# Lektion 1 (ca 60min)

Syfte och lärandeobjekt

Syftet med lektionen är att introducera programmeringsmiljön Scratch och utveckla algoritmiskt tänkande i programmering. Speciellt ska eleverna få en första erfarenhet av grundbegrepp inom programmering och hur dessa relaterar till block i Scratch genom att göra små förändringar i annars färdiga program.

Material

Skapa en läraranvändare[[1]](#footnote-1) på scratch och logga in. Gå till ”Mina klasser” och skapa en ny klass. Här har du möjlighet att samla dina elever, och skapa studios kopplat till enskilda lektioner. Det finns fyra saker du behöver göra innan första lektion:

1. Skapa en studio som heter Lektion 1
2. Lägga till de fyra projekt (Uppgift 1-3 och fördjupning) som är kopplat till uppgifterna[[2]](#footnote-2) i studion
3. Skapa en länk för eleverna att ansluta till klassen[[3]](#footnote-3)
4. Distribuera länken till eleverna på ett lämpligt vis, t.ex. om ni har en IT-plattform på skolan. Ha ditt Scratch-klassrum tillgängligt under hela lektionen, till exempel via en projektor eller interaktiv skrivtavla.

Lektionsbeskrivning

Ni kommer genomföra en tinkering-aktivitet i Scratch, eleverna får färdiga program som de ändrar värden i utifrån uppgiftsbeskrivningar. Eleverna får uppdrag där de ska ändra i befintlig kod utifrån olika uppdrag, och sedan köra koden för att se effekten på varje förändring.

**Eleverna har sin egen dator men de sitter i par. EPA modellen, de tar sig an uppdragen individuellt men med stöd av sin bänkkamrat. Varje uppgift följs upp av en kort helklassdiskussion som eleverna förbereder sig för i paren. Lämpliga frågeställningar finns på projektsidan till varje uppgift.**

Varje uppgift innehåller fördjupningsuppgifter, det är inte säkert att några elever hinner med dessa, eller att ni ska diskutera dessa i helklass. De ska ses som extra utmaning för elever som till exempel redan har jobbat med Scratch.

Lärartips: Uppmana eleverna att göra en ändring i taget och testa programmet. Var försiktig med att ge den rätta lösningen till eleverna under arbetet, ge istället eleverna möjlighet att prova sig fram. Uppmuntra eleverna att först kommunicera med sin arbetskamrat i stället för att ställa frågor till läraren.

**Introduktion**

Vi ska hjälpa åk 4-6 att träna huvudräkning, deras lärare önskar att det fanns något roligare sätt att göra detta på. Ni kommer vara programmerare med ett uppdrag. Ert uppdrag blir att skapa roliga sätt att träna ett eller flera utav de fyra räknesätten som passar elever i åk 4-6. Så ni ska alltså skapa digitala matematikspel som dessa elever kan använda när de övar. Men först, så ska vi repetera grundfunktioner och möjligheter i scratch, som är miljön vi kommer jobba i. Gå nu in på lärplattformen och klicka på länken, där skapar ni er Scratch-användare med ert förnamn som användarnamn.



1. Ladda programmet Geometriska former – Tid för denna del drygt 10 min (inklusive diskussion)

(finns inuti Scratch)

Testa att göra kvadraten större.

Testa att ändra i koden så det bildas en triangel.

Testa att göra en annan månghörning.

Fördjupning: Testa att göra en rätvinklig/likbent triangel.

Fördjupning: Testa att göra en diagonal i kvadraten.

Teman för den uppsamlande diskussionen (först par, sen alla):

Vilka svårigheter stötte ni på?

Spelar ordningen på blocken någon roll? Varför/varför inte?

Vad gör de Blåa och Gröna blocken? Var hittar man de blocken i Scratch?

*Vad?*

Eleverna ska förstå hur scratch-kod utförs stegvis, uppifrån och ner, och hur de blå blocken fungerar.

*Hur?*

Eleverna klickar på uppgift 1 i studion och sedan ”se inuti”. Där finns alla uppdrag, och koden som de ändrar i. När eleverna haft tid till att testa på de olika uppdragen ber du dem reflektera över reflektions/diskussionsfrågorna i paren (eleverna kan se diskussionsfrågorna på sina egna enheter om de klickar tillbaka till projektsidan). Samla sedan upp i helklassdiskussion. Det kommer antagligen vara elever som menar att triangeln är svår, eftersom det inte är helt intuitivt hur sprajten vänder sig i varje hörn. Om du planerar att hinna med denna lektionsplanering inom 60 minuter föreslår vi att ni sparar just den diskussionen till en annan lektion.

*Varför?*

Vi börjar med de blå blocken för att de är konkreta, både i sin funktion och vad som händer på skärmen. De får exemplifiera hur koden utförs stegvis uppifrån och ner. Vänta-blocken finns med för att förstärka den stegvisa strukturen, då det blir möjligt att koppla ihop varje rörelse som sker med ett block.

1. Ladda programmet Snöflingan - Tid för denna del knappt 10 min (inklusive diskussion)

Testa att ändra i koden så att den skapar något annat upprepande mönster.

Fördjupning: Googla fraktaler, testa att göra om koden så att den ritar någon typ av fraktal.

Teman för den uppsamlande diskussionen:

Vilka svårigheter stötte ni på?

Vad innebär de gula blocken (framförallt ”loop”)? Och var kommer blocken ifrån?

Vad innebär de mörklila blocken? Var kommer blocken ifrån? (utdata)

*Vad?*

Eleverna ska förstå hur kod kan förenklas med hjälp av loopar (repetera-blocket) och hur program kan kommunicera ut med hjälp av utdata (säg-blocket).

*Hur?*

Eleverna klickar på uppgift 2 i studion och sedan ”se inuti”. Där finns alla uppdrag, och koden som de ändrar i. När eleverna haft tid till att testa på de olika uppdragen ber du dem reflektera över reflektions/diskussionsfrågorna i paren (eleverna kan se diskussionsfrågorna på sina egna enheter om det klickar tillbaka till projektsidan). Samla sedan upp i helklassdiskussion.

Om ingen elev preciserar vad den inre och yttre loopen har för funktion behöver du vara förberedd på att själv leda in diskussionen mot det.

*Varför?*

Två grundelement i programmering är utdata och loopar. Eleverna får här erfarenheter av hur dessa kan fungera i en kod innan de själva gör sin egen kod.

Det finns olika sätt som en programmerare, genom sin kod, kan kommunicera med en användare, vi kallar det utdata. Här stöter eleverna på möjligheten att få programmet att stanna upp och skriva ut en text, vilket kan användas både som ett sätt att kommunicera till en tänkt användare men också som ett verktyg under kodningsprocessen för att kontrollera att koden gör som det är avsett.

Loopar används för att minska antalet instruktioner som behöver kodas, genom att upprepa väl valda delar av koden. Vanligt är att göra upprepningar tills ett villkor har uppnåtts, t.ex. att stegvis gå igenom en lista tills koden hittar en speciell rad eller listan tar slut, eller som i det här fallet upprepa ett visst antal rörelseinstruktioner för att skapa ett upprepande mönster.

1. Ladda programmet ”Större än…”

1) Testa att rätta stavfelen i frågan som katten ställer.

2) Testa att ändra kattens svar.

3) Testa att få katten att ställa en annan fråga, med ett annat tal än 10.

4) Testa att få programmet att reagera på rätt sätt utifrån den nya frågan.

Fördjupning: Ändra i koden så att det blir som ett spel, där spelaren ska försöka hitta tal som är större än ett slumpmässigt tal. Varje gång spelaren lyckas så delas det ut poäng, och när spelaren gissar fel så blir det ”game over"

Teman för den uppsamlande diskussionen:

Vilka svårigheter stötte ni på?

Vad innebär de gröna blocken och var kommer blocken ifrån? Vilka mer block finns här? (villkor och räkne-operatorer)

Vad innebär de ljusblåa blocken och var kommer blocken ifrån? (indata/variabel)

Vad innebär det nya gula blocket och var kommer blocket ifrån? (villkorade satser)

Fördjupningsuppgiften: Vad innebär de roströda blocken och var kommer blocken ifrån? (listor och variabler)

*Vad?*

Eleverna ska förstå hur ett program kan skapa interaktivitet med en användare genom att vänta på indata (fråga-blocket) och sedan reagera på olika sätt utifrån vilket indata som ges (om-då-annars-blocket).

Eleverna ska förstå hur programmet kan skapa interaktivitet genom att be en användare att ge indata (fråga-blocket) som sedan programmet reagerar och ger återkoppling på (om-då-annars-blocket).

Eleverna ska förstå hur ett program kan skapa interaktivitet genom att be en användare att ge indata till programmet (fråga-blocket) som det sedan ger återkoppling på (om-då-annars-blocket).

*Hur?*

Eleverna klickar på uppgift 3 i studion och sedan ”se inuti”.

*Varför?*

Programmering i F-3 innebär att få förståelse för hur stegvisa instruktioner kan skapas och följas. I miljöer så som bluebots och scratch JR innebär det att ett fysiskt eller virtuellt objekt följer enkla instruktioner. Här vill vi att eleverna utvecklar sitt algoritmiska tänkande till en punkt där de kan tänka in hur ett program kan reagera på omständigheter som uppstår i interaktion med en framtida användare.

1. Ladda programmet Procentspelet (fördjupning)

Klicka på ”remix” i kod-läget och försök få spelet att handla om en annan typ av räkneoperation än procenträkning.

*Hur?*

Eleverna klickar på uppgift 4 Fördjupning i studion, sedan ”se inuti” och sedan ”remix”. På så sätt skapas en kopia av koden i ett eget projekt hos eleven. Där kan sedan eleven göra ändringar i koden som kan sparas och delas.

*Varför?*

Elever som använt scratch tidigare kan behöva en större utmaning. Här erbjuder du ett mer avancerat program som eleven måste förstå för att sedan kunna göra förändringar inom. Det är tänkt för elever som redan arbetat en del med programmering när lektionen börjar snarare än elever som är lite snabba.

Uppsamling:

Sammanfatta programmering i termer om Instruktioner som innehåller

* indata (ljusblåa ”känna av” ), indata är när programmet väntar på att användaren ska ge programmet information, t.ex. genom att skriva in en sträng, peka med muspekaren eller använda speciella tangenter.
* utdata (mörklila ”Utseende” , ljuslila ”Ljud”  och mörkgröna ”Penna” ), utdata är programmets sätt att kommunicera med användaren, t.ex. genom att spela upp ett ljud, visa en bild, eller göra någon annan visuell förändring på skärmen.
* operatorer (ljusgröna ”operatorer” ), operatorer är programmets sätt att bearbeta data som sparats i t.ex. variabler eller listor, ofta från indata av användaren.
* villkorade satser (gula ”kontroll” ), villkorade satser är ett programs sätt att välja mellan två alternativ. Det gör valen utifrån villkor, och det är inte ovanligt att flera villkorade satser används i varandra för mer avancerade resultat.
* loopar (gula ”kontroll” ), upprepningar är en effektivisering av kod, där programmet kan upprepa samma instruktion flera gånger utan att koden för den sakens skull är skriven flera gånger.
* ev också variabler och listor (orange/roströd ”data” ), variabler och listor är ett programs sätt att spara data, ofta som ett resultat av indata, temporärt eller över tid för att användas i olika operationer eller villkor.

1. En läraranvändare behöver skapas några dagar i förväg, eftersom det tar några dagar innan din användare blir godkänd av scratch-teamet. [↑](#footnote-ref-1)
2. <https://scratch.mit.edu/studios/5537076/> [↑](#footnote-ref-2)
3. Under fliken Elever finns en knapp för att skapa en Länk för elevregistrering. Klicka på den så får du fram en lång länk som du ska ge till eleverna. När de klickar på länken så uppmanas de att skapa en ny användare, Påminn dem att döpa den till sitt eget namn, de måste skapa en ny användare via länken för att få tillgång till ditt klassrum. Den användare eleverna skapar knyts automatiskt till ditt klassrum. [↑](#footnote-ref-3)