Kapitel 2: Starta programmering på riktigt

Övning 3: Upprepade beräkningar

I den tredje övningen i enhet 2 får du lära dig hur du på ett enkelt sätt kan upprepa tilldelningar.

Syfte:

- Hur man använder kommandot **while**
- Exempel på hur man använder **while** i enkla situationer

Antag att du vill upprepa en serie kommandon ett antal gånger men du vet inte i förväg hur ofta. Då kan du använda kommandot **while**. Här är vad kommandot gör:

Så länge ett visst villkor gäller, fortsätt att upprepa

Vi ska nu skapa ett program för följande problem:

Kasta en tärning och upprepa tills en sexa kommer upp.

Det går inte i förväg att veta hur många gånger du måste kasta tärningen. Öppna nu ett nytt pythonprogram och ge det ett namn. Eftersom vi ska använda slumptal, behöver vi först slumptalsmodulen ”from random import”. Den ska vi sätta in i programmet först. Välj F1 (Fns...) sedan Modul och därefter 2: random. Du kan också trycka på tangenten [math] och sedan 2:random.

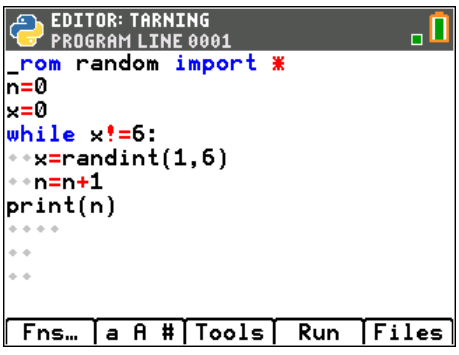
Vi kommer att behöva två variabler i programmet. Variabeln ***n*** som håller reda på antalet kast och variabeln ***x*** som håller reda på vad som kastas (1 t.o.m. 6). Vi ger båda variablerna värdet 0 från början.

Därefter lägger vi till mallen för while. Tryck på f1 (Fns…) och välj alternativ 8:while condition. Villkoret blir ”inte lika med 6” vilket kan skrivs som != i Pythonspråket. Upprepning ska alltså ske tills d får värdet 6.

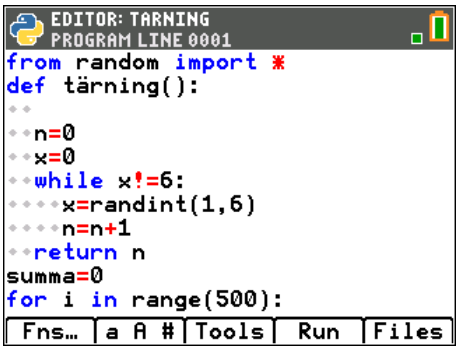
while-blocket innehåller två tilldelningar. Först ska vi ha ett slumptal *d* mellan 1 och 6 och sedan ska vi öka räkneverket *n* med 1. Får vi nu 6 som resultat vid alstringen av slumptalet avslutas körningen och vi får en utskrift av hur många gånger vi fick kasta tills en sexa kom upp.

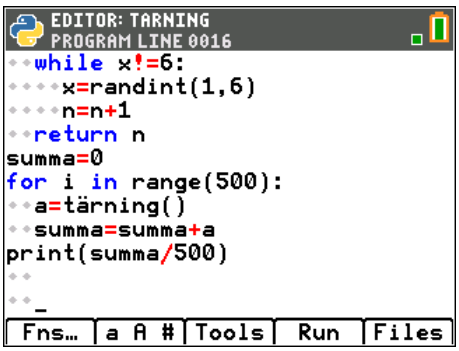
Om vi nu kör programmet ett antal gånger så får vi lite olika värden på *n*.

Om du ska köra programmet flera gånger så kan du trycka på f3 (Tools) och välja 1:Rerun last Program.



- Vi ska nu bygga på programmet så att vi inte bara gör ett kast utan ett stort antal kast och där vi sedan beräknar medelvärdet från de många kasten.
- Först så att själva kastet läggs i en funktion ”tills en sexa kommer upp”. Vi kallar denna funktion tärning(). Om du anropar denna funktion så kommer den att rapportera antalet kast.
- Om vi sedan t.ex. anropar denna funktion 500 gånger, adderar resultatet och dividerar denna summa med 500 så får vi ett medelvärde.
- Vi lägger alltså till en **for**-loop som gör denna upprepning 500 gånger.
- Då måste vi först lägga till en ny variabel, *summa*, som har startvärdet 0 från början.
- I programmet så körs for-loopen 500 gånger och summan uppdateras efter varje kast. Till sist så beräknas medelvärdet som summan/500.





Lärarkommentar: Vi människor och programmerade datamaskiner har minne och vet att det t.ex. ska bli ungefär lika många sexor som ettor om vi kastar en tärning många gånger. Det vet inte tärningen eller kronan när den hänger i luften. I ett äkta slumpförsök finns det ingenting som heter minne. Allt börjar om igen när vi kastar tärningen nästa gång.

Räknaren kan alltså egentligen *inte* generera riktiga slumptal. Den använder faktiskt en *formel för att generera en sekvens av tal som verkar vara helt slumpmässiga*. Varje tal i sekvensen beror på det föregående talet. Detta betyder då att hela sekvensen av tal beror på det första talet. Detta tal kallas för en *kärna*. Om du låter två räknare få samma kärna, t.ex. talet 273, så genereras samma sekvens. Det kan väl knappast kallas slumpen.

Normalt så behöver man dock inte tänka på detta eftersom räknaren hela tiden använder det sista slumptal som har alstrats som en ny kärna. I den övre skärmbilden har vi två gånger lagrat talet 273 som kärna (med kommandot 173 STO▸ slump. Vi får samma sekvens av tal båda gångerna. I den nedre skärmbilden har vi lagrat talet 273 bara en gång och vi ser att vi får olika sekvenser av tal varje gång.