bengmark och Sanela Mehanovic

Examinator: Tommy Gustafsson

Rapportnummer:

Nyckelord: Matematikundervisning, Programmeringsundervisningen, Undervisning, Ämnesintegration, Matematik, Programmering

Sammanfattning:
Syftet med studien var att sammanställa exempel på hur programmering används i gymnasieskolan i Sverige för att lära ut ett matematiskt innehåll, samt att undersöka lärarnas syfte med, och de upplevda effekterna av, denna undervisningsmetod. Målet var även att själva den här rapporten skulle kunna användas som ett konkret underlag för lärare som skulle vilja använda sig av programmering för att lära ut ett matematiskt innehåll. Studiens empiri bygger på djupintervjuer där analysenheterna valdes ut med hjälp av snöbollsurval. Sammanfattningsvis kan man säga att trots att området började utforskas redan för ca 30 sedan är det väldigt få som använder sig av programmering för att lära ut ett matematiskt innehåll. En av orsakerna är troligen att trots en del positiva forskningsresultat så har de tänkta fördelarna inte otvetydigt kunna bevisas. Dessutom råder det troligen en kunskapsbrist på området. Det finns helt enkelt inte så många lärare som är kunniga inom både programmering och matematik. De lärare som ändå har kunskapen och viljan verkar uppleva tidsbrist som det största hindret. Man upplever sig inte ha tid med att experimentera och förbereda nytänkande lektioner. Trots alla hinder finner jag fortfarande området intressant för två grupper av elever. För de svaga elever som inte finner motivation till att lära sig matematik genom ”normal” undervisning kan programmering, framför allt spelprogrammering, erbjuda ett intressant alternativ som kan vinna deras intresse. För de starka elever som upplever den normala matematikundervisningen för statisk kan finna utmaning i skapandet och programmerandet av matematiska algoritmer för att lösa problem med hjälp av en dator (eller miniräknare).

Illustration 1: Karta över städerna där de intervjuade personernas skolor ligger. (Nationalencyklopedin, 2011)

Innehållsförteckning

1 Inledning

”Hur kan jag få mina elever motiverade?”. ”Hur ska jag på bästa sätt lära ut det här innehållet”. Det är två frågor som jag i olika form har hört från många lärare. Kanske ställs dessa frågor ännu mer på sin spets när det gäller undervisning i matematik. Skolverket visade i en rapport att majoriteten av elever tyckte att matematik är roligt fram till ca år 4-5. Sedan sker den stora uppdelningen mellan de som tycker matematik är roligt och är duktiga på det och de som inte tycker matematik är roligt och inte är duktiga på det (Skolverket, 2003, s 19). Högstadielärare och gymnasielärare inom matematik möter alltså vanligen en stor andel elever som inte tycker att matematik är roligt och som inte har tilltro till sin egen förmåga i matematik. ”Hur kan jag få mina elever motiverade?”. ”Hur ska jag på bästa sätt lära ut det här innehållet?”. Denna studie undersöker användandet av programmering för att lära ut ett matematiskt innehåll. Studien undersöker hur man arbetar med denna typ av ämnesintegration i Sverige, med vilka syften samt vilka resultat man har observerat som en effekt av detta arbetssätt. Är en sådan ämnesintegration ett tänkbart alternativ för att vinna tillbaka intresset för de elever som inte längre känner motivation för matematikämnet?

Den andra aspekten ligger i en dynamisk utmanande undervisning för de starka eleverna. När jag studerade en datavetenskaplig kurs inom algoritmer upptäckte jag att jag egentligen tyckt att större delen av den matematik jag läst tidigare kändes väldigt statisk. Kursen algoritmer öppnade för mig upp en typ av matematisk problemlösning där intuition, diskussion och undersökande matematik var centrala. Det kändes nästan som att jag för första gången under mina matematiska studier utmanades till att tänka egna tankar istället för att bara lära mig att förstå det som andra tänkt. Används denna typ av algoritmskapande inom matematikundervisningen på gymnasiet i Sverige? Hur tycker lärarna att det fungerar?

När jag läser skolverkets rapport ”Lusten att lära – med fokus på matematik” tycker jag att den talar på ett sätt som stämmer överens med det de intervjuade lärarna i denna rapport kommunicerar om att använda programmering för att lära ut ett matematiskt innehåll. Nedan följer ett par citat från skolverkets rapport.

> *”De undervisningssituationer, där vi har mött många engagerade och intresserade elever som har givit uttryck för lust att lära har, i sammandrag, kännetecknats av att det finns utrymme för både känsla och tanke, upptäckarglädje, engagemang och aktivitet hos både elever och lärare. Dessa undervisningssituationer har kännetecknats av variation i innehåll och arbetsformer. […] Det har funnits inslag av laborativt, undersökande arbetssätt.” (Skolverket, 2003, ss 14–15)*
>
> *”Matematik behöver ha någonting med livet utanför skolan att göra.” (Skolverket, 2003, s 30)*
>
> *”Variation, flexibilitet och att undvika det monotona i undervisningen är viktigt för lusten att lära. Formen för inlärning behöver växla för att tillgodose elevers olika sätt att lära.” (Skolverket, 2003, s 30)*
>
> *”Många elevers uppfattning är att det blir roligare i alla ämnen om de får möjlighet att påverka sina studier, både innehåll och redovisningsformer. De faktiska möjligheterna att påverka undervisningen i matematik under senare skolår förefaller dock generellt vara små” (Skolverket, 2003, s 31)*

”Elevers självtillit och lust att lära skulle otvivelaktigt stärkas om också något av all den kunskap de producerar kom till användning på ett konstruktivt sätt. Det gäller också för de utvärderingsformer som används och den återkoppling som ges. Att man som elev får tillfällen att visa vad man lärt sig” (Skolverket, 2003, s 33)

2 Syfte och frågeställning

2.1 Syfte

Syftet med studien var att sammanställa exempel på hur programmering används i gymnasieskolan i Sverige för att lära ut ett matematiskt innehåll, samt att undersöka lärarnas syfte med, och de upplevda effekterna av, denna undervisningsmetod. Målet var även att själva den här rapporten skulle kunna användas som ett konkret underlag för lärare som skulle vilja använda sig av programmering för att lära ut ett matematiskt innehåll. Jag hade alltså en ambition av att försöka knyta insamlat material till specifika gymnasiekurser i den mån jag fann det möjligt.

2.2 Frågeställning

Frågeställningarna som ligger till grund för denna studie var följande:

- På vilka sätt använder man programmering för att lära ut ett matematiskt innehåll i gymnasieskolan i Sverige idag?
- Vilka syften anser sig lärarna ha för denna undervisningsmetod?
- Vilka resultat av denna undervisningsmetod har lärarna observerat?

Jag var inte ute efter att undersöka hur snittundervisningen inom matematik ser ut, utan snarare att finna de lärare som faktiskt använder sig av programmering inom sin matematikundervisning. Jag var i första hand intresserad av att hitta exempel på där programmering användes inom matematikkurser, men då jag fann mycket få exempel på det, utvidgades området för studien till att även innefatta programmeringskurser där läraren har ett tydligt mål att eleverna ska lära sig ett matematiskt innehåll.

3 Relaterad forskning och relaterat utvecklingsarbete

3.1 En historisk tillbakablick

> *”There are revolutionary changes afoot in education, in its contents as well as its methods. Widespread computer access by schools is at the heart of these changes. Throughout the world, but particularly in the U.S.A., educators are using computers for learning activities across the curriculum, even designing their own software. But virtually all educators are as anxious and uncertain about these changes and the directions to take as they are optimistic about their ultimate effects. “Now that this admittedly powerful symbolic device is in our schools,” they ask, “what should we do with it?”” (Pea & Kurland, 1984, s 137)*

Citatet ovan är 27 år gammalt. Redan då var man alltså övertygad om att datorn skulle revolutionera utbildningen.

3.1.1 De tänkta fördelarna

Man tänkte sig att: ”Computers can make the abstract concrete and personal as they help children learn more effectively by making their thinking processes conscious” (Clements & Gullo, 1984, s 1051; Papert, 1980) och ”The computer programming environment holds the promise of being an effective device for cognitive process instruction—teaching how, rather than what, to think” (Clements & Gullo, 1984, s 1051; Lochhead & Clement, 1979). Här är en utav de längsta listorna på kognitiva förmågor som ett resultat av programmering:

> *”(1) rigorous thinking, precise expression, recognized need to make assumptions explicit (since computers run specific algorithms);*
>
> *(2) understanding of general concepts such as formal procedure, variable, function, and transformation (since these are used in programming);*
>
> *(3) greater facility with the art of “heuristics”, explicit approaches to problems useful for solving problems in any domain, such as planning, finding a related problem, solving the problem by decomposing it into parts, etc. (since “programming provides highly motivated models for the principle heuristic concepts”);*
>
> *(4) the general idea that “debugging” of errors is a “constructive and plannable activity” applicable to any kind of problem solving (since it is so integral to the interactive nature of the task of getting programs to run as intended);*
>
> *(5) the general idea that one can invent small procedures as building blocks for gradually constructing solutions to large problems (since programs composed of procedures are encouraged in programming);*
>
> *(6) generally enhanced “self-consciousness and literacy about the process of solving problems” (due to the practice of discussing the process of problem solving in programming by means of the language of programming concepts*);*

> *(7) enhanced recognition for domains beyond programming that there is rarely a single "best" way to do something, but different ways that have comparative costs and benefits with respect to specific goals (learning the distinction between "process" and "product", as in Werner, 1937)." (Feurzeig, Horwitz, & Nickerson, 1981; Pea & Kurland, 1984, s 143)*

Dessa tänkta fördelar var nog för många:

> *"several million pre-college age children in the U.S.A. are already receiving instruction in computer programming each year, and France has recently made programming compulsory in their precollege curriculum, on a par with mathematics and native language studies." (Pea & Kurland, 1984, ss 138–139)*

3.1.2 Studier för att belägga fördelarna

För att belägga argument för att använda programmering in undervisningen genomfördes ett antal experimentella studier, bl.a. av Douglas Clements. Clements genomförde flera experimentella studier på sex- och åttaåringar, med en grupp som fick lära sig programmering i LOGO[1] och olika typer av kontrollgrupper. De barn som hade arbetat med LOGO-programmering hade efter experimenten signifikant bättre resultat på testen för kreativt tänkande, förstå/analysera sitt eget tänkande och ange riktningar (Clements, 1986, ss 315–317; Clements & Gullo, 1984, ss 1056–1057). När det gäller de metakognitiva testerna (testerna för över det egna tänkandet) verkar det som att LOGO-programmeringen framför allt påverkade vissa metakognitiva delar. En av dem var att förstå ett problems natur och göra sig en mental bild av problemet (Clements, 1986, s 316; Clements & Gullo, 1984, s 1057). En annan var att förstå när man inte förstod. Clements trodde att det hade att göra med det metakognitiva tänkandet som upplevs i samband med felsökning av buggar (Clements, 1986, s 316; Clements & Gullo, 1984, s 1056).

Utifrån övriga test menade Clements och Gullo dock ändå att det inte fanns bevis för att LOGO-programmering som komplement till "normal" matematikundervisning påverkar den generella kognitiva utvecklingen eller prestationerna i matematik (Clements, 1986, s 317; Clements & Gullo, 1984, s 1057). När det gäller annan programmering än LOGO-programmering har Clements dock sett negativa effekter på prestationerna i matematik (Clements, 1985). Clements och Gullo sa också att det krävdes omfattande datorträning för att kunna genomföra experimentet (Clements & Gullo, 1984, s 1057).

3.1.3 Svenska studier kring LOGO-programmering på 80-talet

Redan på 70-talet började datorer att introduceras i den svenska skolan genom DIS-projektet (Datorn I Skolan). Syftet var att studera "de pedagogiska konsekvenserna av datoriseringen för skolan, det vill säga påverkan på undervisningens innehåll, organisation och metodik samt fortbildning och läromedel" (Lindh, 1993, s 71). På mitten av 80-talet genomförde Rolf Hedrén sedan en studie i två delar om programmering, i programmeringsspråket LOGO, skulle kunna användas inom matematikundervisningen på mellanstadiet. Angående resultatet av studien sa han själv att:

> *"Det är svårt att dra några säkra slutsatser av försöket, så länge utvärderingen endast bygger på resultaten i pilotförsöket, där det endast fanns en försöksklass*

1 LOGO är ett språk där programmerar en sköldpadda till att gå omkring på skärmen och rita en figur. Se http://sonic.net/~nbs/webturtle/

> *och ingen jämförelseklass. Det enda som går att få fram är antydningar om i vilken riktning man kan tänka sig att söka svaren" (Hedrén, 1988, s 26)*

Med detta i åtanke kan man titta på de resultat han ändå såg.

> *"Först och främst fanns en hög korrelation mellan de båda testen i programmering och testet på talbegrepp och problemlösning […]. Detta torde peka på att det finns ett starkt samband mellan programmeringsförmåga och säkerhet på talbegreppet och i problemlösning." (Hedrén, 1988, s 26)*

Han fann också att intresset för programmering var svårt att hålla uppe under så lång tid som projektet varade (två år). Han trodde att en stor anledning var att programmeringen blev för svår för många elever. Resultatet var för övrigt starkt polariserat. På ena sidan fann pojkarna, som i försöksklassen presterade bättre än flickorna på matematiktesten, och som "vid försökets slut kom att uppfatta matematik som både lättare och roligare än vid dess början." (Hedrén, 1988, s 27) På andra sidan fanns flickorna, som i försöksklassen presterade sämre än pojkarna på matematiktesten, och där resultatet var det motsatta.

3.1.4 Tidig kritik

Många var alltså mycket positiva till programmering i början på 80-talet och det hade börjat komma en del positiva resultat från studier. Roy Pea och D. Midian Kurland gav sig därför i kast med att undersöka vilka positiva effekter som programmering medför som det egentligen fanns bevis för. De var mycket kritiska. Det första problemet de pekade på var att de menade att de dittills genomförda studierna var designade för att titta på utvecklandet av de högre nivåerna av kognitivt tänkande som tänktes komma av en hög nivå av färdighet inom programmering. Programmeringsnivån hos de elever som deltog i dessa studier var emellertid låg, p.g.a. de endast hann arbeta med programmering under en kortare tid. För övrigt menade de att det fann inga bevis för att programmering främjar matematisk korrekthet eller att de hjälper eleverna att utforska matematiken. De ifrågasätter även om de problemlösningsförmågor som eleverna lär sig inom programmeringen kan appliceras även utanför programmeringen. Resultaten kring detta område var blandade (Pea & Kurland, 1984, ss 158–159).

Även 13 år senare menade Pedersen att det var svårt att bekräfta de positiva effekterna av LOGO-programmering (Pedersen, 1997, s 23).

3.2 Modern forskning och moderna utvecklingsprojekt

3.2.1 Ett nutida svenskt projekt

2008 genomförde två lärare vid Angeredsgymnasiet, Sofia Bäckström och Marie Rudenstam ett projekt finansierat av .SEs internetfond för att se om det var möjligt att införa webbprogrammering som en del av matematikundervisningen. De menade att den typ av matematikundervisning som bedrivs på Sveriges gymnasieskolor var för ensidig och inte speciellt rolig för många elever (Bäckström & Rudenstam, 2010, s 1).

> *För stenålderns barn var stenar ett sätt att lära sig hantera matematiska begrepp som antal. De kunde med sina händer flytta på stenarna och se vad som hände. För dagens elever är programmering av dataspel samma sak, menar lärarna på Angeredsgymnasiet. Eleverna kan modellera och experimentera till exempel med*

> *andragradskurvor och får på så sätt en förståelse för begreppen. (Näslundh, 2008)*

I Deweys anda menade de att argumentet att eleverna behöver kunskaperna för universitetet inte är tillräckligt för dagens ungdomar. Istället såg de en möjlighet ”att knyta an till många ungdomars stora intresse - och även ofta stora - kunskaper om Internet och media” (Bäckström & Rudenstam, 2010, s 2).

Som en del av projektet designade de material för webbprogrammering menat att användas inom kursen Matematik A[2]. Deras långsiktiga mål var att ta fram material som skulle kunna ”användas för att träna upp de matematiska begrepp som enligt läroplanen ingår i gymnasiekurserna Matematik A , Matematik B och Matematik C” (Bäckström & Rudenstam, 2010, s 1).

Projektet fick viss uppmärksamhet, både nationellt och internationellt (Bäckström & Rudenstam, 2010, s 3). Tre år senare har jag dock under datainsamlingen till min studie inte träffat på en enda person som använt sig av det framtagna materialet. Bäckström och Rudenstam har dessutom själva hittills (18 nov 2011) endast använt materialet inom kursen Programmering A (L5, 2011a).

3.2.2 Internationellt

The Joint Mathematical Council of the UK arbetar i Storbritannien för att främja matematisk framgång och förbättra undervisning i matematik. I en rapport från i år skriver de att:

> *”We consider that mathematical programming should be a staple part of the mathematics courses of the future. Just as a calculator has to be on hand if arithmetic is to be taught to best advantage, so a computer will be needed if algebra, geometry, statistics and other branches of mathematics are to be taught to the best advantage” (Joint Mathematical Council of the United Kingdom, 2011, s 24)*

De tänker sig då programmering i den vida betydelsen att ge instruktioner till en dator för ett bestämt syfte, alltså inte nödvändigtvis att programmera ett program i dess vanliga betydelse (Joint Mathematical Council of the United Kingdom, 2011, s 25).

En grupp datavetare, matematiker, mm från ett antal universitet, mm, i USA som vill få in mer matematik inom datavetenskap-kurserna har slagit sig samman i en grupp som kallar sig ”Integrating Mathematical Reasoning into Computer Science Curricula”[3]. Kirby Urner[4], en utav de aktiva inom den gruppen , pratar mycket om att använda programmering inom matematikundervisningen. I sin självbiografi säger han ”I became convinced that high school mathematics should include a larger programming component, making it more technologically relevant.” (Urner, 2011a). Han var även med och försökte starta en ”high school”-matematikkurs i Oregon, USA, med mer fokus på datavetenskap och viss programmering. Målet var att ”rädda” ungdomar som håller på att hoppa av utbildningen eller i största allmänhet är i ”riskzonen” (Urner, 2010). Detta verkar dock ha gått i stöpet (Urner,

2 http://angeredswebben.se/webbbok/version5/

3 ”Mathematical reasoning is central to computer science. It should therefore be an integral part of the entire CS curriculum [...] We are a group of computer scientists, mathematicians, and others interested in fostering such change.” (Integrating mathematical thinking in computing curricula, 2011)
Webbsida: http://www.math-in-cs.org/

4 http://www.grunch.net/4dsolutions/kirby.html

2011b). Han har även föreläst om hur man kan använda programmeringsspråket Python i matematikundervisningen (Urner, 2008).

3.2.3 Vad kommer att hända nu?

Kairos Future[5] tog nyligen temperaturen på temperaturen på IT-användning och digital kompetens i den svenska skolan. Med en studie med stort underlag[6] kom de fram till följande:

> *"• **IT kommer att revolutionera skolan**. Svensk skola har inlett en resa som bara börjat, och utvecklingen går snabbt.*
>
> *• **Det finns en klyfta mellan ledning och klassrum**, där förvaltningschefer, IT-strateger och rektorer är betydligt mer entusiastiska inför att använda IT i undervisningen jämfört med lärare och elever.*
>
> *• **Lärare och rektorer är i stort behov av kompetensutveckling**. 4 av 10 lärare och elever tycker inte att lärarna har tillräckligt bra IT-kompetens för att använda IT i undervisningen på ett bra sätt. Centrala framgångsfaktorer för att IT ska förbättra elevernas prestationer är att lärarna tycker det är kul att använda datorer, att de vågar experimentera och att IT används för att eleverna effektivare lär sig baskunskaper." (Kairos Future, 2011b, s 3)*

Liksom man gjorde på 80-talet tror nu även Kairos Future (och alla lärare och elever de frågat) att IT kommer att revolutionera skolan. En majoritet bland alla grupper (rektorer, lärare, elever, föräldrar och IT-chefer) tycker att IT borde användas mer i skolan, men entusiasmen bland rektorer och IT-chefer är mycket större än övriga grupper (Kairos Future, 2011b, s 7).

Det som här talar för programmering som en del av matematikundervisningen är att de tror att IT kommer att revolutionera skolan och att ett utav framgångsfaktorerna är att våga experimentera, där programmering kan vara en bra plattform. En sak som talar emot är att var tredje elev inte tycker att IT borde användas mer i utbildningen (Kairos Future, 2011b, s 7). En annan sak som talar emot är att om man nu trodde att IT skulle revolutionera skolan redan för knappt 30 år sedan (Pea & Kurland, 1984, s 137) och det enligt Kairos future inte redan hänt (Kairos Future, 2011b, s 6), vad är det då som talar för att det kommer att hända nu[7]?

I rapporten "The Opportunity Equation" talar man om att USA dramatiskt måste öka alla studerandes kunskaper inom STEM (Science, Technology, Engineering and Mathematics) för att klara den hårda internationella kunskapsbaserade konkurrensen. Liknande Kairos future menar de att en dramatisk förändring i skolans upplägg behövs:

> *"The world has shifted dramatically – and an equally dramatic shift is needed in educational expectations and the design of schooling to provide our students with the STEM knowledge and skills that are crucial to virtually every endeavor of individual and community life." (The Opportunity Equation, 2009)*

5 "Kairos Future är ett internationellt forsknings- och konsultföretag som hjälper företag att förstå och forma sin framtid." (Kairos Future, 2011a)
Webbsida: http://www.kairosfuture.com/

6 "...genom en litteraturstudie, enkätundersökning med över 4000 respondenter, fokusgrupper och ett expertseminarium." (Kairos Future, 2011b, s 3). Genom att läsa rapporten ser man att respondenterna består av (åtminstone) följande grupper: IT-strateger/förvaltningschefer, rektorer, lärare, elever och föräldrar.

7 Se kap 3.4 för en fortsättning på denna tanke, samt kap 6.5 för en diskussion kring varför det inte hänt mer på området.

Det som här talar för programmering inom matematikundervisningen är det starka fokuset på STEM, där programmering har en stark anknytning till hela STEM-begreppet, och att man talar om STEM som en enhet. Det som talar emot programmering inom matematikundervisningen är att även om man talar starkt för att vi måste ändra sättet att ha skola på så pratar man inte alls om ämnesintegration eller om just programmering som någon del av lösningen.

3.3 Matematikens roll för datavetenskap

3.3.1 Datavetarna ser inte matematikens roll - ämnesintegration efterlyses

En utav forskarna på IMPEd (Improving Mathematics and Programming Education[8], Finland), Linda Mannila, efterlyser mer interaktion mellan matematik och programmering på gymnasiet. Hon pratar i sin doktorsavhandling om svårigheter med undervisning inom matematik och programmering för datavetare på högskolan. Den största orsaken till att svårigheterna kring matematikundervisningen på programmen för datavetenskap menar hon är att eleverna inte ser dess relevans. Hon menar att mycket av detta beror på hur kursplanerna är upplagda. Istället för att, inom kursplanerna, se matematik som en integrerad del av programmering ser man det som en separat del, vilket skapar det minskade intresset för matematik bland datavetarna. Mannila undersöker därför i sin rapport hur man på olika sätt kan integrera ämnena mer både inom gymnasiet och högskolan. De sätt som hon undersöker, och förespråkar är ”Strukturerade härledningar” inom matematiken, Python som första programmeringsspråk, och användandet av ”Invariant-based programming” (Mannila, 2009, s i–ii). ”Strukturerade härledningar är en praktisk teknik för att konstruera bevis som är rigorösa, men som inte kräver att den underliggande teorin är totalt axiomatiserad.” (Back, 2008, s 8). Metoden bygger på en bevismetod utvecklad för datateknik och som Back anser kan ses som en standard inom programmeringsmetodik. Tekniken har tydliga kopplingar mellan datavetenskap och matematik (Back, 2008, s 47). ”Invariant-based programming” är på liknande sätt en typ av bevismetod att använda på datorprogram, med tydliga kopplingar till logik och matematik. Metoden går lite förenklat ut på att man först specificerar ett bevis som sedan ligger till grund för själva koden (Back, 2006, s III). Python är ett programmeringsspråk med fokus på läsbarhet (Python Software Foundation, 2011). De hävdar själva att ett program i Python är 3-5 gånger kortare än motsvarande program i Java, vilket minskar utvecklingstiden. Nackdelen är långsammare exekvering (Python Software Foundation, 1997). Mannila och de andra på IMPEd försöker alltså föra matematik och programmering närmare varandra, utan att man för den skull programmerar på matematiklektionerna. De försöker snarare föra elevernas tänkande inom matematik och programmering närmare varandra.

8 http://www.imped.fi/wordpress/?lang=se
IMPEd () är ett...
”...resurscenter som ger nya idéer och metoder för undervisning av grundläggande matematik och programmering på olika utbildningsstadier. [...] Vår forskning strävar efter att förbättra förståelsen av matematik och programmering i gymnasiet och bland första årets studerande vid universitet och yrkeshögskolor.” (IMPEd, 2011)

3.3.2 Andra röster om matematikens roll för datavetenskapen

I september 2003 hade ”Communications of the ACM”[9] ett temanummer om varför studenter inom datavetenskap behöver matematik. I artikeln ”Why math” skriver de fyra författarna följande:

> *”Software solutions to most problems (e.g. banking, on-line commerce, airline reservations, etc.) consist of constructing a (mathematical) model of the real (physical) domain and implementing it in software. Mathematics can be helpful in all stages of software development: design, specification, coding, verifying security and correctness of the final implementation. In many cases, particular topics in mathematics are not as important as a high level of mathematical sophistication.” (Bruce, Drysdale, Kelemen, & Tucker, 2003, s 40)*

Detta passar mycket väl ihop med en utav förmågorna som undervisningen inom matematik ska utveckla hos eleverna enligt ämnesplanen för matematik:

> *”Undervisningen i ämnet matematik ska ge eleverna förutsättningar att utveckla förmåga att [...] tolka en realistisk situation och utforma en matematisk modell samt använda och utvärdera en modells egenskaper och begränsningar.” (Skolverket, 2011, avs Ämnets syfte)*

I en annan artikel i samma nummer skriver Henderson om matematikens roll för mjukvaruutveckling:

> *”What role does mathematics play in software engineering? Consider the following two statements. “Software practitioners do not need or use mathematics.” “Software practitioners do need to think logically and precisely.” An apparent contradiction since the reasoning underlying software engineering and mathematics are similar” (Henderson, 2003, s 45)*

3.3.3 Diskret matematik och datavetenskap

Ett par intresseorganisationer inom datavetenskap i USA samarbetar med varandra för att med jämna mellanrum ge ut riktlinjer, en typ av läroplan, för högskoleutbildning inom datavetenskap. I rapporten från 2001 ägnas ett helt subkapitel åt strategier för att integrera diskret matematik in i den introducerande läroplanen för datavetenskap (The Joint Task Force on Computing Curricula, 2001, ss 25–26).

I artikeln “Why math” (som nämndes I kapitel 3.3) lyfter författarna fram sex områden inom diskret matematik som tas upp i läroplanen ovan: ”Funktioner, relationer och mängder”, ”Grundläggande logik”, ”Bevistekniker (inklusive induktion och motsägelsebevis)”, ”Grunderna inom räkning”, ”Grafer och träd” och ”Diskret sannolikhet” (Bruce m.fl., 2003, ss 40–41).

Henderson pekar i en rapport på vikten av diskret matematik och logik för utbildning inom datavetenskap och att att koppla dessa ämnesområden till dataämnen.

> *”Discrete mathematics and logic are important foundations for the education of computer scientist and software engineers. Often this material is not introduced sufficiently early, its important connections with computer and software topics*

9 http://cacm.acm.org/

> *are not made sufficiently clear, and the material is not reinforced in computer courses." (Henderson, 2002, s 1)*

För flera artiklar inom området datavetenskap och diskret matematik, se följande sida: http://www.cs.geneseo.edu/~baldwin/math-thinking/math-thinking-body.html#pubs

3.3.4 Matematik för programmering och spelutveckling

På Högskolan på Gotland håller de en kurs med namnet "Matematik för programmering och spelutveckling"[10] med följande innehåll (Utbildnings- och forskningsnämnden vid Högskolan på Gotland, 2007):

- Trigonometri
- Koordinatsystem
- Matriser och linjära ekvationssystem.
- Vektorrum och affina rum.
- Kurvor

Trigonometri, koordinatsystem, linjära ekvationssystem, vektorer och kurvor (grafer) är centrala begrepp inom gymnasiematematikens kursplaner[11] (Skolverket, 2011). Då Högskolan på Gotland anser att dessa begrepp är centrala för programmering så är det, enligt mig, inte långt till att gissa att dessa begrepp även kan läras ut genom programmering.

3.4 Sammanfattning

Sammanfattningsvis kan man säga att rösterna spretade för 30 år sedan och de spretar fortfarande. Programmeringens roll inom matematiken är definitivt inte given. Frågan man ställer sig är: Varför skulle det hända något nu om det inte hänt något under de senaste 30 åren. Samtidigt skjuts det in mer och mer datorer i skolan och i och med den marknadisering vi har av framför allt gymnasieskolan tror jag själv att alla gymnasieelever inom de teoretiska programmen kommer att ha varsin dator inom ett par år. Detta är en stor skillnad jämfört med tidigare och det kommer troligen att tvinga lärare att använda datorn i mycket högre grad som en del av undervisningen.

10 https://portal.hgo.se/courses/course/view.php?id=1457&element=1

11 Trigonometri nämns inom Matematik 3c och är sedan ett centralt begrepp inom Matematik 4. Koordinatsystem är en grundläggande del av geometri och funktioner. Själva ordet förekommer inom Matematik 1a och 1c. Ordet "funktion" förekommer i någon form 50 gånger och ordet "graf" i någon form 26 gånger inom alla kursplanerna för Matematik tillsammans och är alltså väldigt centrala begrepp inom hela matematikämnet för gymnasiet. Linjära ekvationssystem är en del av Matematik 2a, 2b och 2c. Begreppen matriser och rum (vektorrum och affina rum) är inte en direkt del av gymnasiematematiken. Eventuellt kan man knyta an till rum via den gnutta talteori som finns inom Matematik 5 (Skolverket, 2011).

4 Metoder och tillvägagångssätt

4.1 Undersökningsenheter

Undersökningsenheter för studien var lärare som använde programmering som undervisningsmetod för att lära ut ett matematiskt innehåll.

4.2 Metod för urval: Snöbollsurval

Då både jag och min handledare delade uppfattningen att programmering inom matematik är en tämligen ovanlig förekommelse valdes metoden snöbollsurval för att hitta de få lärare som frågeställningen gällde. Snöbollsurval är lämpligt då de analysenheter man försöker hitta är ovanliga och det är troligt att om man hittar en enhet så känner denne till ytterligare enheter (Esaiasson, Gilljam, Oscarsson, & Wängnerud, 2007, s 216).

4.2.1 Hur urvalet gick till

Min handledare kände till en lärare som var lämplig för min studie, Sofia Bäckström. En annan tidig kontakt var Per Söderhjelm, på Lunds Universitet, som ansvarar för Programmeringsolympiaden[12]. Han gav mig kontaktuppgifter till samtliga lärarkontakter för årets (2011) Programmeringsolympiad (mot ett löfte att jag skulle dölja deras e-postadresser för varandra om jag gjorde ett massutskick). Jag googlade även runt lite för att se om detta kunde leda mig till några intressanta kontakter eller något intressant arbete som görs.

Det som snabbt kom att bli grundtanken i mitt sätt att söka kontakter var att söka bland lärare i programmering. De borde vara avsevärt färre och det är enligt mig även ganska troligt att dessa även är lärare inom matematik. Om en lärare använder sig av programmering inom matematik så brinner hon troligen för programmering och då är steget inte så långt till att gissa att hon även är programmeringslärare.

4.2.2 Det faktiska urvalet

Totalt kontaktades 65 lärare från 58 skolor, men 39 av lärarna (60 %) svarade inte och 2 lärare (3%) fick inte mitt e-postbrev p.g.a. tekniska problem med e-postadressen. 46 st (71%) av de kontaktade lärarna var lärare som hade elever med i programmeringsolympiaden. Av de lärare som svarade var 12 lärare (18%) intressanta för studien och det är informationen från dem som resultat-delen i denna rapport bygger på. Med ”intressanta för studien” menas att de medvetet använde programmering för att lära ut ett matematiskt innehåll, oberoende av inom vilken kurs detta gjordes. Då jag upplevde att denna kategori lärare var ganska få inkluderades även lärare som hade *långt gångna tankar* om att lära ut ett matematiskt innehåll med hjälp av programmering, i gruppen ”intressanta för studien”. Dessa lärare var tre av de tolv intressanta lärarna. Det fanns två lärare (3%) som var av visst intresse för studien, men som ändå inte är med i studien. De svarade vid en första kontakt via e-post, men kommunikationen kom aldrig så långt att jag fick data som kunde användas i denna studie. Detta berodde enligt dem själva på att de hade för ont om tid.

12 http://www.csc.kth.se/contest/ioi/

Det faktum att 60% av de kontaktade lärarna inte svarade har ingen större inverkan på denna rapports trovärdighet, då detta inte är någon statistisk undersökning. På sin höjd innebär det att jag inte fick med hela bredden av syften, arbetssätt och observerade effekter av att arbeta med programmering för att lära ut ett matematiskt innehåll. Det finns dock inte heller någon direkt anledning att tro att de lärarna som inte svarade hade någon större ämnesintegration mellan matematik och programmering. Alla utom en utav de som inte svarade kontaktades bara p.g.a. att de troligen hade kurser inom programmering på skolorna. Även om programmering och matematik har mycket med varandra att göra (se t.ex. kap 3.3 Matematikens roll för datavetenskap), tror jag inte att det skulle tillföra så mycket till denna studie om de inte medvetet arbetade med programmering med det bestämda målet att lära ut ett matematiskt innehåll. Om en kontaktad lärare faktiskt skulle använda programmering som undervisningsmetod för att lära ut ett matematiskt innehåll så är min uppfattning att läraren troligen skulle ha svarat någonting på mitt första försökt att ta kontakt, även om de inte skulle ha tid att genomföra en intervju. Men detta kan jag självklart inte veta. Tyvärr gjorde tidsbegränsningen för denna studie det praktiskt omöjligt att följa upp de kontakter som inte svarade överhuvudtaget.

Utöver de kontaktade lärarna fick jag genom snöbollsurvalet kontakt med fem andra personer, mestadels från olika universitet, som bidrog med kontakter och material.

4.3 Metoder för datainsamling

4.3.1 Samtalsintervjuer (via telefon eller på plats)

Då jag inte på förhand visste syftet med varför undersökningsenheterna hade valt att använda programmering för att lära ut ett matematiskt innehåll, eller hur denna ämnesintegration skulle kunna se ut valdes samtalsintervju som huvudsaklig metod för datainsamling. Esaiasson m.fl. har ställt upp fem tänkbara områden som de tycker passar sig för samtalsintervju. Det första området kallar de ”När vi ger oss in på ett outforskat fält” (Esaiasson m.fl., 2007, s 285). Även om forskningsfältet att använda programmering för att lära ut matematik i sig inte är outforskat, så är själva praktiken, att faktiskt använda sig av det i undervisningen, tämligen outforskat, vilket gör att denna studie passar inom området att ge sig in på ett outforskat fält. Valet av samtalsintervju som metod för datainsamling passar även väl ihop med att jag inte förväntar mig att hitta fler än en handfull undersökningsenheter (Esaiasson m.fl., 2007, kap 14). Då jag använder mig av snöbollsurval förväntar jag mig inte heller att uppnå någon större extern validitet, vilket även styrker samtalsintervju som metod för datainsamling. Esaiasson m.fl. kopplar även de ihop samtalsintervjuer och snöbollsurval som en lämplig kombination (Esaiasson m.fl., 2007, s 291). Som grund för intervjuerna utgick jag från de frågor som kan ses i Bilaga A.

Alla intervjuer spelades in samtidigt som jag förde anteckningar, med ett undantag. Intervjun med den anonymiserade läraren L5 spelades inte in p.g.a. att jag glömde bort det. Dock fördes anteckningar under denna intervju.

Konversation via e-post användes för att klargöra eventuella otydligheter eller luckor.

4.3.2 Intervju via e-postbrev (en variant av samtalsintervju)

Även om målet för studien var att genomföra djupintervjuer med analysenheterna var detta inte alltid möjligt. En orsak var att vissa analysenheter upplevde att tiden var för knapp och att

de därför hade svårt att boka in en specifik tid. De kunde då välja att svara skriftligt på de frågor jag hade till grund för samtalsintervjuerna.

En annan orsak till att normala samtalsintervjuer inte alltid var möjligt var att analysenheterna inte kände sig tillräckligt ”färdigtänkta” för att känna att de skulle kunna genomföra en vanlig intervju. Om jag ändå upplevde att deras tankar och erfarenheter var av intresse för studien bad jag dem då vanligen att bara skriva ner sina tankar som de var i ett e-postbrev. Jag upplevde att det var bättre att ta del av deras data på något sätt än inget sätt alls. Frågor och svar för eventuella följdfrågor sköttes i dessa fall också via e-post.

4.3.3 Källhantering

För att hantera alla källor till rapporten användes referenshanteringsprogrammet Zotero[13]. Anteckningarna från intervjuerna skannades till pdf och lades tillsammans med inspelningarna in i Zotero. Alla e-postbrev som refereras i denna rapport exporterades till pdf och lades in i referenshanteringsprogrammet. Alla övriga texter som fanns tillgängliga elektroniskt lades även de in i referenshanteringsprogrammet. Självklart hanterades även de källor som inte var elektroniska med hjälp av Zotero, även om jag i dessa fall inte kunde lägga in själva dokumenten i Zotero.

4.4 Ontologi och epistemologi

Esaiasson m.fl. gör två grundantaganden gällande ontologi och epistemologi, som de menar präglar större delen av dagens empiriska samhällsvetenskap och som även ligger till grund för denna studie:

> *”dels ett ontologiskt antagande om att det finns en verklighet som är oberoende av våra subjektiva medvetanden; dels ett epistemologiskt antagande om att det genom systematiska observationer går att erhålla välgrundad kunskap om denna verklighet” (Esaiasson m.fl., 2007, s 17)*

Då min data består av lärares upplevelse av att använda programmering som undervisningsmetod för att lära ut ett matematiskt innehåll kan man säga att min ontologi hamnar nära en livsvärldsontologi. Det är alltså den upplevda verkligheten som är av intresse och jag vet inte på förhand vilka aspekter som bygger upp denna upplevda verklighet. Jag utgår istället från att verkligheten är komplex och mångfacetterad och låter analysenheterna själva definiera vad de anser är viktigt (Claesson, 2009, ss 57–71). Denna ontologi stämmer, vilket jag antydde i första meningen i detta stycke, väl överens med den metod för datainsamling som jag valt (Claesson, 2009, s 74).

Om min epistemologi när det gäller denna studie tror jag inte mer behöver sägas än något liknande det som Esaiasson säger, d.v.s. att jag antar att jag genom samtal med undersökningsenheterna kan förstå hur de använder programmering som undervisningsmetod för att lära ut ett matematiskt innehåll, samt varför de anser sig göra det och vilka resultat de har observerat.

13 Se www.zotero.org

5 Resultat

”Användningen av datorer är nu så självklar (trots att den inte ännu blivit självklar i undervisningen) att man inte automatiskt får en intresse-boost av att låta eleverna få tillgång till ett elektroniskt verktyg eller programvara.” (L6, 2011)

”Eleverna växer upp framför tangentbordet. […] Einstein sa att hans penna var smartare än han själv. Vi är vana vid att tänka med papper och penna. Eleverna tänker med tangentbordet.” (L5, 2011a)

Detta kapitel beskriver empirin för studien. Empirin är uppdelat i tre delar: angivna syften, vad man gör och observerade resultat. De kopplingar som görs till kursplanerna för matematik är dock, med några få undantag, ett resultat av min, författarens, egen analys. I kapitel 5.4 ges en överblick över alla innehållet i hela detta kapitel.

5.1 Angivna syften

Detta kapitel beskriver de syften som de intervjuade lärarna angav till varför de valde att använda programmering för att lära ut ett matematiskt innehåll.

5.1.1 Ett exempel på hur matematik kan användas

Flera av de intervjuade lärarna pekade på att programmering är ett bra exempel på hur matematiken kan komma till användning (L1, 2011; L11, 2011; L2, 2011; L8, 2011). Detta blir extra starkt när eleverna dessutom har möjlighet att praktisera sin matematik inom applikationsområdet, d.v.s. programmering i detta fall. Eleverna har därmed själva möjlighet att utforska kopplingen mellan matematik och dess applikationsområde (L1, 2011; L11, 2011; L2, 2011; L5, 2011a; L8, 2011).

En anledning som nämndes till varför just programmering är ett extra bra område för laborationer inom matematik är att man får direkt feedback. Man måste skriva exakt rätt och kan direkt se resultatet av det man programmerat (L5, 2011a).

5.1.2 Mer dynamisk undervisning i matematik ock ökad förståelse för centrala matematiska begrepp

Vissa av de intervjuade lärarna ansåg att den ”normala” matematikundervisningen var för statisk. De ansåg att det inte fanns så mycket möjligheter för eleverna att utforska matematiken och fundera mer på egen hand och se saker från olika synvinklar. Programmering kunde vara ett verktyg för att göra matematikundervisningen mer dynamisk, med en friare kunskapsutveckling ansåg de (L1, 2011; L11, 2011; L5, 2011a).

Ett annat relaterat syfte som endast någon enstaka av de intervjuad lärarna uttryckte explicit, men som fanns med i bakgrunden för många av de intervjuade lärarna var att man tänkte sig att programmering ger en ökad förståelse för följande centrala begrepp/områden inom matematiken: algebra, funktioner och geometri (Bäckström, 2009; L1, 2011; L12, 2011; L8, 2011). Se även introduktionen till kapitel 5.2 för mer information.

Ett par av de intervjuade lärarna gick ännu länge och pratade om en tänkt ökad logisk förmåga och förmåga till problemlösning i största allmänhet (Bäckström, 2009; L3, 2011).

5.1.3 Programmering som ett kraftfullt verktyg för matematik

Lite närbesläktat med tanken om dynamik menade ett par av de intervjuade lärarna att programmering inom matematiken var användbart p.g.a. det är ett så kraftfullt verktyg för matematiken (L6, 2011; L7, 2011). De menade alltså att programmering kan användas för att skapa/räkna matematik. Vanligen tänkte man då på programmering av algoritmer och användningen av matematiska programmeringsprogram så som Mathematica, Matlab, Maple, mm.

En utav de intervjuade lärarna påpekade även att matematikprogrammeringsprogram så som Mathematica och liknande är ett naturligt inslag på matematiska och tekniska utbildningar. Han menade därför att det var ”en naturlig tanke att förbereda och försöka bidra till en viss vana att använda programvaror på de högskoleförberedande programmen” (L6, 2011).

5.1.4 Intresse, status och att koppla undervisningen till elevernas vardag

En utav de intervjuade lärarna berättade om när hon på sitt första fasta arbete som lärare undervisade i matematik för en samhällskunskapsklass. Vissa av eleverna där hade hon även i en valbar kurs inom webbdesign. Hon menade att matematikämnet hade låg status inom klassen, medan webbprogrammering och att programmera spel hade hög status. Detta trots att man inom webbprogrammeringen ofta hanterade rent matematiskt innehåll, så som räta linjens funktion, mm. Detta triggade igång hennes tankar om att föra in programmering in i matematikundervisningen (L5, 2011a). Flera av lärarna, och även andra källor, pratade i liknande ord om att många elever som var svagare i matematik ofta tyckte det var roligare att programmera (Bootstrap, 2011a; L1, 2011; L5, 2011a; Urner, 2010). Detta hänger i sin tur samman med att relatera undervisningen i skolan till elevernas vardag och intresse, vilket också var något som många lärare nämnde (L1, 2011; L10, 2011; L5, 2011a; L8, 2011). En lärare nämnde spelet ”Angry birds”[14] som har nått ofantlig framgång[15] och som många elever spelar på sina smartphones (L8, 2011). Eleverna har en relation till mobil/data/tv-spel i sin vardag och därigenom finns det en öppning för att relatera till spelprogrammering som man i sin tur kan relatera till matematik.

Dewey menade att det, för elever, inte räcker med ett diffust mål i framtiden för utbildning. Utbildningen måste vara relevant här och nu och relatera till elevernas vardag utanför skolan (Dewey, 1997, ss 93–95).

5.1.5 Repetition av matematiken på programmeringslektionerna

För de lärare som undervisar ett matematikinnehåll inom kurserna för programmering finns det möjlighet att relatera undervisningen i programmeringen till den i matematik. En utav de intervjuade lärarna menade att den främsta anledningen för henne till ämnesintegration mellan matematik och programmering var möjligheten till repetition av det matematiska innehållet på programmeringslektionerna och därigenom förhoppningen att eleverna ska nå bättre resultat inom matematik (L1, 2011).

14 http://www.rovio.com/en/our-work/games/view/1/angry-birds

15 ”"Angry Birds" has become the fastest growing game in history, scoring 500 million downloads in less than two years since its December 2009 release, the most for any game all-time.” (Smith, 2011)

5.1.6 Matematik som grund för att lära sig (spel)programmering

De flesta lärare verkade vara överens om att om man vill lära ut programmering är spelprogrammering en bra ingång för elever. För att kunna programmera spel behövs kunskaper inom matematik (L5, 2011a; L8, 2011). Även lärare inom programmering som inte direkt tänkte på spelprogrammering såg matematik som en naturlig del av programmering (L2, 2011; L4, 2011a).

En utav de intervjuade lärarna berättade att de brukade introducera programmeringsmoment inom de första matematikkurserna för att underlätta introduktionen till programmeringskurserna som då började i årskurs två (L3, 2011).

5.2 Vad man gör

Detta kapitel beskriver vad de intervjuade lärarna konkret gjorde när de använde sig av programmering för att lära ut ett matematiskt innehåll.

Flera av de intervjuade lärarna nämnde direkt eller indirekt ett par allmänna kopplingar mellan programmering och matematik. Dessa allmänna kopplingar kan ses i Tabell 1.

Programmeringsmoment	**Matematikkurs**
Funktionsbegreppet De matematiska funktioner som hanteras på gymnasiet är vanligen (hyfsat) kontinuerliga och har alltid med tal att göra. De funktioner som används inom programmeringen ser visserligen annorlunda ut, oftast lite mer lika diskreta funktioner, men grundtanken är ändå densamma. Ett par av de intervjuade lärarna menar därför att man genom att hålla på med funktioner inom programmering även ökar den allmänna förståelsen för funktioner inom matematiken.	Alla utom 1a. ”funktion” nämns totalt 50 ggr i kursplanerna för matematik.
Algebra Argumentationen här liknar den för funktionsbegreppet. Variabler inom programmering hanteras precis på samma sätt som variabler inom matematiken. Genom att använda variabler inom programmering ökar även ens förståelse för tankekonstruktionen algebra inom matematiken.	Alla. ”algebra” nämns totalt 56 ggr i kursplanerna för matematik.
Geometri Genom att programmera grafiska saker (spel, användargränssnitt, webbsidor, mm) stöter man oundvikligen på geometri. Tydligast blir detta kanske inom spelprogrammering där olika typer av geometriska objekt ska interagera med varandra och där man även kan få in trigonometri på olika sätt.	Alla utom 5. ”geometri” nämns totalt 22 ggr i kursplanerna för matematik.

Tabell 1: Översikt över nämna allmänna kopplingar mellan programmering och matematik och hur de knyter anknytning till kursplanerna inom matematik för gymnasiet. (L1, 2011; Skolverket, 2011)

5.2.1 Genomgående ämnesintegration mellan matematik och programmering

Två av de intervjuade lärarna berättade om att ämnesintegrationen mellan matematikkurserna och programmeringskurserna gick som en röd tråd genom utbildningen. De hade ingen statisk plan för hur ämnesintegrationen skulle gå till utan anpassade ämnesintegrationen till de klasser de arbetade med. Ämnesintegrationen gick åt båda håll. De förde in programmering på matematiklektionerna och matematik på programmeringslektionerna. I båda fallen gällde det skolor där klasserna hade en inriktning mot spelprogrammering. Det var alltså inte vilka klasser som helst (L1, 2011; L3, 2011).

5.2.2 Programmering inom matematikkurser

5.2.2.1 Lösa uppgifter i matematikboken med hjälp av program

På ett par av de intervjuades skolor fanns det programinriktningar mot programmering eller liknande (L1, 2011; L4, 2011b). En utav de intervjuade lärarna berättade att undervisningen i matematik och programmering var väldigt integrerad, speciellt vad gäller avsnittet om geometri (Matematik 1a, 1b, 1c, 2a, 2b, 2c (Skolverket, 2011)). Eleverna fick då välja om de skulle lösa uppgifterna i läromedlet för matematik som vanligt eller om de skulle programmera program som löste uppgifterna, vilket var det mest populära valet. Uppgifterna bestod då i att programmera program som räknade ut olika typer av areor och volymer, mm (L1, 2011).

5.2.2.2 Programmera algoritmer på miniräknaren

I flera matematikböcker för gymnasiet förekommer uppgifter där man programmerar enklare matematiska algoritmer på sin grafritande miniräknare (se kap 7.1). Ett par utav de intervjuade lärarna nämnde att de har olika undervisningsmoment där hon låter eleverna programmera ett antal olika algoritmer. Nämnda algoritmer och dess kopplingar till gymnasieskolans kursplaner i matematik kan ses i Tabell 2.

Flera av de intervjuade lärarna pratade om den tidigare kursen Matematik – Diskret[16] som en kurs som mer eller mindre var upplagd som en programmeringskurs. En utav de intervjuade lärarna berättade att de använde ett läromedel[17] för kursen där programmering på grafritande miniräknare var en väsentlig del av hela kursen (L2, 2011).

16 http://www.skolverket.se/forskola_och_skola/gymnasieutbildning/gymnasieskola_fore_ht_2011/2.3034/sok_amnen_och_kurser?_xurl_=http%3A%2F%2Fsvcm.skolverket.se%2Fsb%2Fd%2F2503%2Fa%2F13845%2Ffunc%2Fkursplan%2Fid%2F3214%2FtitleId%2FMA1207%2520-%2520Matematik%2520-%2520diskret

17 Diskret matematik för gymnasiet: http://www.liber.se/Gymnasium/Matematik-naturvetenskapliga-amnen-teknikutveckling/Matematik/Matematik---diskret/Kurslaromedel/Diskret-matematik/

Moment	Matematikkurs
Primtal och delbarhet	1b, 1c
Geometriska[18] och aritmetiska summor[19]	3b, 3c
Newton-Raphsons metod[20]	3b, 3c
Trapetsmetoden för integralberäkning[21]	3b, 3c
Eulers formel för lösning av differentialekvationer[22]	5
Diskret matematik	(1), (3), 5

Tabell 2: Översikt över matematiska algoritmer som ett par av de intervjuade lärarna låter sina elever programmera på sina grafritande miniräknare, och dessa algoritmers kopplingar till gymnasieskolans kursplaner för matematik (L2, 2011; L5, 2011a; L7, 2011; Skolverket, 2011).

5.2.2.3 Demonstrationer

Ett par av de intervjuade lärarna använde programmering under någon genomgång för att bättre illustrera något specifikt moment inom matematiken. Syftena varierade mellan programmering som ett kraftfullt verktyg (kap 5.1.3) och hur man kan använda sig av matematik (kap 5.1.1). En översikt över moment där intervjuade lärare använde sig av programmering

Demonstration	Matematikkurs
Slumpförsök Exempelvis 10000 tärningskast	1a, 1b, 1c
Primtal Eratosthenes såll[23]	1b, 1c
Mittpunktsformeln Sierpinskitriangel[24]	1c
Cirkelns funktion Animation av cirkel som rör sig	3c
Komplexa tal Mandelbrotfraktalen[25]	4

Tabell 3: Översikt över hur programmering har använts för att demonstrera moment inom matematiken, samt dessa moments kopplingar till kursplanerna för matematik (L2, 2011; L6, 2011; L7, 2011; Lindholm, 2011; Skolverket, 2011).

18 http://sv.wikipedia.org/wiki/Geometrisk_summa
19 http://sv.wikipedia.org/wiki/Aritmetisk_summa
20 http://en.wikipedia.org/wiki/Newton's_method
21 http://en.wikipedia.org/wiki/Trapezoidal_rule
22 http://en.wikipedia.org/wiki/Euler_method
23 http://sv.wikipedia.org/wiki/Eratosthenes_s%C3%A5ll
24 http://sv.wikipedia.org/wiki/Sierpinskitriangel
25 http://sv.wikipedia.org/wiki/Mandelbrotm%C3%A4ngden

5.2.2.4 Kursen ”Matematik - breddning” som en programmeringskurs

På skolan där en utav de intervjuade lärarna arbetade samarbetade en lärare inom matematik och en lärare inom programmering om att ansvara för samma kurs, kursen ”Matematik – breddning”[26]. Kursen var där upplagd som en programmeringskurs med matematiskt innehåll och båda lärarna närvarade på alla lektioner. Tabell 4 visar den intervjuade lärarens grovplanering för kursen ”Matematik – breddning”. Förutom momenten i tabellen ingick även en projektuppgift.

Moment	Matematikkurs
Första- och andragradsfunktioner Studsande boll	1b, 1c, 2a, 2b, 2c
Matriser och linjära funktioner med matriser Vrida bilder	(ingen)
Kryptering	(5)
Kombinatorik Kortspel	5

Tabell 4: Grovplanering för kursen "Matematik - breddning" på skolan där en utav de intervjuade lärarna arbetade, samt dessa moments koppling till kursplanerna för matematik. (L4, 2011c; Skolverket, 2011)

För mer material se kapitel 7.8.

5.2.3 Matematiklärande inom programmeringskurser

5.2.3.1 Program som löser uppgifter liknande de i matematikboken

Liksom det fanns exempel på de som löste uppgifter i matematikboken med hjälp av att programmera program, fanns det liknande exempel på de som använde uppgifter mycket liknande de i matematikboken för att lära ut grunderna inom programmering. Tabell 5 ger en överblick över ett par exempel på sådana uppgifter och deras kopplingar till gymnasieskolans kursplaner för matematik.

Exempel på uppgifter	Matematikkurs
Procentbegreppet Gör ett program där man kan mata in en varas pris före moms. Momsen beräknas som 25% av varans pris före moms.	1a, 1b, 1c

26 http://www.skolverket.se/forskola_och_skola/gymnasieutbildning/gymnasieskola_fore_ht_2011/2.3034/sok_amnen_och_kurser?_xurl_=http%3A%2F%2Fsvcm.skolverket.se%2Fsb%2Fd%2F2503%2Fa%2F13845%2Ffunc%2Fkursplan%2Fid%2F3213%2FtitleId%2FMA1206%2520-%2520Matematik%2520-%2520breddning

Exempel på uppgifter	Matematikkurs
Linjära funktioner Markus är ute och kör bil. Tanken rymmer 50 liter. Priset per liter är 13,50 kr. Skriv ett program som frågar hur mycket bensin som finns i tanken. Om mängden är mindre än 10 liter ska programmet presentera en utskrift där det framgår att han måste tanka, hur mycket han ska fylla på och vad det kostar att få full tank. Är mängden minst 10 liter innehåller utskriften bara en uppmaning att köra vidare.	1b, 1c
Enkel sannolikhet Tillverka ett program som slår en sexsidig tärning 1000 gånger. Använd en vektor med sex fack för att lagra antalet slagna ettor, tvåor o.s.v. Gör programmet så kort som möjligt!	1a, 1b, 1c
Sinus och cosinus Tillverka en tabell över sinus och cosinus för alla vinklar mellan 0 och 6,3 med intervallet 0,7.	1a, 1c
Primtal Ett primtal är ett tal som inte är delbart med några andra tal än med talet 1 och med sig själv. Skriv ett program som skriver ut alla primtal som är mindre än eller lika med n. Talet n ges som indata till programmet. Det enklaste sättet att undersöka om ett visst tal i är ett primtal är att dividera det med alla tal mellan 2 och i-j och för varje tal undersöka om resten blir noll.	1b, 1c
Funktionsvärden för en tredjegradsfunktion Konstruera ett program som skriver ut en snygg tabell med värden för funktionen: $f(x) = 3x^3 - 5x^2 + 2x - 20$	1b, 1c, (3c)
Exponentialfunktioner En man erbjuds ett ovanligt riskfyllt arbete. Lönesättningen är också ovanlig. För första dagen erbjuds han 1 öre, för andra 2, för tredje 4, för fjärde 8 o.s.v. Lönen fördubblas alltså varje dag. Skriv ett program som beräknar hur många dagar mannen måste arbeta för att tjäna en miljon kronor.	2a, 2b, 2c

Tabell 5: Exempel på programmeringsuppgifter i QBasic med kopplingar till kursplanerna i matematik (Egeberg, 2007; Skolverket, 2011; Westh, 2011).

5.2.3.2 Spelprogrammering

Något som var genomgående för majoriteten av de intervjuade lärarna var att de pratade om möjligheterna med att lära ut ett matematiskt innehåll inom ramen för spelprogrammering.

Programmeringsmoment	Matematikkurs
Koordinatsystem • Spelet sänka skepp[27] • Grafiska spel i största allmänhet	1a
Enkel statistik Poäng, mm	1a
Linjära funktioner Skjuta burkar: Spelet går ut på att man ska skjuta ner ett antal burkar genom att ange lutningen på en rät linje (y=kx+m). Pistolens mynning finns i ett tänkt origo och burkarna ligger utplacerade på ett fast y-värde. Varje burk upptar ett område mellan två x-värden. Då man ska skjuta anger man k (lutningen) och programmet räknar ut om man får en träff. Man får skjuta t.ex. 10 skott. Vill man kan man utöka programmet med att lägga burkarna på olika y-värden samt be användaren mata in vilket m-värde man skjuter från. Man kan slumpa fram positionerna eller ange dem ”fast” i programmet.	1b, 1c
Andragradsfunktioner • Studsande boll • Gravitation • Acceleration • Kasta granater mot ett mål. Ställ in vinkel och styrka.	2a, 2b, 2c
Kombinatorik Kortspel	5

Tabell 6: Översikt över moment inom spelprogrammering och deras kopplingar till kursplanerna för matematik (L1, 2011; L5, 2011a; L8, 2011; Skolverket, 2011).

5.2.3.3 Webbdesign

Spelprogrammering och webbprogrammering är områden som går ihop med varandra. Vissa av de intervjuade lärarna berättade om spelprogrammering på webben. Detta subkapitel beskriver dock de moment inom webbprogrammering som inte är spelprogrammering. En översikt över nämnda moment inom webbprogrammering och deras kopplingar till kursplanerna för matematik kan ses i tabell 7.

Moment	Matematikkurs
Koordinatsystem Layout på webbsida.	1a
Procent • Layout på webbsida. • Skalning av bilder på ett online fotogalleri	1a, 1b, 1c

Tabell 7: Översikt över moment inom webbprogrammering och deras kopplingar till kursplanerna för matematik (Bäckström, 2010; Skolverket, 2011).

27 http://en.wikipedia.org/wiki/Battleship_(game)

5.2.3.4 Rita geometriska figurer

Två av de intervjuade lärarna nämnde geometriska moment liknande de som Hedrén använde vid för sina studier kring den effekt programmering i LOGO har på matematikinlärningen (se kap 3.1.3 och kap 7.6) (Hedrén, 1988). Den ena läraren berättade att han brukade ge eleverna uppgiften att programmera en ritad snögubbe (dock ej i LOGO, utan i Comal[28]) (L2, 2011). Den andra läraren berättade att han brukade introducera programmering med hjälp av olika uppgifter i LOGO. Han hade bl.a. gjort ett spel där eleverna skulle skriva ett program som förde en figur runt en bana (L8, 2011).

5.2.3.5 Samarbete med läraren i matematik

På en skola där en utav de intervjuade lärarna tidigare arbetat hade de en inriktning mot spelutveckling. Lärarna inom matematik och de inom programmering samarbetade då på så sätt att man var en del av samma arbetslag och att man arbetade aktivt med att försöka ta in varandras matematiska innehåll i sina egna kurser. Det matematiska innehåll som diskuterades på matematiklektionen försökte användas på programmeringslektionen och vice versa (L5, 2011a).

5.2.3.6 Project Euler

En av lärarna visar Project Euler (se kap 7.3) för sina elever i programmering som en frivillig uppgift. Ett par elever per år hakar på (L8, 2011).

5.3 Observerade effekter

Detta kapitel beskriver de observerade effekter, som ett resultat av att de valt att använda programmering för att lära ut ett matematiskt innehåll, som de intervjuade lärarna angav.

En utav de intervjuade lärarna hade genomfört sin ämnesintegration mellan matematik och programmering på ett mer forskningslikt sätt. För de övriga intervjuade lärarna genomfördes denna ämnesintegration inom ramen för den normala undervisningen. Det var därför ofta svårt att särskilja syfte från observerade effekter och det var i princip omöjligt att fastställa kausaliteten för de observerade effekterna. Med detta i bakhuvudet presenteras nedan en översikt över de effekter av ämnesintegrationen som de intervjuade lärarna hade observerat.

5.3.1 Fördelar

5.3.1.1 Ökad färdighet för begrepp/områden som genomsyrar hela matematiken

Ökad generell matematisk färdighet var inte något som observerats av många av de intervjuade lärarna. Det var snarast den intervjuade läraren som hade genomfört sin ämnesintegration på ett mer forskningslikt sätt som beskrev dessa fördelar. Jag upplevde att hennes beskrivningen av fördelarna var en blandning av relaterad forskning på området och hennes egna observationer:

> *”Att programmering har positiva effekter på utvecklingen av matematiska begrepp som variabler och funktioner är konstaterat i flera oberoende undersökningar. Forskare som har studerat samband mellan möjligheter att utöva programmering och förståelsen för algebra är bla Thomas Tall, Anna Sfard, Clements. Resultat från deras studier visa bland annat att elever som tränar matematiska begrepp genom programmering:*

28 http://en.wikipedia.org/wiki/COMAL

får djupare förståelse för begreppen som tränas
kan använda begreppen på ett mognare sätt
kan lösa svårare problem
och har bättre strategier för problemlösning

jämfört med elever i kontroll grupp som inte programmerar. Vi har inte ännu studerat effekten som spel och webb programmering har på kvaliteten på elevernas matematiska begrepp [...]. Våra erfarenheter pekar dock tydligt på att eleverna börjar se matematiken ur ett nytt perspektivt, något man kan använda...” (Bäckström, 2009)

Hon tog upp ökad förståelse för följande begrepp/områden (Bäckström, 2009):

- Algebra
- Funktioner
  - Reifikation (objektifiering) av funktionsbegreppet (Sfard, 1991)
- Generalisering
- Formell beskrivning

Detta kan relateras till syftet beskrivet i kapitel 5.1.2 och introduktionen till vad man gör i kapitel 5.2.

5.3.1.2 Ökad motivation för att lära sig matematik (dock bara under programmeringen)

Genom att använda sig av matematik inom programmeringskurser fann eleverna mer motivation att lära sig ett matematisk innehåll än de fann inom matematikkurserna. Denna motivation verkade dock genomgående vara knuten till kontexten, d.v.s. eleverna tyckte fortfarande inte att det blev roligare med matematik i största allmänhet, bara den matematik som behandlades genom programmering. Flera olika källor till motivationen angavs, vilket även kan ses i det relaterade syftet i kapitel 5.1.4. Bland annat ansåg en utav de intervjuade lärarna att viss typ av programmering hade högre status i vissa klasser än matematikämnet. Den direkta feedbacken angavs också vara en källa som triggar eleverna till att själva ta sig framåt (L1, 2011; L5, 2011a). Flera lärare nämnde att eleverna blev inspirerade av att det faktiskt gick att använda matematiken till något de själv ansåg vara användbart (se kap 5.1.1) (L2, 2011).

Även om en utav de intervjuade lärarna menade att teknik och själva datoranvändandet i sig inte längre ger ökat intresse (p.g.a. elevernas extensiva kontakt med teknik och datorer i vardagen) (L6, 2011), menade ändå flera av de intervjuade lärarna att vissa typer av programmering knöt an till elevernas vardag och intressen, så som webbprogrammering och spelprogrammering. Detta menade man ökade motivationen (L5, 2011a; L8, 2011).

5.3.1.3 Göra oöverskådliga förlopp överskådliga och abstrakta begrepp greppbara

Hur ska man kunna visa att sannolikhet verkligen stämmer? Jo, låt datorn kasta en tärning 10000 gånger eller en miljon gånger. Hur kan man få eleverna att förstå hur fraktaler är uppbyggda. Visa den enkla algoritm som ligger till grund (L6, 2011). Det uppgivna syfte som relaterar bäst till detta är programmering som kraftfullt verktyg för matematik (kap 5.1.3). På liknande sätt beskrev en utav de intervjuade lärarna programmeringens möjlighet att göra abstrakta begrepp greppbara, genom att det sätts in i en kontext som eleven själv skapar (Bäckström, 2009).

5.3.1.4 Ökad färdighet inom programmering

Lärarna som undervisade i programmering verkade generellt sett se matematik som en del av programmering. För att kunna göra vissa saker inom programmeringen måste eleverna helt enkelt kunna ett visst matematiskt innehåll. Genom att föra in detta matematiska innehåll i programmeringsundervisningen blir eleverna helt enkelt bättre på att programmera. (L3, 2011; L4, 2011a; L5, 2011a; L8, 2011; L9, 2010).

En av de intervjuade lärarna som angav förenklad introduktion till programmering som ett utav syftena till ämnesintegrationen mellan matematik och programmering hade funnit att det gav resultat: ”elever […] blir inte skrämda av att se kod” (L3, 2011).

5.3.1.5 Lärarens glöd

När jag frågade frågor kring hur mycket extra tid som lades på att ämnesintegration mellan matematik och programmering svarade ett par av de intervjuade lärarna att det var något de tyckte var kul, så den extra tiden som lades ner upplevdes inte som en belastning. Ämnesintegrationen gjorde arbetet roligare. Förutom att dessa lärarna själv tyckte att jobbet blev roligare fanns även förhoppningen om att denna glöd skulle sprida sig till eleverna (L4, 2011b; L6, 2011).

5.3.2 Nackdelar / svårigheter

5.3.2.1 Tidsbrist

Tidsbrist var en faktor som återkom i de flesta intervjuer av lärare som jag genomförde. Det nämndes framför allt som anledning till att ämnesintegration mellan matematik och programmering inte var implementerad i större utsträckning (se 6.5.3 Tidsbrist).

På de skolor där det förekom ett samarbete mellan lärare inom programmering och lärare inom matematik tyckte de dock inte att den extra tid som lades på samarbetet var någon extra upplevd belastning (L4, 2011b; L5, 2011a).

5.3.2.2 Tekniska problem

Ett par av de intervjuade lärarna berättade om strikta regler kring vilken mjukvara som får installeras på skolornas datorer och hur installationen ska gå till. Detta gjorde det i vissa fall mycket svårt att installera de program som behövdes, eller upplevdes som mest lämpliga, för den typ av programmering läraren tänkte sig (L5, 2011a; L6, 2011). En av de intervjuade lärarna berättade exempelvis att hon lät eleverna programmera i notepad (Anteckningar) för att hon inte lyckades få till stånd en installation av något mer användarvänligt utvecklingsverktyg (L5, 2011a).

5.3.2.3 Vissa elever tycker inte om att programmera

En lärare uttryckte att vissa elever ville räkna matematik ”som vanligt” (trots att de valt en gymnasial inriktning mot spelprogrammering) och inte såg meningen med en ämnesintegration mellan matematik och programmering. För dessa elever upplevdes ämnesintegrationen som ett hinder (L4, 2011b).

5.4 Översikt över de intervjuade

Tabell 8 ger en översikt över de intervjuade och de syften de angav till ämnesintegrationen mellan matematik och programmering. Tabell 9 ger en översikt över vad som gjordes inom ämnesintegrationen samt vilka effekter av den som hade observerats.

Id	Intervjutyp	Typ av int.		Syfte												
		P inom M?	M inom P?	Anv-omr.	Lab-möjl.	Dyn.	Alg. Funk. Geo.	Allm. log. & prob.	P kraftf. verkt. för M	Förb. högsk.	Rep M	M kraft. för P	Mer tid för P	Intres. / Koppla till elev.	Status	Lär. glöd
L1	1 – Intervju	Ja	Ja	X	X	X	(X)				X	(X)		X		
L2	1 – Intervju	Ja	Ja	X	X				(X)			(X)				
L3	2 – Mailintervju	Ja	Ja					X					X			
L4	1 – Intervju	Ja	Nej									X				X
L5	1 – Intervju	Lite	Ja		X	X	X	X				X		X	X	
L6	2 – Mailintervju	Lite	Nej						X	X						X
L7	3 – Fritt mail	Lite	Nej						X							
L8	1 – Intervju	Nej	Lite	X	X		X					X		X		
L9	3 – Fritt mail	Nej	Lite									(X)				
L10	3 – Fritt mail	Nej	Lite									X		X		
L11	3 – Fritt mail	Nej	?	X	X	X										
L12	3 – Fritt mail	Nej	?				X									

Tabell 8: Översikt över de intervjuade lärarna, deras typ av ämnesintegration, samt de syften de angav till ämnesintegrationen. Färgerna är ett försök att klumpa ihop liknande syften. Nedan följer ett förtydligande över syftesrubrikerna.

M kraft. för P: *Matematik är ett kraftfullt verktyg för programmering och krävs för vissa typer av programmering.*
Anv-omr.: *Programmering är ett exempel på användningsområde för matematik.*
Lab-möjl.: *Programmering är ett sätt att laborera inom matematikområdet.*
Dyn.: *Programmering är ett mer dynamiskt, icke-statiskt, sätt att lära sig matematik än ”normal” matematikundervisning.*
Alg. Funk. Geo.: *Programmering ökar förståelsen för centrala matematiska begrepp/områden, framför allt algebra, funktioner och geometri.*
Allm. log. & prob.: *Programmering ökar den allmänna logiska förmågan och förmågan till problemlösning.*
P kraft. verkt. för M: *Programmering är ett kraftfullt verktyg för matematik*
Förb. högsk.: *Användandet av matematiska programmeringsverktyg, t.ex. Mathematica, är en förberedelse för tekniska högskolestudier.*
Rep M: *Genom att föra in matematiskt innehåll på programmeringslektionerna repeteras innehållet från matematikkurserna*
Mer tid för P: *Genom att introducera programmering på matematiklektionerna får eleverna lättare när de läser programmeringskurser.*
Intres. Koppla till elev: *Att använda sig av webb- och spelprogrammering i undervisningen är ett sätt att knyta an till elevers intresse och vardag.*
Status: *Webb- och spelprogrammering har högre status än matematik i vissa typer av klasser. Det finns därför högre motivation där.*
Lär. glöd: *Att använda programmering inom matematiken ökar glöden och lusten i undervisningen för läraren själv.*

Id	Genomg.	P inom M						M inom P						Resultat							
		Lösa uppg. i M-bok mha P	Prog. minir.	M alg	Demo	Diskr.	M-breddn.	Lösa M-uppg. inom P	Spel-prog.	Webb-des.	Rita geo.	Samarb. med M-lärare	Proj. Euler	Ökad motiv.	Ökad färd. i P	Ökad färd. i M	Lär. glöd	Tids-brist	Tek. prob.	Arb. lag tänk	E ville ej P
L1	X	X						X	X					X	X						
L2			X	X	X	X				X	X										
L3	X	Data saknas													X						
L4							X								X		X				X
L5			X	X					X	X		X		X		X		X			
L6					X												X	X	X	X	
L7			X	X	X											(X)					
L8									(X)		X		X	X	X			X			
L9								X	(X)						X						
L10									X					Data saknas							
L11	Data saknas																				
L12	Data saknas																				

Tabell 9: Mer specificerad översikt över hur de intervjuade lärarna arbetade med ämnesintegration mellan matematik och programmering, samt vilka resultat de observerat. Nedan följer ett förtydligande av de rubriker som används i tabellen.

Genomg.: *Genomgående, d.v.s. ämnesintegrationen mellan matematik och programmering genomsyrar båda ämnena.*
P inom M *Programmering inom matematikkurser*
M inom P *Matematik inom programmeringskurser*
Lösa uppg. i M-bok mha P: *Lösa uppgifter i läromedlen för matematikkurserna med hjälp av programmering.*
Diskr.: *Programmering genomsyrar matematikkursen för diskret matematik.*
Prog. minir.: *Programmering på grafritande miniräknare.*
M alg: *Programmera matematiska algoritmer.*
Demo: *Programmering används av läraren för att demonstrera något för eleverna.*
M-breddn. *Kursen ”Matematik – breddning” körs som en programmeringskurs med matematiskt innehåll.*
Lösa M-uppg. inom P *Lösa uppgifter liknande de i läromedlet för matematik med hjälp av programmering.*
Spelprog. *Spelprogrammering används med det medvetna syftet att lära ut ett matematiskt innehåll.*
Webbdes. *Webbdesign används med det medvetna syftet att lära ut ett matematiskt innehåll.*
Rita geo. *Rita geometriska figurer med hjälp av programmering. Se Hedrén (kap 3.1.3) och LOGO turtle (kap 7.6).*
Samarb. med M-lärare *Läraren i programmering samarbetar med läraren i matematik så att man tar upp varandras matematiska innehåll på sina lektioner.*
Proj. Euler *Project Euler (se kap 7.3).*
Ökad motiv. *Ökad motivation bland eleverna.*
Ökad färd. i P *Ökad färdighet i programmering.*
Ökad färd. i M *Ökad färdighet i matematik.*
Lär. glöd *Ökad glöd och energi för läraren i undervisningen.*
Tidsbrist *Tidsbrist.*
Tek. prob. *Tekniska problem.*
Arb. lag tänk *Att man har olika tankar i arbetslaget kring hur teknik ska användas i matematikundervisningen.*

5.5 Översikt över nämnda exempel på undervisning av ett matematiskt innehåll med hjälp av programmering

Tabell 10 nedan ger en översikt över alla de exempel, på undervisning av ett matematiskt innehåll med hjälp av programmering, som de intervjuade lärarna nämnde.

Moment	Matematikkurs
Matriser och linjära funktioner med matriser Vrida bilder	(ingen)
Enkel statistik Poäng, mm	1a
Enkel sannolikhet Tillverka ett program som slår en sexsidig tärning 1000 gånger. Använd en vektor med sex fack för att lagra antalet slagna ettor, tvåor o.s.v. Gör programmet så kort som möjligt!	1a, 1b, 1c
Koordinatsystem • Spelet sänka skepp[29] • Grafiska spel i största allmänhet • Layout på webbsida	1a
Procentbegreppet • Gör ett program där man kan mata in en varas pris före moms. Momsen beräknas som 25% av varans pris före moms. • Layout på webbsida. • Skalning av bilder på ett online fotogalleri	1a, 1b, 1c
Slumpförsök Exempelvis 10000 tärningskast	1a, 1b, 1c
Geometri Rita geometriska figurer med hjälp av programmering, exempelvis LOGO turtle	1a, 1b, 1c, 2a, 2b, 2c
Sinus och cosinus Tillverka en tabell över sinus och cosinus för alla vinklar mellan 0 och 6,3 med intervallet 0,7.	1a, 1c

29 http://en.wikipedia.org/wiki/Battleship_(game)

Moment	Matematikkurs
Linjära funktioner Skjuta burkar: Spelet går ut på att man ska skjuta ner ett antal burkar genom att ange lutningen på en rät linje (y=kx+m). Pistolens mynning finns i ett tänkt origo och burkarna ligger utplacerade på ett fast y-värde. Varje burk upptar ett område mellan två x-värden. Då man ska skjuta anger man k (lutningen) och programmet räknar ut om man får en träff. Man får skjuta t.ex. 10 skott. Vill man kan man utöka programmet med att lägga burkarna på olika y-värden samt be användaren mata in vilket m-värde man skjuter från. Man kan slumpa fram positionerna eller ange dem ”fast” i programmet. Markus är ute och kör bil. Tanken rymmer 50 liter. Priset per liter är 13,50 kr. Skriv ett program som frågar hur mycket bensin som finns i tanken. Om mängden är mindre än 10 liter ska programmet presentera en utskrift där det framgår att han måste tanka, hur mycket han ska fylla på och vad det kostar att få full tank. Är mängden minst 10 liter innehåller utskriften bara en uppmaning att köra vidare.	1b, 1c
Primtal Ett primtal är ett tal som inte är delbart med några andra tal än med talet 1 och med sig själv. Skriv ett program som skriver ut alla primtal som är mindre än eller lika med n. Talet n ges som indata till programmet. Det enklaste sättet att undersöka om ett visst tal i är ett primtal är att dividera det med alla tal mellan 2 och i-j och för varje tal undersöka om resten blir noll. Eratosthenes såll[30] (se kap 7.1.3)	1b, 1c
Funktionsvärden för en tredjegradsfunktion Konstruera ett program som skriver ut en snygg tabell med värden för funktionen: $f(x) = 3x^3 - 5x^2 + 2x - 20$	1b, 1c, (3c)
Första- och andragradsfunktioner Studsande boll	1b, 1c, 2a, 2b, 2c
Mittpunktsformeln Sierpinskitriangel[31] Se kap 7.10.	1c
Andragradsfunktioner • Studsande boll • Gravitation • Acceleration • Kasta granater mot ett mål. Ställ in vinkel och styrka.	2a, 2b, 2c

30 http://sv.wikipedia.org/wiki/Eratosthenes_s%C3%A5ll
31 http://sv.wikipedia.org/wiki/Sierpinskitriangel

Moment	Matematikkurs
Exponentialfunktioner En man erbjuds ett ovanligt riskfyllt arbete. Lönesättningen är också ovanlig. För första dagen erbjuds han 1 öre, för andra 2, för tredje 4, för fjärde 8 o.s.v. Lönen fördubblas alltså varje dag. Skriv ett program som beräknar hur många dagar mannen måste arbeta för att tjäna en miljon kronor.	2a, 2b, 2c
Geometriska[32] och aritmetiska summor[33] Se kap 7.1.8.	3b, 3c
Newton-Raphsons metod[34]	3b, 3c
Trapetsmetoden för integralberäkning[35] Se kap 7.1.9.	3b, 3c
Cirkelns funktion Animation av cirkel som rör sig	3c
Komplexa tal Mandelbrotfraktalen[36]	4
Diskret matematik	5
Eulers formel för lösning av differentialekvationer[37] Se kap 7.1.10.	5
Kombinatorik Kortspel	5
Kryptering	5

Tabell 10: Översikt över alla de exempel som de intervjuade lärarna nämnt på undervisning av ett matematiskt innehåll med hjälp av programmering.

32 http://sv.wikipedia.org/wiki/Geometrisk_summa
33 http://sv.wikipedia.org/wiki/Aritmetisk_summa
34 http://en.wikipedia.org/wiki/Newton's_method
35 http://en.wikipedia.org/wiki/Trapezoidal_rule
36 http://sv.wikipedia.org/wiki/Mandelbrotm%C3%A4ngden
37 http://en.wikipedia.org/wiki/Euler_method

6 Diskussion

6.1 Grundläggande syften och mål med ämnesintegrationen

Jag tycker mig, i det insamlade materialet, kunna urskilja grupperingar utmed två olika axlar vad gäller syftet med att använda programmering för att lära ut ett matematiskt innehåll. Den ena axeln utgör målet med undervisningen. Är målet för läraren främst att eleven ska bli duktig på matematik eller programmering? Målet avgör hur ämnesintegrationen går till. Vill man relatera programmeringen till matematiken eller tvärtom? Den andra axeln utgörs av vem ämnesintegration är till för. Är det till för de svaga eleverna eller de starka? De två axlarna illustreras i illustration 2. Jag ska nedan diskutera vidare kring hur jag ser på det rum som de två axlarna spänner upp.

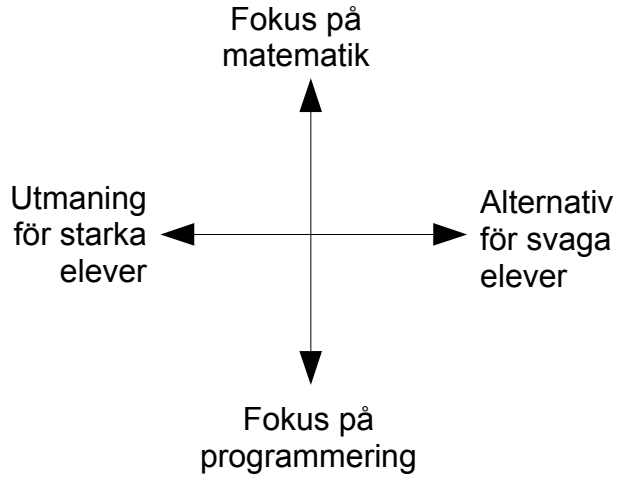


Illustration 2: Olika syn på syftet med programmering för att lära ut ett matematiskt innehåll.

6.2 Lärares och elevers glöd

Flera av de intervjuade lärarna och andra jag har varit i kontakt med under studien har pratat om programmering inom matematiken som något man behöver ha en dynamisk inställning till. Lärare är olika och har därför olika sätt att undervisa som passar dem bäst. Klasser är olika. Vissa tycker om att ”räkna i matteboken”, vissa kanske inte tycker om att räkna i matteboken, men tycker inte mycket bättre om programmering heller. Läraren måste helt enkelt undervisa på sitt sätt och anpassa sig till klassen. Flera av de intervjuade lärarna pratade om glöden och engagemanget i undervisningen. Först trodde jag mest det hade att göra med glädje över att ha en ”bra” undervisning, men tanken sträckte sig längre än så. Som människa och person finner man helt enkelt energi i att arbeta med vissa saker på ett visst sätt. Personligen tror jag att entusiasm är en utav de viktigaste aspekterna i en lärares undervisning. I kontakt med andra människor överförs alltid känslor, åt båda håll, i någon grad (Siring, 2011). Jag är övertygad om att entusiasm och liknande inre motivation är de starkaste redskapen för lärande. Att lära för att man själv vill lära sig. Det torde därför vara av högsta vikt för en lärare att finna undervisningsmetoder som öka den egna entusiasmen och motivationen.

6.3 En lösning för de svaga eleverna?

Ove Sernhede, som har ägnat sitt liv åt att arbete med och forska kring ungdomar i förorten, pratade om att de unga på 70- och 80-talen ”vägrade att låta sig reduceras till objekt för inlärning. Detta var […] kanske […] ett resultat av att eleverna helt enkelt inte förmådde vara elever på samma sätt som under tidigare generationer” (Sernhede, 2011, s 19). Han pratade om att många unga hade för mycket oro i sig och att föräldrarna inte längre framstod som identifikationsobjekt, utan att personer inom popkulturen blev starkare förebilder. Han fortsätter med:

> *”1970- och 80-talens radikala pedagogik såg det som progressivt att föra in elevernas värld i skolans arbete. Denna hållning blev också en hållbar strategi för skolorna i de stora städernas förorter. Det gick helt enkelt inte längre att upprätthålla den traditionella katederundervisningen. Skolan var, om man över huvud taget skulle ha någon möjlighet att kommunicera med de unga, tvungen att närma sig eleverna på deras villkor.” (Sernhede, 2011, s 20)*

> *”Kan det möjligen vara så att skolan i förorten åter igen måste öppna sig för den verklighet som finns utanför skolan.” (Sernhede, 2011, s 22)*

Det som Sernhede pratar om passar väl ihop med det jag läst och hört under min studie. Flera av de intervjuade lärarna såg på programmering för att lära ut ett matematiskt innehåll som en metod att motivera de svaga eleverna. Man menade att de starka eleverna var på ett annat sätt. Man behövde inte motivera dem på samma sätt och dessutom föredrog de ofta den traditionella undervisningen. Det var för de svaga eleverna som lärarna fann metoden intressant. Samma sak kommuniceras från andra sidan Atlanten. I kap 3.2.2 nämnde jag Urner som försökte skapa en matematikkurs inom en gymnasieskola (high school) i Oregon, USA, med en del programmeringsmoment, som en del av den vanliga utbildningen. Han sa ”In theory, we'd like computational math to rescue dropouts and youth at risk” (Urner, 2010). Även organisationen Bootstrap (se kap 7.4) pratar på liknande sätt om deras målgrupp. De säger bl.a. att 70% av deras studerande ungdomar får fri eller rabatterad lunch, vilket man bara får om familjen har en låg inkomst (Bootstrap, 2011a; US Department of Agriculture - Food and Nutrition Service, 2011).

Sammanfattningsvis kan man alltså säga att om läraren tycker att det är spännande med programmering inom matematikundervisningen och klassen är sådan att katederundervisning och ”räkna i matteboken” inte fungerar så bra så erbjuder programmering, då framför allt spel- och webbprogrammering, inom matematikundervisningen intressanta möjligheter.

6.4 Utmaning för de starka eleverna

Vissa av de intervjuade lärarna såg programmering som ett kraftfullt verktyg för matematiken. En utav de intervjuade lärarna såg det till och med som en förberedelse för högskolan. Här pratar vi inte om en ämnesintegration som i första hand är till för de svaga eleverna. Ett annat syfte som nämndes var möjligheten till en mer dynamisk matematikundervisning. Detta ser jag som intressant både för svaga och starka elever. De svaga eleverna kanske behöver det för att de inte ”klarar av” att vara bundna av matematikboken. För de starka eleverna ser jag möjligheten att utvecklas utöver de färdiga algoritmerna som bara ska memoriseras och appliceras. Här tänker jag mig framför allt skapandet, programmerandet, av algoritmer. Eleverna tvingas då till generalisering av

algoritmen ifråga och skapandet av egna algoritmer upplever jag nog själv som det starkaste lärandet jag haft inom matematik. Ämnesplanen för matematik nämner också undersökande aktiviteter, kreativitet och generalisering som centrala (Skolverket, 2011). Man kanske kan börja med att eleverna får skriva av en algoritm för miniräknaren eller datorn och sedan försöka skapa en egen algoritm för en specifik uppgift eller ett specifikt problem.

Då många upplever svårigheter vid introduktionen till programmering vid tekniska studier på högskolan, är det inte helt fel att ha sett en gnutta programmering innan man kommer dit (se kap 3.3.1).

6.5 Varför har det hänt så lite?

När jag började undersöka området programmering inom matematikundervisningen fann jag snabbt att tanken var långt ifrån ny (se kap 3.1). Frågan som då direkt ”ploppade upp” var: Om man nu tänkte i dessa banor för redan ca 30 år sedan, varför har det fortfarande nästan inte hänt någonting på området? Trots de tänkta fördelarna (se kap 3.1.1 och 5.1) måste det helt enkelt vara något lurt med det hela, något slags stort motargument eller något annat stort motstånd som helt enkelt väger tyngre än de tänkta fördelarna.

Jag diskuterade denna fråga lite mer med den av de intervjuade som troligen var mest insatt i området ämnesintegration mellan matematik och programmering. Under ett projekt finansierat av .se:s internetfond tog hon och hennes kollega fram ett material för att programmera för användning inom Matematik A (se kap 3.2.1). Detta var tre år sedan och de har själva fortfarande bara använt materialet inom programmeringskurser. Den självklara frågan var varför hade hon inte använt materialet inom Matematik A? Hon svarade följande:

> *”Det är enklare att arbeta med en färdig bok, kräver inga förberedelser och man kan utgå ifrån att eleverna får med sig hela kursen bara de räknar igenom boken. Etablerat och prövat, inga föräldrar som klagar. Och utan tid för förberedelser och knapp med tid för efterarbete så kör man något väl beprövat....” (Bäckström, 2011a)*

Det är många aspekter i svaret ovan, varav tidsbrist verkar vara den mest betydelsefulla. Detta stämmer även väl med den respons jag fått från andra intervjuade och icke-intervjuade lärare under studien. Den andra starka aspekten är säkerheten att eleverna får med sig ”hela kursen”. Nedan följer en lista på dessa och andra tänkbara orsaker till att programmering inte används mer inom matematikundervisningen.

- De tänkta fördelarna är inte otvetydigt bevisade
- Kunskapsbrist
- Tidsbrist
- Oro för att inte få med ”hela kursen” (se ovan)
- Korta lektioner
- Uppdelningen i kurser
- Skolledningen
- Felsatsningar
- Tekniska hinder

6.5.1 De tänkta fördelarna är inte otvetydigt bevisade

Jag kommer gång på gång på mig själv med att tänka på fördelarna med att använda programmering för att lära ut ett matematiskt innehåll som just de *tänkta* fördelarna. För trots ivriga försök att bevisa fördelarna så har bevisen inte varit otvetydigt genomgripande övertygande. Detta trots att så lång tid har förflutit sedan forskningen på området påbörjades.

6.5.2 Kunskapsbrist

Det har tidigare inte funnits någon tydlig utbildning för att bli gymnasielärare inom programmering. Samtliga av de intervjuade lärarna hade tillägnat sig kunskaper inom programmering utanför lärarprogrammet. För att bli lärare inom programmering har det alltså mer eller mindre krävts en annan utbildning, som sedan kompletteras med en lärarutbildning, alltså en dubbelexamen. Vanliga utbildningar bland de intervjuade lärarna var ingenjörsutbildningar och datavetenskapliga utbildningar. Denna typ av utbildning ger enligt mig möjlighet till signifikant bättre lön och status en läraryrket. Man har alltså varit tvungen att studera ytterligare 1-1½ år pedagogik, samt eventuella ämneskompletteringar för att få ett arbete med lägre lön och status.

6.5.3 Tidsbrist

> *”Som gymnasie lärare får du räkna med att arbeta minst 60 h i veckan om du vill ha någon kvalitet i din undervisningen och du hinner ändå inte göra allt som du är tillordnad att göra. Ska du pröva ny metoder så får du räkna med mer jobb ändå och gå r troligen in i väggen. dvs det handlar om att hålla näsan över ytan och inte mer. klimatet är alltså inte sådant att det främjar någonsomhelst utveckling” (L5, 2011b)*

Flera av de lärare jag kontaktat i samband med studien, både av de jag intervjuade och de jag inte intervjuade[38], angav också tidsbrist som orsak till att man inte utvecklat en ämnesintegration mellan matematik och programmering mer (L8, 2011; m.fl.).

6.5.4 Oro för att inte få med hela kursen

Läromedlen för matematik är väl beprövade och mappar troligen även väl mot de nationella proven. Använder man sig av samma läromedel genom alla matematikkurserna kan man även vara säker på att eleverna alltid har rätt förkunskaper för nästa moment, rakt igenom alla kurserna. Väljer man att i högre utsträckning använda sig av programmering är det mycket svårare att säkerställa att ”hela kursen” kommer med så som den bedöms på de nationella proven och så som läromedlen för de nästkommande kurserna inom matematik tänker sig.

6.5.5 Korta lektioner

De flesta skolor har en schemastruktur som bygger på korta lektioner. Detta blir problematiskt när det gäller att hålla på med datorer i allmänhet och programmering i synnerhet. Det tar alltid lite tid att starta datorerna och komma igång och när det gäller programmering tar det enligt egen erfarenhet också ett tag innan man har satt sig in i vad man höll på med att göra sist och vad man ska göra härnäst.

38 Många av de lärare jag kontaktade svarade att de inte arbetade med någon typ av ämnesintegration mellan programmering och matematik, varför det inte var intressant för mig att intervjua dem.

6.5.6 Uppdelningen i kurser

En utav de intervjuade lärarna menade att forskning pekade på att programmering som undervisningsmetod för att lära ett matematiskt innehåll ger sämre resultat i början, men bättre resultat över tid (L5, 2011a). Jag tänker mig att detta resultat inte matchar bra ihop med dagens uppdelning i olika matematikkurser, där man då eventuellt inte kan se bättre resultat inom samma kurs, utan först senare.

6.5.7 Skolledningen

En utav de intervjuade lärarna menade att ledningen på de flesta skolor i Sverige har en bakgrund som lärare inom samhällsvetenskapliga ämnen och är därför inte alltid tillräckligt insatta i de naturvetenskapliga ämnenas pedagogik och problematik. Det man själv inte är så intresserad av har självklart inte så mycket fokus hos en själv. Om det nu är så att en majoritet av personer i skolledningar i Sverige har en dragning åt det samhällsvetenskapliga hållet är det därför naturligt att de naturvetenskapliga ämnena inte ägnas lika mycket uppmärksamhet (L5, 2011b).

6.5.8 Felsatsningar

En utav de intervjuade lärarna menade att det de senaste åren har satsats mycket pengar på att höja kvalitén på matematikundervisningen på svenska skolor, men att dessa pengar inte har använts optimalt. Hon menade att pengarna borde ha gått till att ge matematiklärare bättre arbetsvillkor, så som mindre elevgrupper, mer tid för planering och utveckling, höjda löner, mm. Istället menar hon att pengarna går till andra organisationer och aktörer, framför allt universiteten, där kopplingen till undervisningsarbetet i gymnasieskolorna blir mycket svag (L5, 2011b).

6.5.9 Tekniska hinder

Flera av de intervjuade lärarna nämnde tekniska problem kring vilka program som fick installeras på skolans datorer och vem som bestämde det. En utav de intervjuade lärarna berättade exempelvis om att de använde notepad (anteckingar) som ”utvecklingsmiljö”, vilket är helt absurt om du frågar mig som har en bakgrund som IT-konsult (L5, 2011a). Detta p.g.a. hon inte fick installera de program hon hade önskat sig på skolans datorer.

6.6 Sammanfattning

En bra lärare är en icke utbränd lärare. Det handlar inte bara om hur mycket du gör, utan även vad du gör. Finner du energi och glöd i tanken på pröva något ”nytt”, så som programmering för att lära ut matematik, så tror jag både du och dina elever kommer att vinna entusiasm och även bättre resultat.

Frågan om varför det inte hänt så mycket på området är något jag återkommit till flera gånger. Sammanfattningsvis kan man säga att man fortfarande inte *vet* att programmering för att lära ut ett matematiskt innehåll verkligen är en metod som är bättre än någon annan. För övrigt verkar kunskapsbrist och tidsbrist vara två stora faktorer. I och med att jag tror att alla gymnasieelever, åtminstone på de teoretiska programmen, kommer att ha varsin laptop inom några få år (se kap 3.4) så tror jag ändå på Kairos futures prognos om att IT kommer att revolutionera utbildningen (se kap 3.2.3). Samtidigt verkar inte lärarna vara mycket mer

förberedda nu än tidigare och då IT-chefer och rektorer är mer positiva till IT än lärarna (Kairos Future, 2011b, s 7) kan förändringen upplevas som något som tryckts på uppifrån. Jag tänker mig att lärarna ändå kommer att vara tvungna att anpassa sig till den nya situationen och att förändringen kommer att ske på bred front, vilket jag tror även kommer att öppna upp för tankar kring programmering och matematik.

De tekniska hinder som lärare har berättat om vet jag inte om jag ska skratta eller gråta åt. Jag förstår idén med att ha en lista med program som är godkända för installation, samt att ha en central administration av installationerna. Jag anser dock att en person som är ansvarig för en sådan lista även bör godkänna program som han/hon vet är gratis och oskadliga[39], även om de inte står med på listan. Om den ansvarige inte besitter sådan kunskap bör denne bara godkänna förfrågningar som kommer från en lärare som ska undervisa i programmering.

Tanken om centrala och lokala (skolledningen) felsatsningar som togs upp av en av de intervjuade lärarna finner jag också intressant. Den nuvarande storsatsningen på matematik på 2,6 miljarder kr läggs på kompetensutveckling och fler matematiktimmar i grundskolan (Juvel & Hoppe, 2011). Det låter ju bra, men är det bra? Skolverkets rapport om ”Lusten att lära – med fokus på matematik” tar faktiskt upp att lärarna efterfrågar kompetensutveckling (Skolverket, 2003, s 49). Dock säger man två stycken ner

> *”Lärarnas ämneskunnande i matematik uppfattas i allmänhet som tillräcklig av skolledning och lärare i grundskolans senare år och gymnasieskolan.” (Skolverket, 2003, s 49)*

Det man vill ha i grundskolans senare år och gymnasiet är alltså inte bättre matematikkunskaper, utan lärarna eftersöker ämnesdidaktisk kunskap. För att vinna detta efterfrågas erfarenhetsutbyte och spridning av goda exempel. Man säger också att ”Behovet av pedagogiska samtal är mycket stort men en majoritet av lärarna anser att tid och utrymme saknas” (Skolverket, 2003, s 49). Här kommer tidsbristen in igen. Skulle de pengar som läggs på kompetensutveckling kanske istället för att läggas på universitetsutbildning läggas på lärare som vill undersöka sätt att göra sina matematiklektioner mer lustfyllda? Dessa sätt skulle kanske kunna bli sådana goda exempel som efterfrågas och som man kan berätta om på matematikbiennalen rapporten också sa var ett uppskattat forum för detta. En utav de intervjuade lärarna har berättat om just att använda programmering för att lära ut ett matematiskt innehåll på två omgångar av matematikbiennalen. Satsningen på fler matematiktimmar i grundskolan kan jag inte riktigt uttala mig om, men jag känner ändå att jag vill ifrågasätta huruvida det verkligen är antal lektionstimmar det brister på? Är det inte lusten som att lära som snarare är kärnan?

39 Hur väl de fungerar och samfungerar med andra program torde kunna lämnas till den som efterfrågar programmen.

7 Material och resurser

Detta kapitel innehåller material för att lära ut ett matematiskt innehåll med hjälp av programmering. Det första och längsta kapitlet består huvudsakligen av programmeringsuppgifter från läromedel i matematik. Godkännande från utgivarna har inhämtats för alla scannade delar från dessa läromedel.

7.1 Algoritmprogrammering på grafritande miniräknare

Introduktion till programmering på Texas Instruments miniräknare av Sofia Bäckström:
http://www.bäckströms.com/texas/pg.pdf

7.1.1 Matematik 1, Enkel sannolikhet

Från läromedlet Diskret matematik för gymnasiet (Wallin, Axelsson, G. Jakobsson, L. Jakobsson, & Nilson, 2002, s 97):

Man kan göra ett program på räknaren som simulerar ett antal (maximalt 999 kast) tärningskast och anger i hur stor andel av fallen som tärning 2 visar mer än tärning 1. När du startar nedanstående program ser du i räknarens fönster texten "ANTAL DUB-KAST?".Skriv då in hur många kast med två tärningar som du vill utföra.

```
PROGRAM:DICES
ClrHome
Input "ANTAL DUB-KAST ",N
randInt(1,6,N) → L1
randInt(1,6,N) → L2
0 → S
For(J,1,N)
If L2(J)>L1(J)
End
Disp "TARN2 MER ANDEL"
Disp S/N
```

UPPGIFTER

4003 Du kastar två tärningar.
Hur stor är sannolikheten
a) att siffersumman blir 6?
b) att siffersumman blir högst 6?
c) att siffersumman blir åtminstone 4?
d) att en av tärningarna visar 3?
e) att minst en av tärningarna visar 3?
f) att ingen av tärningarna visar 3?

4004 Utför beräkningen av uppgift 4003 genom simulering på räknaren

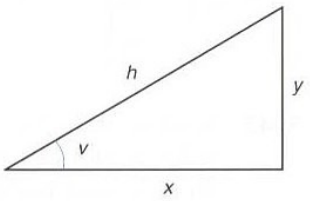
4005 * Ändra i programmet DICES så att du i stället undersöker relativa frekvensen för att tärningarna visar lika antal prickar. Upprepa försöket och jämför med det teoretiskt funna värdet.

7.1.2 Matematik 1b-1c, Trigonometri

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 90):

3056 ✳ Skriv ett program vilket frågar efter vinkeln v och längden på hypotenusan. Sedan beräknar programmet sidorna x och y.

```
prgmKATETER
VINKEL: 25
HYPOTENUSA: 10
X=
      9.06307787
Y=
      4.226182617
```



3057 ✳✳ Skriv ett program som frågar efter längden på sidorna i en rätvinklig triangel. Om användaren anger värdet 0 för en sida så beräknar programmet längden på denna sida.

```
prgmSIDOR
KATET 1: 5
KATET 2: 0
HYPOTENUSA: 13
SIDA=
                12
              Done
```

7.1.3 Matematik 1b-1c, Primtal, Eratosthenes såll

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, ss 43–44):

Erathostenes såll

Med hjälp av "Erathostenes såll" kan man generera mängden av primtal i intervallet $[2, n]$.

Så här fungerar Erathostenes såll:

Räkna upp samtliga heltal i aktuellt intervall, t.ex [2, 30]:

2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19
20 21 22 23 24 25 26 27 28 29 30

Välj det första talet i uppräkningen (2) och stryk sedan samtliga multipler av talet (alla jämna tal):

② 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ 9 ~~10~~ 11 ~~12~~ 13 ~~14~~ 15 ~~16~~ 17 ~~18~~ 19
~~20~~ 21 ~~22~~ 23 ~~24~~ 25 ~~26~~ 27 ~~28~~ 29 ~~30~~

Upprepa med nästa icke strukna tal i uppräkningen (talet 3) och stryk multipler:

2 ③ 5 7 ~~9~~ 11 13 ~~15~~ 17 19 ~~21~~ 23 25 ~~27~~ 29

Upprepa med nästa icke strukna tal, stryk multipler. Fortsätt detta förfarande tills nästa icke strukna tal är större än $\sqrt{30} \approx 5{,}5$:

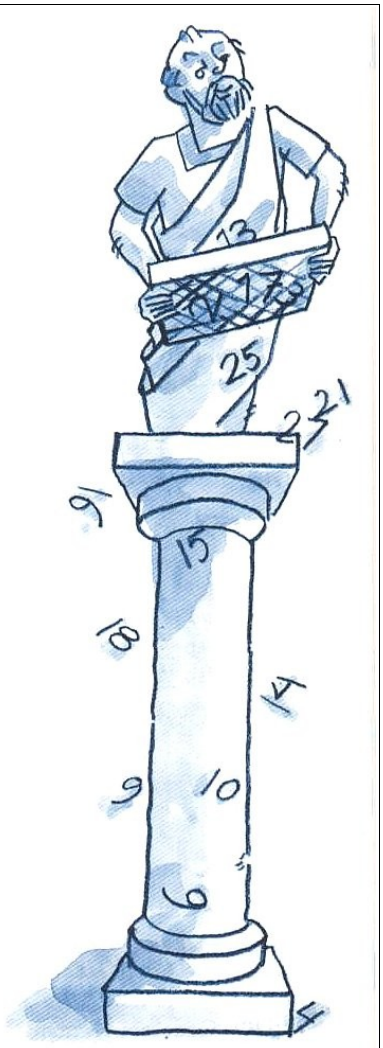
2 3 ⑤ 7 11 13 17 19 23 ~~25~~ 29

Nästa tal, dvs 7, är inte större än 5,5 och man kan därmed visa att alla tal som inte är primtal är bortsorterade.

Program för Erathostenes såll på räknaren

Ett programförslag till räknaren TI83 för Erathostenes såll finner du nedan. Programmet beskrivs här kortfattat:

PROGRAM:ERATOST	
ClrHome	Rensa skärmen
Input "OVRE GRANS: ",N	Mata in övre gräns
N →dim(L2)	Skapa plats för N st tal i lista 2
0 →L2(1)	Lagra 0 (false) i element 1 i lista 2
For(J,2,N,1)	Upprepa för 2 till N i steg om 1
1 →L2(J)	Lagra 1 i element J i lista 2
End	Slut på For-loop



Fortsättning på nästa sida...

√(N) → Q	Lagra $\sqrt{N}$ i minne Q
2 → P	lagra 2 i minne P
While P≤ Q	Så länge P är ≤ Q
For(J,2*P,N,P)	För J från 2·P till N i steg om P
0 → L2(J)	lagra 0 (false) i L2 (stryk alla tal som är delbara med P)
End	Slut på For-loop
P+1 → P	Öka P med 1 och lagra i P
While L2(P)=0 and P≤ Q	Så länge L2(P) = 0 och P ≤ Q
P+1 → P	Öka P med 1 och lagra i P
End	Slut While (L2(P) = 0)
End	Slut While (P ≤ Q)
0 → dim(L1)	Lagra 0 som längd på lista L1 (den kan då utökas efterhand)
1 → P	lagra 1 i P
For(J,1,N,1)	För J från 1 till N i steg om 1
If L2(J) > 0	Om L2(J) > 0 (dvs. talet är märkt True)
Then	Så
J → L1(P)	Lagra värdet J i lista L1 (element nr P)
P+1 → P	Öka P med 1
End	Slut If
End	Slut For
Disp "PRIMTALEN FINNS	Utskrift
Disp "I LISTA L1"	

UPPGIFTER

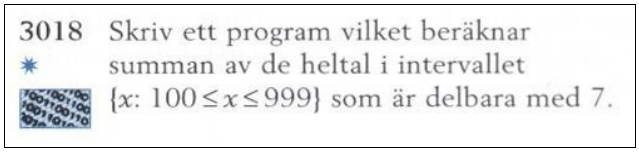
2030 Generera en mängd av samtliga primtal i intervallet [2, 100] på din räknare. Du kan använda programmet ovan.

2031 Beräkna antalet primtal i intervallet [2, 500].

2032 ✷ Gör ett program som beräknar antalet primtalstvillingar i lista L1. En primtalstvilling är två udda tal efter varandra som båda är primtal, t.ex 11 och 13 eller 17 och 19.
Beräkna sedan antalet tvillingar i intervallet [2, 500].

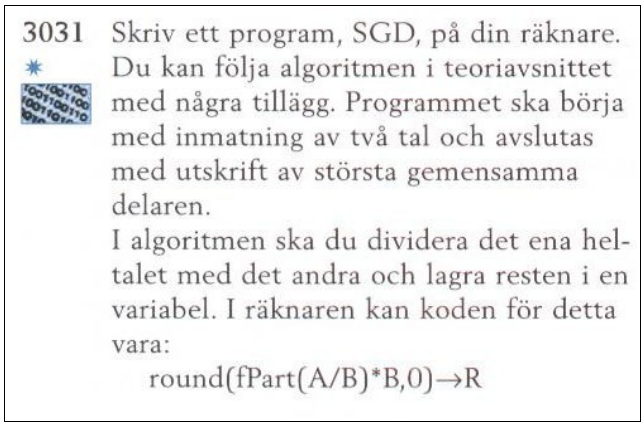
7.1.4 Matematik 1b-1c, Delbarhet

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 75):



3018 ✷ Skriv ett program vilket beräknar summan av de heltal i intervallet $\{x: 100 \leq x \leq 999\}$ som är delbara med 7.

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 82):

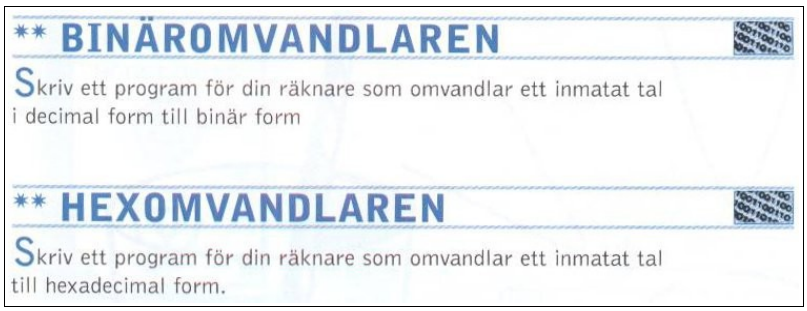


3031 ✷ Skriv ett program, SGD, på din räknare. Du kan följa algoritmen i teoriavsnittet med några tillägg. Programmet ska börja med inmatning av två tal och avslutas med utskrift av största gemensamma delaren.
I algoritmen ska du dividera det ena heltalet med det andra och lagra resten i en variabel. I räknaren kan koden för detta vara:

round(fPart(A/B)*B,0)→R

7.1.5 Matematik 1b-1c, Talbaser, Talbasomvandlare

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 61):



** BINÄROMVANDLAREN

Skriv ett program för din räknare som omvandlar ett inmatat tal i decimal form till binär form

** HEXOMVANDLAREN

Skriv ett program för din räknare som omvandlar ett inmatat tal till hexadecimal form.

7.1.6 Matematik 2, Andragradsekvationer

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 80):

Lösning av andragradsekvationer

Om man vill lösa ekvationen $ax^2+bx+c=0$ kan man använda lösningsformeln för andragradsekvationer. Man får:

$ax^2+bx+c=0$ Leden divideras med a $(a\neq 0)$

$$x^2+\frac{b}{a}x+\frac{c}{a}=0$$

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

Man måste ta hänsyn till ett par fall innan man använder formeln:

- Om $a=0$ så är det ej en andragradsekvation
- Om dessutom $b=0$ så är det ej en ekvation
- Om $b^2-4ac<0$ så har ekvationen inga reella lösningar

En algoritm i halvkod kan se ut så här:

om $a\neq 0$ så (andragradsekvation)
  om $b^2-4ac\geq 0$ så (ekvation med reella rötter)
    $x_1=\frac{-b-\sqrt{b^2-4ac}}{2a}$
    $x_2=\frac{-b+\sqrt{b^2-4ac}}{2a}$
  annars (ej några reella rötter)
    Ekvationen har ej några reella rötter
annars om $b\neq 0$ så (förstagradsekvation)
  $x=-c/b$
annars (ej en ekvation)
  Uttrycket är ej en ekvation ($a=0$ och $b=0$)

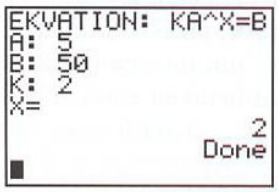
```
PROGRAM: EKVATION
Disp "AX²+BX+C=0"
Input "A:",A                                  Mata in A
Input "B:",B                                  Mata in B
Input "C:",C                                  Mata in C
If A≠0                                        Om A≠0 så andragradsekvation
Then
If B²–4AC≥0                                   Om B²–4AC≥0 så lösbar
Then                                          Utskrift av lösningar
Disp "X= ",(-B–√(B²–4AC))/(2A),(-B+√(B²–4AC))/(2A)
Else                                          Annars (då B²–4AC<0)
Disp "INGA LOSNINGAR"                         Inga (reella) lösningar
End
Else
If B≠0                                        Om B≠0 (och A=0) så förstagradsekv.
Then
Disp "X=",–C/B                                Utskrift av lösning
Else                                          Annars (då A=0 och B=0)
Disp "EJ EKVATION"                            ej ekvation
End
End
```

7.1.7 Matematik 2, Exponentialekvationer

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 90):

3054 Skriv ett program som löser ekvationen $k \cdot a^x = b$. Lösning på ekvationen ges av $x = \frac{\ln(b/k)}{\ln a}$. Lös sedan följande ekvationer:

a) $4 \cdot 2^x = 1024$ b) $100 \cdot 1{,}08^x = 200$
c) $-5 \cdot 3^x = 15$ d) $5 \cdot 3^x = -15$



7.1.8 Matematik 3, Aritmetisk och geometrisk summa

7.1.8.1 Sofia Bäckström

http://www.bäckströms.com/texas/progdel2.docx

7.1.8.2 Diskret matematik för gymnasiet

Från läromedlet Diskret matematik för gymnasiet (Wallin m.fl., 2002, s 75):

3016 Skriv följande program på din räknare. Vad betyder utskriften som programmet ger?

```
1→A
0→S
While (A≤100)
S+A→S
A+1→A
End
Disp S
```

3017 Skriv ett program som bestämmer summan av kvadraterna av de 100 första positiva heltalen, dvs.
$1^2 + 2^2 + 3^2 + 4^2 + \ldots + 99^2 + 100^2$

7.1.9 Matematik 3-4, Integraler, Rektangel- och trapetsmetoden

7.1.9.1 Delta

Från läromedlet Delta (Björup, Oscar, & Sandhall, 2001, s 132):

Om du har en programmerbar miniräknare kan du lätt skriva ett program som beräknar integraler numeriskt med rektangelmetoden respektive trapetsmetoden. I marginalen finns en skiss till programmet för rektangelmetoden och nedan för trapetsmetoden.

Programmet för trapetsmetoden bör arbeta på följande sätt:

1) fråga efter integrationsgränserna a och b och antal delintervall n
2) beräkna steglängden h
3) lägg $f(a)+f(b)$ i ett minne A
4) upprepa $n-1$ gånger
   a) öka a med h
   b) addera $2f(a)$ till minne A
5) multiplicera innehållet i minne A med $h/2$

Använd programmen för att jämföra effektiviteten hos de två metoderna. Det kan du naturligtvis också göra genom att använda något datorprogram som beräknar integraler med olika metoder och där man fritt kan välja steglängden.

Det är inte svårt att skriva ett program för rektangelmetoden för din programmerbara miniräknare. Försök att göra det enligt följande skiss.

1) fråga efter integrationsgränserna a och b och antal delintervall n

2) beräkna $h = \frac{b-a}{n}$

3) sätt $x = a + h/2$ och $A = 0$

4) upprepa följande n gånger

a) öka A med $f(x)$

b) öka x med h

5) multiplicera A med h

6) skriv ut närmevärdet A

7.1.9.2 Matematik 4000

Från läromedlet Matematik 4000 (Alfredsson, Erixon, Heikne, & Palbom, 2009a, s 371):

7414 *För dig som kan programmera:*

Skriv ett program för trapetsmetoden enligt följande stegvisa beskrivning (algoritm), där a och b är integrationsgränserna, n antalet intervall, w intervallbredden och T trapetssumman.

1 Fråga efter a, b och n.
2 $(b-a)/n \rightarrow w$
3 $T = 0$
4 Upprepa för $k = 0$ till $n-1$.
5 $T + w \cdot (f(a + k \cdot w) +$
$+ f(a + (k+1) \cdot w))/2 \rightarrow T$
6 Slut på upprepningen.
7 Skriv "T = ", T
8 Slut

a) Testa programmet på integralerna

$$\int_0^2 4e^x\,dx = 4e^2 - 4 \quad \text{och} \quad \int_0^{\pi} \sin x\,dx = 2$$

Det exakta svaret är angivet för att du ska kunna kontrollera ditt program.

b) Ändra programmet så att du för givna a och b kan välja olika n-värden. Avbryt med $n = 0$.

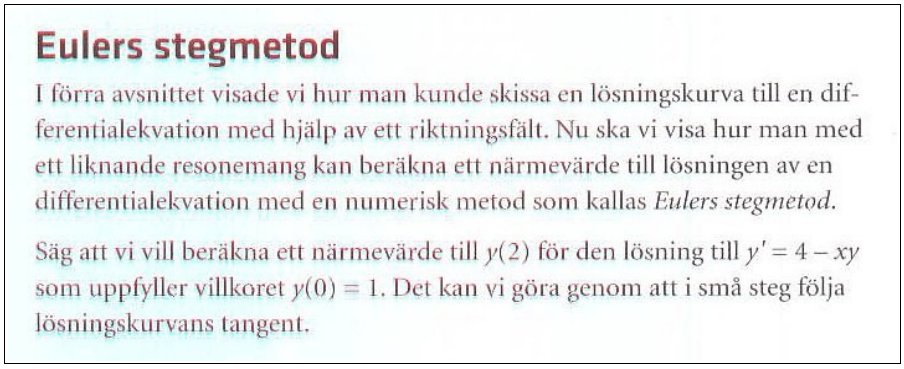
c) Undersök hur felet förändras då du dubblar antalet intervall.

d) Gör en egen enklare algoritm.

7.1.10 Matematik 4-5, Differentialekvationer, Eulers stegmetod

7.1.10.1 Origo

Från läromedlet Origo (Szabo, Larson, Viklund, & Marklund, 2009, ss 93–94):



Eulers stegmetod

I förra avsnittet visade vi hur man kunde skissa en lösningskurva till en differentialekvation med hjälp av ett riktningsfält. Nu ska vi visa hur man med ett liknande resonemang kan beräkna ett närmevärde till lösningen av en differentialekvation med en numerisk metod som kallas *Eulers stegmetod*.

Säg att vi vill beräkna ett närmevärde till $y(2)$ för den lösning till $y' = 4 - xy$ som uppfyller villkoret $y(0) = 1$. Det kan vi göra genom att i små steg följa lösningskurvans tangent.

Fortsättning på nästa sida...

Lutning $\frac{\Delta y}{\Delta x}$ betyder för en rät linje att om du tar Δx steg åt höger, så ska du ta Δy steg uppåt för att åter komma till den räta linjen. Om lutningen är 4, så kommer vi 2 steg uppåt om vi tar 0,5 steg åt höger.

Vi startar i punkten $(x_0, y_0) = (0, 1)$, som är det villkor lösningen ska uppfylla. I den punkten är $y' = 4 - xy = 4 - 0 \cdot 1 = 4$.

Det betyder att tangenten till lösningskurvan har lutningen 4 i punkten (0, 1). Följer vi lösningskurvan 0,5 steg till höger i tangentens riktning, så kommer vi till funktionsvärdet $y_1 = 0{,}5 \cdot 4 + 1 = 3$.

Se figuren här intill. Eftersom vi tagit 0,5 steg åt höger, så befinner vi oss nu i punkten (0,5; 3).

Nu utgår vi från den nya punkten på lösningskurvan och tar ytterligare 0,5 steg till höger.

I punkten (0,5; 3) är lutningen
$y' = 4 - xy = 4 - 0{,}5 \cdot 3 = 2{,}5$

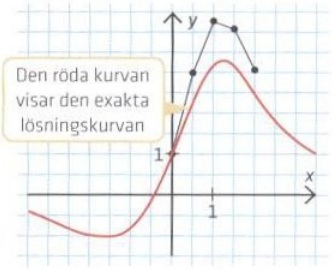
och vi får på samma sätt som här ovanför
$y_2 = 0{,}5 \cdot 2{,}5 + 4 = 4{,}25$

$x_2 = x_1 + 0{,}5 = 0{,}5 + 0{,}5 = 1$

I punkten (1; 4,25) är
$y' = 4 - 1 \cdot 4{,}25 = -0{,}25$ och
$y_3 = 0{,}5 \cdot (-0{,}25) + 4{,}25 = 4{,}125$

I punkten (1,5; 4,125) är
$y' = 4 - 1{,}5 \cdot 4{,}125 = -2{,}1875$

Och slutligen får vi
$y_4 = 0{,}5 \cdot (-2{,}1875) + 4{,}125 = 3{,}03125$



som alltså är ett närmevärde till $y(2)$ för den lösning till $y' = 4 - xy$ som uppfyller villkoret $y(0) = 1$.

I resonemanget här ovanför har vi använt steglängden 0,5. Det brukar betecknas som $h = 0{,}5$. Ju fler och därmed kortare steg man tar desto bättre blir närmevärdet.

Vi kan bestämma en rekursiv formel för att med Eulers stegmetod lösa differentialekvationer av första ordningen. Om man utgår från punkten (x_n, y_n) på lösningskurvan och om man följer resonemanget här ovanför, så kan vi approximativt bestämma punkten (x_{n+1}, y_{n+1}) med

$y_{n+1} \approx y_n + h \cdot y'_n$

Här är $x_n = x_0 + h \cdot n$ och derivatan y'_n beräknas i punkten (x_n, y_n).

Med hjälp av formeln kan man programmera sin räknare att lösa differentialekvationer av första ordningen. I marginalen här intill hittar du ett förslag på ett räknarprogram.

```
PROGRAM:STEGMET
:INPUT "X-START=",X
:INPUT "Y-START=",Y
:INPUT "H=",H
:INPUT "X-SLUT=",S
:ClrList L1,L2
:X→L1(1)
:Y→L2(1)
:1→A
:While L1(A)<S
:L1(A)+H→L1(A+1)
:L2(A)+H*Y1(X)→L2(A+1)
:A+1→A
:L1(A) →X
:L2(A) →Y
:End
:FnOff
:PlotsOff
:Plot1(Scatter,L1,L2,·)
:ZoomStat
```

Formel för Eulers stegmetod

Börja med att skriva in programmet. Du beräknar sedan ett närmevärde till lösningen av $y' = 4 - xy$, genom att trycka [Y=] och skriva in högerledet till differentialekvationen efter Y1=. När du kör programmet läser du in begynnelsevillkor och steglängd genom att besvara de frågor som dyker upp på skärmen. Lösningen läser du av i grafen med [TRACE].

7.1.10.2 Optima

Från läromedlet Optima (Axelsson m.fl., 2002, ss 72–73):

Eulers metod med programmerbar räknare

Med stora steglängder i Eulers metod avlägsnar sig den lösningskurva man får raskt från den riktiga. Felen är ofta *kumulativa*, dvs. de läggs på varandra för varje steg. Det gäller alltså att använda små steglängder, Δx. Detta ger omfattande räkningar och man utnyttjar därför datorer för beräkningarna.

Vi kan komma långt även med en programmerbar räknare. Programmet nedan, som är exemplifierat för *Texas TI-83* men enkelt kan översättas till andra räknare, löser differentialekvationer med Eulers metod.

PROGRAM: EULER

1 Rensa skärmen	: ClrHome
2 Ange startvärde för x, dvs. x_0	: Input "X-START?", X
3 Ange startvärde för y, dvs. y_0	: Input "Y-START?", Y
4 Ange steglängd, dvs. Δx	: Input "STEG?", S
5 Ange slutvärde för x, dvs. x_{slut}	: Input "SLUT-X?", Z
6 Töm listor för lagring av funktionsvärden	: ClrList L_1,L_2
7 Lagra x_0 som första element i lista 1	: X STO L_1(1)
8 Lagra y_0 som första element i lista 2	: Y STO L_2(1)
9 Lagra talet 1 i en indexräknare	: 1 STO A
10 Så länge som $x < x_{slut}$ så	: WHILE L_1(A) < Z
11 Öka x med Δx och lagra som nästa element i lista 1	: L_1(A) + S STO L_1(A + 1)
12 Öka y med $\Delta y = \Delta x \cdot y'$ och lagra som nästa element i lista 2	: L_2(A) + S * Y_1(X) STO L_2(A + 1)
13 Öka indexräknaren med 1	: A+1 STO A
14 Kopiera listelement som innehåller senaste y-värde till minne Y	: L_2(A) STO Y
15 Kopiera listelement som innehåller senaste x-värde till minne X	: L_1(A) STO X
16 Avsluta "så länge som"-loopen	: End
17 Stäng av funktioner	: FnOff
18 Stäng av statistiska plottar	: PlotsOff
19 Initiera ritning av lista 2 som funktion av lista 1	: Plot1(Scatter,L_1,L_2, •)
20 Utför ritandet i ett anpassat fönster	: ZoomStat

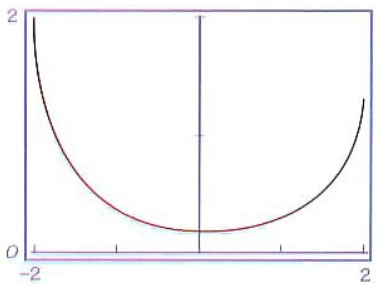
Fortsättning på nästa sida...

För att kunna använda programmet definieras den aktuella funktionen y' som Y_1 i funktionseditorn. Då ska X användas för x och Y för y. Skriv in programmet i räknaren. Avaktivera funktionen Y_1 så att den inte kommer att ritas. Vi provkör det på differentialekvationen $y' = xy$ som vi tidigare behandlat. Funktionen definieras då alltså som $Y_1 = Y \cdot X$.

Man kör programmet genom att trycka på PRGM-tangenten och välja programmet EULER. Pröva med olika steglängder, t.ex. $\Delta x = 0{,}2$. Men observera att Δx inte kan göras hur litet som helst, eftersom listan är begränsad till ett visst antal element.

Kör programmet med samma begynnelse- och slutvärden som tidigare och jämför dina värden med tabellen på s. 71. Med hjälp av TRACE kan du kontrollera dina värden mot tabellens. Det går också bra att använda STAT, EDIT, som visar listan med x- och y-värdena direkt.

I figuren redovisas en lösningskurva till differentialekvationen $y' = xy$ i intervallet $[-2, 2]$. Lösningskurvan går genom $(-2; 2)$ och steglängden är 0,1.



Differentialekvationen $y' = xy$ går att lösa exakt. Lösningen är $y = 2e^{\frac{x^2}{2} - 2}$ om lösningskurvan ska gå genom punkten $(-2; 2)$. Jämför utseendet av denna kurva med den lösning vi tagit fram numeriskt genom att rita båda i samma fönster.

Om man vill stega mot mindre x ska steglängden vara negativ. Då måste man vända på olikhetstecknet i avbrottsvillkoret WHILE.

7.1.10.3 Matematik 4000

Från läromedlet Matematik 4000 (Alfredsson, Erixon, Heikne, & Palbom, 2009b, s 199):

Program 2 beräknar med Eulers stegmetod ett närmevärde till det sökta y-värdet och ritar motsvarande stegkurva. Övergången från kurva till beräknade värden sker med ENTER. Du kan minska steglängden, t ex genom upprepad fördubbling av antalet steg N, tills den sista värdesiffran inte ändras längre.

```
Program:DIFFEU
FnOff :PlotsOff
ClrDraw:ClrHome
Input "START X =" ,A
Input "START Y =" ,C
Input "SLUT X =" ,B
Lbl 1
Input "N=" ,N
(B−A)/N→H
A→X:C→Y
For(I ,1 ,N)
X→U:Y→V
Y+HY1→T
X+H→S
Line(U ,V ,S ,T)
S→X:T→Y
End
Pause
Disp" GER Y=" ,Y
ClrDraw
Goto 1
```

7.1.11 Matematik 4-5, Differentialekvationer, Riktningsfält

Från läromedlet Matematik 4000 (Alfredsson, Erixon, Heikne, & Palbom, 2009b, s 199):

Program 1 ritar ett riktningsfält i $12 \times 8 = 96$ punkter för den valda fönsterrektangeln.

```
Program:DIFFRF
FnOff :PlotsOff :ClrDraw
7(Xmax − Xmin)/83→H
7(Ymax − Ymin)/55→K
1/(0.4H)²→A
1/(0.4K)²→B
Xmin+H/2→X
For(I ,1 ,12)
Ymin+K/2→Y
For(J,1,8)
Y1→T
1/√(A + BT²)→C
X→U:Y→V
Line(U−C ,V−CT ,U+C ,V+CT)
V+K→Y
End
U+H→X
End
```

7.1.12 Matematik 4-5, Differentialekvationer, Runge-Kuttas stegmetod

Från läromedlet Matematik 4000 (Alfredsson, Erixon, Heikne, & Palbom, 2009b, s 199):

Program 3 är en kopia av program 2 men använder istället Runge-Kuttas stegmetod (se Stegmetoder s 168). Skriv av program 2, men byt ut det blåskuggade området i program 2 mot det blåskuggade nedan.
(Program:DIFFRK)

```
Y1→P
X+H/2→X
V+HP/2→Y
Y1→0
V+HQ/2→Y
Y1→R
X+H/2→X
V+HR→Y
Y1→K
X→S
V+H(P+2Q+2R+K)/6→T
```

7.2 Webbprogrammerings kurs för Matematik A

Ny version:
http://angeredswebben.se/webbbok/version5/

Beräknad tidsåtgång: ca 20-40 timmar (Bäckström, 2011a).

> *”Jag har svårt att komma på någon matematiskt begrepp från kurserna A och B som det inte finns solklara tillämpningar av i en spelmotor. //det skulle vara stapeldiagram i s å fall...” (Bäckström, 2011a)*

7.3 Project Euler

Ursprung: Internationellt

Webbsida: http://projecteuler.net/

> *”Project Euler is a series of challenging mathematical/computer programming problems that will require more than just mathematical insights to solve. Although mathematics will help you arrive at elegant and efficient methods, the use of a computer and programming skills will be required to solve most problems.” (Project Euler, 2011)*

7.4 Bootstrap

Ursprung: USA

Webbsida: http://www.bootstrapworld.org/

> *”Bootstrap is a standards-based curriculum for middle and high-school students, which teaches them to program their own videogames using purely algebraic and geometric concepts. Bootstrap is used by educators around the country to bring*

> *technology and real math instruction together in the classroom." (Bootstrap, 2011a)*

De samarbetar med ett antal organisationer som tillhandahåller stöd och aktiviteter efter skolan, med fokus på barn från familjer med låg inkomst, och anordnar även sommarläger. Fokus ligger på mellan- och högstadieelever (Bootstrap, 2011b). Bootstrap använder sig av "Program by Design"[40].

7.5 Material som användes för presentationen på matematikbiennalen 2009

http://bäckströms.com/wgm/

7.6 LOGO Turtle programmering på webben

En av de intervjuade lärarna menade att Turtle graphics var en bra introduktion till programmering. Det går ut på att man har en tänkt sköldpadda som går omkring på skärmen och ritar. Programmeringen består i att man ger sköldpaddan en sekvens med instruktioner för hur han ska rita, röra sig och vända sig. Han ansåg att man inte behövde några förkunskaper för att börja programmera i Turtle graphics och rekommenderade Web Turtle för att enklast komma igång med detta: http://sonic.net/~nbs/webturtle/ (L8, 2011).

7.7 Exempel på matematiklaborationer i Programmering A

(Westh, 2011)

4_2

Markus är ute och kör bil. Tanken rymmer 50 liter. Priset per liter är 13,50 kr.

Skriv ett program som frågar hur mycket bensin som finns i tanken.
Om mängden är mindre än 10 liter ska programmet presentera en utskrift där det framgår att han måste tanka, hur mycket han ska fylla på och vad det kostar att få full tank.
Är mängden minst 10 liter innehåller utskriften bara en uppmaning att köra vidare.

5

Du ska skapa en enkel miniräknare enligt följande med hjälp av switch - case:

Läs in tal1, räknesätt och tal2. //tal kan vara decimaltal

Om räknesättet är +
    skriv ut summan
annars om räknesättet är –
    skriv ut differensen
annars om räknesättet är *
    skriv ut produkten
annars om räknesättet är /
    skriv ut kvoten
annars

40 http://programbydesign.org/

meddela ”felinmatning”

Lab 8_2

Skapa ett program där du matar in radien på en cirkel.

Programmet skall sedan räkna ut både area och omkrets på cirkeln. Till beräkningarna skall du använda dig av metoder, en för areaberäkningen och en för beräkningen av omkretsen. Utskriften sker sen i main.

Tips: runda gärna av till två decimaler med Math.Round().

Lab 8_3

Skriv en metod som avgör om ett tal är primtal eller ej. Ett primtal är ett tal som bara är delbart med sig själv eller 1. Primtal under 100:
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97

Läs in ett tal

Anropa metoden Primtal

I metoden bool Primtal(int tal) undersöks om tal är ett primtal eller ej.

Om tal är ett primtal returneras true

Annars returneras false

I Main skivs det ut om talet var ett primtal eller ej.

Skjuta ner burkar (Linjära funktioner)

Spelet går ut på att man ska skjuta ner ett antal burkar genom att ange lutningen på en rät linje (y=kx+m).

Pistolens mynning finns i ett tänkt origo och burkarna ligger utplacerade på ett fast y-värde. Varje burk upptar ett område mellan två x-värden.

Då man ska skjuta anger man k (lutningen) och programmet räknar ut om man får en träff.

Man får skjuta t.ex. 10 skott.

Vill man kan man utöka programmet med att lägga burkarna på olika y-värden samt be användaren mata in vilket m-värde man skjuter från. Man kan slumpa fram positionerna eller ange dem ”fast” i programmet.

7.8 Material för kursen Matematik – breddning

Se kap 5.2.2.4 för mer info.

Material för kursen Matematik – breddning:
http://www.labutb.falun.se/new/?id=matte-breddning

7.9 Turingmaskiner

En av de intervjuade lärarna funderade på att låta eleverna bygga enkla turingmaskiner[41] med papper och tejp, till exempel räkna upp alla heltal, plus, minus, mm. Eleverna skulle sedan få pröva varandras turingmaskiner och försöka vilka operationer som utfördes av de olika turingmaskinerna (L5, 2011b). Detta för att träna tänkande inom både matematik och programmering.

7.10 Sierpinskitriangel i DECIMAL Basic[42]

Se kap 5.2.2.3.

```
SET WINDOW 0, 100, 0, 100
LET xa=0
LET xb=50
LET xc=100
LET ya=0
LET yb=100
LET yc=0
SET POINT STYLE 1
FOR i = 1 TO 50000
LET r=RND*3
IF r<1 THEN
SET POINT COLOR (1)
LET xx=(xx+xa)/2
LET yy=(yy+ya)/2
ELSEIF r<2 THEN
SET POINT COLOR (2)
LET xx=(xx+xb)/2
LET yy=(yy+yb)/2
ELSE
SET POINT COLOR (3)
LET xx=(xx+xc)/2
LET yy=(yy+yc)/2
END IF
PLOT POINTS: xx,yy
NEXT i
END
```

(Lindholm, 2011)

7.11 Övrigt halv relaterat material

7.11.1 Material för användning av grafritande miniräknare

http://www.bäckströms.com/texas/

7.11.2 Material för programmeringskurser

- Introduktionskurs i C# för programmering A:
  http://angeredswebben.se/spelutveckling/csharpintro/index.php

41 http://sv.wikipedia.org/wiki/Turingmaskin
42 http://hp.vector.co.jp/authors/VA008683/english/

- Minikurs i programmering av en spelmotor i C# med DirectX (exempelvis programmering B):
  http://bäckströms.com/cv/spelmotorkurs/del1/index.htm
  Beräknad tidsåtgång: Från några veckor (Bäckström, 2011b)
- Bibliotek i JavaScript för att förenkla spelprogrammering:
  https://github.com/datorklubben/Spelprogrammering-med-JavaScript-och-Canvas
- http://www.spelprogrammering.nu/
- Tankar från Lennart Rolandsson som forskar kring undervisning i programmering:
  http://www.newsdistro.nu/blogg/

8 Referenslista

Alfredsson, L., Erixon, P., Heikne, H., & Palbom, A. (2009a). *Matematik 4000 - Kurs C&D blå* (Första upplagan.). Stockholm: Natur & Kultur.

Alfredsson, L., Erixon, P., Heikne, H., & Palbom, A. (2009b). *Matematik 4000 - Kurs E blå* (Första upplagan.). Stockholm: Natur & Kultur.

Axelsson, R., Bratt, H., Jakobsson, G., Jakobsson, L., Nilson, K., & Sikö, Arne. (2002). *Optima E* (3:1 uppl.). Malmö: Liber.

Back, R.-J. (2006). *Invariant Based Programming*. Åbo, Finland: Åbo Akademi University, Turku, Finland.

Back, R.-J. (2008). *Matematik med lite logik - Introduktion till strukturerade härledningar*. TUCS Lecture Notes - IMPEd Series. TUCS (Turku Centre for Computer Science) & IMPEd (Improving Mathematics and Programming Education).

Björup, K., Oscar, E., & Sandhall, Å. (2001). *Nya Delta - Matematik för gymnasiet, naturvetenskapligt och tekniskt program, kurs D* (Andra upplagan.). Malmö: Gleerups.

Bootstrap. (2011a). Bootstrap. Hämtad November 22, 2011, från http://www.bootstrapworld.org/

Bootstrap. (2011b). Bootstrap: Take a Course. Hämtad November 22, 2011, från http://www.bootstrapworld.org/learn/

Bruce, K. B., Drysdale, R. L. S., Kelemen, C., & Tucker, A. (2003). Why math? *Communications of the ACM*, *46*(9), 40-44.

Bäckström, S. (2009). Web och spel programmering. Hämtad December 13, 2011, från http://xn--bckstrms-0za9p.com/wgm/bakgrund/

Bäckström, S. (2010). Matematiken i webbapplikationer. Hämtad December 13, 2011, från http://www.xn--bckstrms-0za9p.com/ihgr/#

Bäckström, S. (2011a, December 16). E-postbrev till Jeremia Mörling. Ämne: Examensarbete kring matematik och programmering.

Bäckström, S. (2011b, December 20). E-postbrev till Jeremia Mörling. Ämne: Examensarbete kring matematik och programmering.

Bäckström, S., & Rudenstam, M. (2010). *Slutrapport för projektet ”Införa webbprogrammering….”, ett uppdrag för .SEs Internetfond 2008*. Hämtad från https://www.iis.se/docs/Slutrapport_Webbprogrammering_i_gymnasieundervisningen.pdf

Claesson, S. (2009). *Lärares hållning* (Första upplagan.). Lund: Studentlitteratur.

Clements, D. H. (1985). Research on Logo in education: Is theturtle slow but steady, or not even in the race? *Computers in the Schools*, *2*(2/3), 55-71.

Clements, D. H. (1986). Effects of Logo and CAI Environmentson Cognition and Creativity. *Journal of Educational Psychology*, *78*(4), 309-318.

Clements, D. H., & Gullo, D. F. (1984). Effects of Computer Programming on Young Children's Cognition. *Journal of Educational Psychology*, *76*(6), 1051-1058.

Dewey, J. (1997). *Demokrati och utbildning* (Första upplagan.). Göteborg: Daidalos.

Egeberg, C. (2007, Augusti 23). Uppgifter i QBasic. Polhemsgymnasiet.

Esaiasson, P., Gilljam, M., Oscarsson, H., & Wängnerud, L. (2007). *Metodpraktikan* (Tredje upplagan.). Stockholm: Norstedts Juridik.

Feurzeig, W., Horwitz, P., & Nickerson, R. S. (1981). *Microcomputers in education* (Prepared for: Department of Health, Education, and Welfare; National Institute of Education; and Ministry for the Development of Human Intelligence, Republic of Venezuela. Bolt Beranek & Newman No. 4798). Cambridge, MA, USA.

Hedrén, R. (1988). *KBI-projektet LOGO-programmering på mellanstadiet* (No. 1988:3). Falun-Borlänge: Högskolan i Falun-Borlänge.

Henderson, P. B. (2002). *Functional and Declarative Languages for Learning Discrete Mathematics*. Butler University, Indianapolis, USA. Hämtad från http://www.cs.geneseo.edu/~baldwin/math-thinking/phlanguages.pdf

Henderson, P. B. (2003). Mathematical Reasoning in Software Engineering Education. *Communications of the ACM*, *46*(9), 45-50.

IMPEd. (2011). IMPEd - Improving Mathematics and Programming Education. Hämtad November 22, 2011, från http://www.imped.fi/wordpress/?lang=se

Integrating mathematical thinking in computing curricula. (2011). Integrating mathematical thinking in computing curricula. Hämtad November 21, 2011, från http://www.math-in-cs.org/

Joint Mathematical Council of the United Kingdom. (2011). *Digital technologies and mathematics education*.

Juvel, S., & Hoppe, K. (2011, September 6). Björklund storsatsar på matte i skolan. *DN.se*. Hämtad från http://www.dn.se/nyheter/sverige/bjorklund-avslojar-budgetnyhet

Kairos Future. (2011a). Om Kairos Future. Hämtad November 30, 2011, från http://www.kairosfuture.com/om-oss

Kairos Future. (2011b). *IT och digital kompetens i skolan*. Hämtad från http://www.kairosfuture.com/sites/default/files/publications/Itiskolan.pdf

L1. (2011, November 25). Intervju av Jeremia Mörling. Ämne: Intervju om hur undervisning i matematik och programmering integreras. Telefon.

L10. (2011, December 15). E-postbrev till Jeremia Mörling. Ämne: Programmering i matematikundervisningen.

L11. (2011, November 28). E-postbrev till Jeremia Mörling. Ämne: Programmering inom matematikundervisningen.

L12. (2011, December 7). E-postbrev till Jeremia Mörling. Ämne: Programmering i matematikundervisningen.

L2. (2011, November 25). Intervju av Jeremia Mörling. Ämne: Intervju om hur undervisning i matematik och programmering integreras. Telefon.

L3. (2011, November 16). Intervjufrågor_svar_L3.

L4. (2011a, December 7). E-postbrev till Jeremia Mörling. Ämne: Re: Ang. Programmering inom matematikundervisningen.

L4. (2011b, November 23). Ämne: Intervju om hur undervisning i matematik och programmering integreras. Telefon.

L4. (2011c, November 23). E-postbrev till Jeremia Mörling. Ämne: Ang. Programmering inom matematikundervisningen.

L5. (2011a, November 18). Intervju av Jeremia Mörling. Ämne: Intervju om hur undervisning i matematik och programmering integreras.

L5. (2011b, November 22). E-postbrev till Jeremia Mörling. Ämne: Examensarbete kring matematik och programmering.

L6. (2011, December 6). Intervjufrågor.doc.

L7. (2011, December 6). E-postbrev till Jeremia Mörling. Ämne: Programmering i matematikundervisningen.

L8. (2011, November 14). Intervju av Jeremia Mörling. Ämne: Intervju om hur undervisning i matematik och programmering integreras. Telefon.

L9. (2010). Intervju av Jeremia Mörling. Ämne: VFU.

Lindh, J. (1993). *Datorstödd undervisning i skolan – möjligheter och problem*. Lund: Studentlitteratur.

Lindholm, J. (2011, December 12). E-postbrev till Jeremia Mörling. Ämne: Programmering inom matematikundervisningen.

Lochhead, J., & Clement, J. (1979). *Cognitiveprocess instruction: Research on teaching thinkingskills*. Philadelphia: Franklin Institute Press.

Mannila, L. (2009). *Teaching Mathematics and Programming- New Approaches with EmpiricalEvaluation*. Åbo, Finland: Åbo Akademi University, Turku, Finland. Hämtad från http://www.doria.fi/bitstream/handle/10024/50375/MannilaLinda.pdf?sequence=3

Nationalencyklopedin. (2011, Juni 9). Karta över Sverige. Nationalencyklopedin. Hämtad från http://media.ne.se/neimage/1072194.gif

Näslundh, C. (2008, Augusti 26). Spelutveckling - en metod i matematikundervisningen. *Skolverket*. Hämtad från http://www.skolverket.se/skolutveckling/itiskolan/reportage/matematik/spelutveckling-1.138377

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

Pea, R. D., & Kurland, M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology*, *2*(2), 137-168.

Pedersen, J. (1997). *Informationstekniken i skolan – en forskningsöversikt*. Stockholm: Skolverket.

Project Euler. (2011). Project Euler. Hämtad November 22, 2011, från http://projecteuler.net/

Python Software Foundation. (1997). Comparing Python to Other Languages. Hämtad December 20, 2011, från http://www.python.org/doc/essays/comparisons/

Python Software Foundation. (2011). What is Python? Executive Summary. Hämtad December 20, 2011, från http://www.python.org/doc/essays/blurb/

Sernhede, O. (2011). *Förorten, skolan och ungdomskulturen - Reproduktion av marginalitet och ungas informella lärande*. Göteborg: Daidalos.

Sfard, A. (1991). On the dual nature of mathematical conceptions - Reflections on processes and objects as different sides of the same coin. *Educational Studies in Mathematics*, *22*, 1-36.

Siring, M. (2011, September 19). *Elever i kris*. Föreläsning, Göteborgs Universitet, Pedagogen, Sal BE036.

Skolverket. (2003). *Lusten att lära - med fokus på matematik* (Nationella kvalitetsgranskningar 2001-2002 No. Rapport nr. 221). Stockholm: Skolverket. Hämtad från http://www.skolverket.se/2.3894/publicerat/2.5006?_xurl_=http%3A%2F%2Fwww4.skolverket.se%3A8080%2Fwtpub%2Fws%2Fskolbok%2Fwpubext%2Ftrycksak%2FBlob%2Fpdf1148.pdf%3Fk%3D1148

Skolverket. (2011). Ämne - Matematik. Hämtad från http://www.skolverket.se/forskola_och_skola/gymnasieutbildning/2.2954/amnesplaner_och_kurser_for_gymnasieskolan_2011/subject.htm?subjectCode=MAT

Smith, D. (2011, December 1). ”Angry Birds” and Wonderful Pistachios Launch New Game. *International Business Times*. Hämtad från http://www.ibtimes.com/articles/259583/20111201/angry-birds-wonderful-pistachios-launch-new-game.htm

Szabo, A., Larson, N., Viklund, G., & Marklund, M. (2009). *Origo kurs E för NV och TE* (Första upplagan.). Stockholm: Bonnier utbildning.

The Joint Task Force on Computing Curricula. (2001). *Computing Curricula 2001 - Computer Science*. Hämtad från http://www.acm.org/education/curric_vols/cc2001.pdf

The Opportunity Equation. (2009). The Report - The Opportunity Equation. Hämtad November 25, 2011, från http://opportunityequation.org/report

Urner, K. (2008). *Python for Math Teachers*. Hämtad från http://showmedo.com/videotutorials/series?id=101

Urner, K. (2010, Februari 13). Ämne: update from River City (Portland). Hämtad från http://mail.geneseo.edu/pipermail/math-thinking-l/2010-February/000541.html

Urner, K. (2011a). Kirby’s Autobiography. Hämtad November 21, 2011, från http://wikieducator.org/User:KirbyUrner/Autobio

Urner, K. (2011b, November 22). E-postbrev till Jeremia Mörling. Ämne: Programming within high school mathematics education.

US Department of Agriculture - Food and Nutrition Service. (2011, November 30). Free and Reduced Price Meal Benefit Information. Hämtad December 19, 2011, från http://www.fns.usda.gov/cnd/frp/frp.process.htm

Utbildnings- och forskningsnämnden vid Högskolan på Gotland. (2007). Matematik för programmering och spelutveckling, halvfart: Kursplan. Hämtad från https://portal.hgo.se/courses/mod/resource/view.php?id=63748

Wallin, H., Axelsson, R., Jakobsson, G., Jakobsson, L., & Nilson, K. (2002). *Diskret matematik för gymnasiet* (Första upplagan.). Malmö: Liber.

Westh, C. (2011, December 8). E-postbrev till Jeremia Mörling. Ämne: Programmering i matematikundervisningen.

Bilaga A: Grundmall för intervjuer

Jag heter Jeremia Mörling och studerar på korta lärarprogrammet på Göteborgs Universitet till gymnasielärare inom matematik och datavetenskap.

Examensarbetets syfte är att sammanställa information om hur man idag arbetar med programmering inom matematikundervisningen och hur man skulle kunna vidareutveckla detta. Jag tänker mig att uppsatsen ska kunna användas både av mig själv och andra som är intresserade av detta område som ett underlag för termins- och lektionsplanering.

Förkortningarna M (Matematik) och P (Programmering) används i frågorna nedan.

Syfte med att integrera M och P

1 Vad har du för **bakgrund**?
   1.1 Utbildning?
   1.2 Arbete?
2 **Hur kom du in på spåret** att integrera M o P?
3 Vem tog **initiativ** till integrationen?
4 Vad är ert **syfte med/orsaken till** att integrera M o P?
5 Har du några **förebilder** inom området?
6 Känner du till **andra lärare** som integrerar M o P?
   6.1 Kontaktuppgifter?

Vad man gör inom integrationen

7 **Vad görs** inom integrationen?
   7.1 Inom **vilka kurser** arbetar du med integreringen?
   7.2 Vilka **moment** inom kurserna?
   7.3 Hur är integrationen **organiserad**?
      7.3.1 Allt på en gång?
      7.3.2 Utspritt?
      7.3.3 Röd tråd?
8 Hur mycket **tid** har lagts på varje moment?
9 Vilken **förkunskap** krävs?
10 Krävdes det någon **startsträcka**?

Resultat av integrationen

11 Hur tycker du att **införandet** har gått?
   11.1 **Tekniskt**?

11.2 **Administrativt**, kollegor, arbetslag, mm?

12 Hur tycker du att **undervisningsresultatet** har blivit (jämfört med att inte integrera)?

12.1 **Fördelar**?

12.2 **Nackdelar**?

13 Tycker du att du har **förlorat tid** (jämfört med att inte integrera)?

Avslutning

14 Har du möjlighet att maila eventuell t**erminsplanering och lektionsplaneringar**?