# Lektion 2 (ca 60min)

Syfte och lärandeobjekt

Syftet med lektionen är att vidareutveckla elevers algoritmiska tänkande i programmering, genom att låta elever skapa egen kod för att lösa avgränsade uppdrag. Speciellt att fördjupa förståelsen hur indata, utdata, loopar, variabler och villkorssatser kan kombineras till ett fungerande program.

Material

Skapa klassrum (Se lektion 1 för instruktioner)

Skriva ut elevinstruktioner

Inledning

Förra lektionen bekantade ni er med Scratch’s möjligheter, ni tittade till exempel på indata (ljusblå), utdata (t.ex. mörklila), loopar (gula) operatorer och villkor (gröna) och villkorssatser (gula). Målet är att skapa matematikspel till yngre barn, som handlar om de fyra räknesätten, därför ska ni ta en lektion till att bekanta oss ytterligare med alla möjligheter i Scratch. Under denna lektion ska ni göra egna program. Att göra egna program, eller att programmera, innebär att bryta ner ett stort problem i många små problem. Lösningarna till dessa små problem ska programmeraren göra instruktioner till datorn utav och på så sätt bygga upp programmet. Detta innebär att eleverna måste träna sin uthållighet och ihärdighet i att lösa problem. För att de enklare ska kunna ta hjälp av varandra, och hämta inspiration från varandra, så ska de dela våra program så att de syns i ert klassrum. Be eleverna att:

1. Logga in på Scratch.com[[1]](#footnote-1) med de uppgifter du använde senaste lektionen, så att du får tillgång till vårt klassrum.
2. Skapa ett nytt projekt, kalla det för ”[Ditt namn] Lektion 2”. T.ex. Per Lektion 2
3. Dela ditt projekt
4. Klicka på ditt användarnamn uppe i höger hörn, välj ”Min klass”, klicka sedan på ”Lektion 2”. Lägg nu till ditt projekt med knappen ”Lägg till projekt”
5. Nu är du redo att börja med lektionens uppgift.

Följande text kan du använda som inspiration till en introduktion till eleverna:

**-------**

Ni ska skapa ett program där sprajten frågar spelaren kluriga matematikfrågor, och som kan berätta om spelaren svarat rätt eller inte, vi kallar detta ett spel (visa multiplikationsspelet <https://scratch.mit.edu/projects/259793797/>). Om ni känner er trygga med den tanken, har en idé hur man skulle kunna börja, och tror sig kunna jobba självständigt kan göra det. Ni som tror att ni skulle behöva mer stöd, får gärna följa denna steg-för-steg-instruktion. Den innehåller uppgifter och fördjupningsuppgifter, fördjupningsuppgifterna gör du om det går snabbt och enkelt och du vill lära dig lite mer. Om du inte vet hur du ska börja:

1. läs instruktionen en gång till
2. sen fråga en person vid dit bord
3. vet du fortfarande inte pröva att klicka in på någon klasskamrat i din klass för att se hur hen har gjort (Klicka på ”Min klass”, sen ”Lektion 2”, sen något av de andra projekten som du ser där).

-----

Om elevens program inte gör som hen har tänkt, be denne att börja med att stega igenom programmet med ett finger samtidigt som hen kör programmet. Om hen inte hittar felet, gör om samma procedur samtidigt som hen visar och förklarar för en annan person vid bordet. Även här kan man kika in i andras projekt i klassrummet, för att se hur någon annan löst liknande problem. Dessa uppmaningar hittar ni också i de blå figurerna i instruktionerna, stega genom dessa innan ni räcker upp handen och ber om hjälp.

Förbered eleverna på att ni kommer plocka ut korta exempel på elevlösningar på tavlan utifrån vad eleverna presterar i varje uppgift. Använd detta som ett verktyg för att driva lektionen framåt, så att eleverna som fastnar på någon uppgift får se förslag på hur de kan tänka och komma vidare. Fundera i förväg på ungefär när i lektionen du förväntar dig att eleverna är klara med varje uppgift.

Här finns exempel på hur koden skulle kunna se ut efter varje uppgift:

<https://scratch.mit.edu/studios/5566033/> (hittas även som bilder sist i lärarguiden)

Aktivitet

1. Input, output och enkla variabler

*Vad?*

Eleverna ska förstå hur man får ett program att kommunicera (säg och fråga-blocken) och be om input (fråga-blocket) som sparas i en variabel (svar-blocket).

*Hur?*

Ge eleverna uppdraget att skapa ett program som frågar efter någon form av i indata, t.ex. med ett textfält, och som sedan kan upprepa denna indata som utdata, t.ex. i form av en textruta.

Till elev:

1. Få sprajten att säga något i en pratbubbla
2. Fortsätt sedan genom att få sprajten att fråga spelaren någonting (tips: leta under ”Känna av”)
3. Få sprajten att svara detsamma som spelaren skriver in i rutan (tips: leta efter blocket ”svar”)

*Varför?*

Ett program/spel behöver interaktivitet åt båda hållen, för ett grundläggande program behöver eleverna behärska åtminstone en form av indata/utdata. Att sedan spara indata i form av olika variabler är utvecklande både för den matematiska resonemangsförmågan och problemlösningsförmågan i programmering.

Fördjupningsuppgift:

Just nu så svarar sprajten det som spelaren själv svarat i första frågan. Utforska hur du kan utveckla detta med hjälp av blocket Förena så att det blir ett mer naturligt svar från sprajten.

1. Villkor

*Vad?*

Eleverna ska förstå hur villkor (”>=<”-blocken) kan användas i en kod för att jämföra en variabel (variabel och svar-blocken) mot ett värde eller en textsträng.

Variabler (svar och variabel-blocken) kan användas i en kod, t.ex. med en räkneoperation (operationer ”+-\*/”-blocken) eller att kolla ett svar mot ett bestämt värde (villkor ”>=<”-blocken).

*Hur?*

Ge eleverna i uppdrag att bygga vidare på sitt program och få det att testa en variabel mot ett förutbestämt värde, och sedan svara med de booleska värdena sant eller falskt.

Till elev:

Du ska nu programmera din sprajt så att den gör ett test. Scratch kan testa om ett tal är större än, mindre än eller lika med ett annat tal, men den kan också testa om du t.ex. svarat ett speciellt namn. Bestäm vad du vill att sprajten testa.

1. Få sprajten att ställa en till fråga till spelaren. (t.ex ”Vad heter du?”, ”Kan du skriva ett tal som är mindre än 7?”, ”Vilken skola går du i?”)
2. Ta fram en operator (”lika med”, ”större än” eller ”mindre än”) som passar med svaret.
3. Få sprajten att ”säga" resultatet av testet du skapat. Även om det inte ser ut så, så passar ditt operator-block i säg-blocket.

Tips: Hinner du inte längre se svaret på den första frågan? Pröva att lägga till att programmet väntar en liten stund efter ”säg”, eller använd blocket ”säg i 2 sekunder”.

*Varför?*

Ett sätt att vidareutveckla förståelsen för vad ett program gör, och hur variabler fungerar, är att precis som i matematiken jobba vidare med dessa. Utforska hur det går att manipulera indata och variabler för att skapa mer intressanta möjligheter till villkorade satser och utdata.

Fördjupningsuppgift:

Just nu så är det alltid samma siffra i ditt test. Utforska hur du kan använda blocket “Slumpa tal” mellan för att göra spelet mer spännande.

1. Villkorade satser

Vad?

Logiska strukturer utifrån om/annars-uttryck som gör att det är möjligt att köra olika instruktioner beroende på hur olika villkor uppfylls eller inte.

Hur?

Ge eleverna i uppdrag att bygga vidare på sitt program så att det kan svara olika beroende på det villkor som användes uppgiften innan. Just nu så svarar programmet automatiskt True eller False.

Till elev:

I uppgiften innan så säger sprajten automatiskt True eller False. Nu ska du bestämma vilka svar sprajten säger. Detta gör du med hjälp av en villkorssats. Sätt samman följande villkorssats bredvid ditt befintliga program och testa hur den fungerar genom att skriva olika siffror i den tomma rutan.



När du känner att du förstår hur villkorssatsen fungerar kan du byta ut det sista ”säg”-blocket mot detta och anpassa det så det fungerar ihop med ditt program.

1. Få sprajten att säga olika beroende på vad spelaren svarar/frågar (liknande bilden ovan).

Tips: Ofta när man programmerar så testar man mindre delar av sin kod, så som du precis gjorde, innan man sammanfogar det med resten av koden.

Varför?

Program bygger på en struktur av villkor, som gör att programmet kan reagera på mer än bara nya indata. Eleverna behöver få en ökad förståelse för hur indata kan ”färdas” i ett program, och utlösa olika händelser beroende på olika villkor.

Fördjupningsuppgift:

Utforska olika operationer, se om du t.ex. kan fråga spelaren ”Vad blir 7\*8?”, och sedan utnyttja multiplikationsblocket tillsammans med lika-med-blocket i din villkorssats.

1. Loopar/Rekursion

Vad?

Förenkla instruktioner genom att upprepa repetitiva moment, eller börja om då programmet kommer till vissa punkter i förgrenade villkorade satser.

Hur?

Ge elever i uppdrag att bygga vidare på sitt program så att det startar från början igen.

Till elev:

För att få programmet att bli mer likt ett spel så ska det inte ta slut. Fundera över vilken del av ditt program som måste repeteras för att detta ska ske.

1. Få sprajten att repetera sin kod automatiskt.

Varför?

Kod riskerar att snabbt bli lång och ineffektiv. Det är då viktigt att hitta möjligheter att förenkla, genom att utnyttja datorers förmåga till att repetera instruktioner effektivt. Loopar ger möjlighet att upprepa operationer, eller se till att program inte når vägs ände utan börjar om vid en bestämd punkt för att vänta på nya indata.

1. Variabler

Vad?

Skapa egna definierade variabler som kan användas t.ex. för poäng i ett spel eller för mer avancerade beräkningar.

Hur?

Ge eleverna i uppdrag att skapa variabler och testa hur de kan ändra dess värden. T.ex. en variabel med poäng som ökar för varje rätt svar och en variabel med liv som minskar för varje fel svar.

Till elev:

Du ska göra ditt program mer likt ett spel, där spelaren får poäng när hen svarar rätt på din matematikfråga.

1. Skapa en ny variabel. Denna har inget värde, så det först du behöver göra är att sätta den variabeln till ett startvärde.
2. Se till att din variabel påverkas beroende på om spelaren svarar rätt eller fel. Notera att du kan påverka dina variabler på två sätt, antingen kan du göra så att den bestämmer ett värde (sätt variabel till…), eller ändra det befintliga värdet (ändra variabel med…). Testa vad som passar bäst i din situation.
3. Testa att istället för att öka/minska din variabel med ett hela tiden, få den att öka/minska med ett slumptal. Slumptal är ett grönt block.

Varför?

Egenskapade variabler är mycket mer mångsidiga än den som automatiskt skapas i och med fråge-blocket. Egenskapade och definierade variabler kan sättas till samma värde och manipuleras på olika sätt samtidigt som spelaren får en ny fråga. Ett annat exempel på användning är att slumpa fram tal som sedan sparas som variabler och kan göra spel mindre förutsägbara.

Fördjupningsuppgift:

Skapa två nya variabler och sätt dem till slumptal. Undersök hur du kan få in dessa två i den matematikfråga som sprajten frågar spelaren.

1. Om en elev glömt bort sitt användarnamn/lösenord kan du som lärare gå in och hjälpa eleverna med detta genom ditt lärarkonto. [↑](#footnote-ref-1)