Kapitel 2: Programmera i Python	Övning 3: While-loopar
I detta avsnitt skall vi titta på en annan slags loopar, while-loopar.	**Mål:** • Att använda while-loopar • Kunna while-loopars syntax i Python

Om du vill upprepa ett block av kod ett antal gånger, men du inte vet i förväg hur många gånger det blir så kan du använda en while ... loop.
En while loop kan ses som att om ett visst villkor är sant så skall vi göra en viss sak om och om igen.

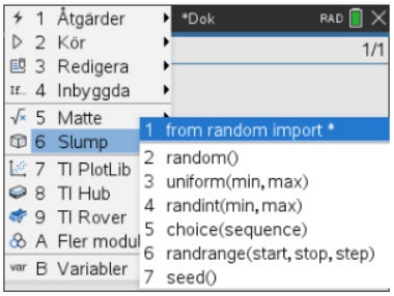
Så länge det finns disk så diska.
Så länge ett visst villkor gäller, fortsätt upprepa.

Vi skall skriva ett program som simulerar följande problem:
Rulla en tärning och upprepa detta tills du får en sexa och räkna hur många gånger du behövde göra detta.

Du vet inte i förväg hur ofta du måste kasta -så här kommer while-loopen till pass.

Öppna ett nytt Python-program och ge det ett namn.
Eftersom vi ska använda slumptal behöver vi en modul kallad random (slump). Den innehåller ett antal slumptalsfunktioner.

I Python är det vanligt att hämta hem moduler i början av programmet.
Du hittar den här modulen i menyn Random och sedan alternativ 1 för att importera alla funktioner.



Tips: När du väljer ett namn till funktionen kan du också välja vilken typ av program det skall vara. Väljer du "Slumpartade simuleringar" så kommer modulen random automatiskt läggas till.

Vi kommer att använda två variabler i programmet. Variabeln "antal" som håller reda på antalet kast och variabeln "resultat" som håller reda på vad resultatet efter ett kast blir. (Du kan naturligtvis också välja andra namn)

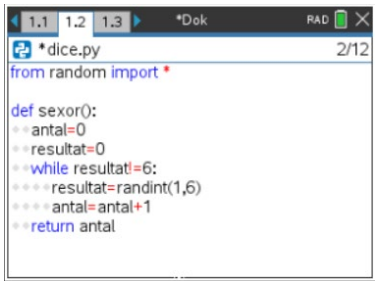
Vi ger båda variablerna utgångsvärdet 0.

Lägg till mallen för en **while..** med Meny> Inbyggda> Kontroller>while..
I det här fallet skall vi loopa så länge resultatet inte är like med 6. I Python skriver vi det som **reultat!=6**, där != betyder "inte lika med".
Vi har sedan en funktion, **randint()** som du hittar under meny>slumt>randint. Den returnerar ett slumpartat tal mellan två tal, inklusive talen. Så randint(1,6) kommer simulera en tärning.

Loopen innehåller två rader. I den första väljs slumpartat ett av talen 1, 2, 3, 4, 5 eller 6 och i den andra ökas antal med 1.
Om resultatet av "tärningskastet" är lika med 6 avslutas loopen och antalet kast som krävdes skrivs ut.
Om du kör programmet ett antal gånger visas förmodligen olika tal.

Tips: I konsolen kan du köra det senast körda programmet igen med Ctrl+R.

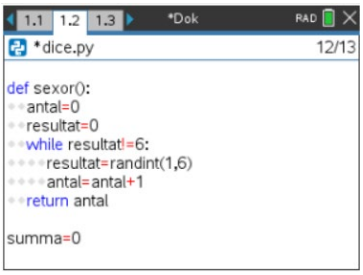
Men hur gör vi om vi vill göra experimentet 100 gånger och ta medelvärdet av resultatet?



Vi kan börja med att göra om det vi har gjort till en funktion som returnerar antalet varv.

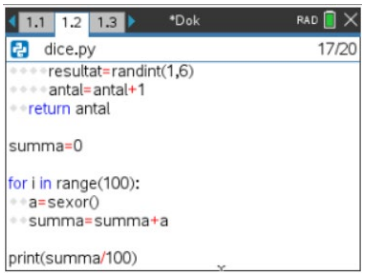
(Du kan "flytta" hela funktionsblocket på en gång genom att välja det med shift-tangenten och piltangenterna och sedan trycka på tabbtangenten.)

Om du anropar denna funktion 100 gånger, successivt adderar resultaten och sedan delar summan med 100 får du det genomsnittliga antalet kast.



Börja med en variabel summa och ställ den till noll.

Sedan så gör vi en for-loop som skall loopa 100 gånger. Inne i den så anropar vi vår funktion, så sparar vi svaret i en variabel och så adderar vi detta till summan. Slutligen så skriver vi ut summan/100.



Mata in detta och provkör några gånger.

Om man vill prova med 1000 experiment så måste man ändra detta på två platser. Bättre är kanske att skapa en variabel som vi sätter till antalet försök, och sedan använder vi den på bägge platserna där vi har talet 100.

Ännu mer generellt är att placera andra delen av programmet i en funktion där parametern är antalet försök som skall göras och som sedan returnerar genomsnittet.