Unit 2 : Programmeren in Python | **Oefenblad 3 : De while lus**

In les 3 van deze unit leren we hoe je opdrachten kunt herhalen

Doelen :

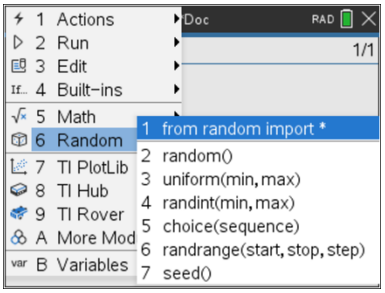
- Hoe gebruik je de opdracht **while ...** .
- Wat is de syntax voor while opdracht in Python

Als je een serie opdrachten een aantal keren wilt herhalen, maar je weet van tevoren niet hoe vaak, dan kun je de **while...** lus gebruiken.
Dit kun je lezen als:
Zolang een bepaalde voorwaarde geldt blijf dan herhalen.

We maken een programma dat het volgende probleem simuleert:
Werp met een dobbelsteen en herhaal dit net zolang totdat je 6 gooit en tel het aantal keren dat je hiervoor nodig hebt.
Je weet dus van tevoren niet hoe vaak je moet gooien.

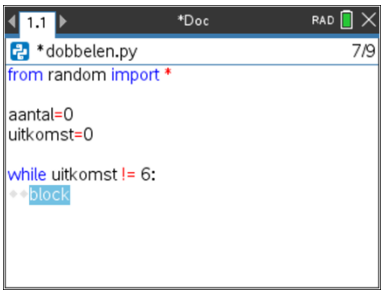
Open een nieuw Python programma en geef het een naam.
Omdat we randomgetallen gaan gebruiken hebben we de random module nodig.
Deze module is een soort bibliotheek met functies voor randomgetallen.

In Python is het gebruikelijk om dit aan het begin van het programma te doen.
Je kunt deze module vinden in het menu bij Random en dan keuze 1 om alle functies te importeren.



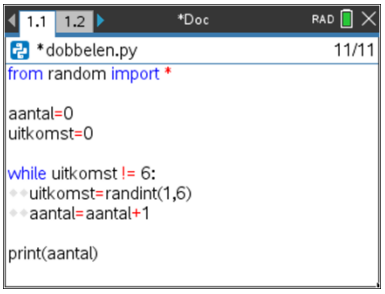
Tip voor de docent: Bij het kiezen voor een naam van het programma kun je ook een type kiezen. Een van die keuzes is Random Simulations.
Als je hiervoor kiest wordt de random module ook toegevoegd.

We gaan in het programma twee variabelen gebruiken.
De variabele ‘aantal’ die bijhoudt hoe vaak er is gegooid en de variabele ‘uitkomst’ die bijhoudt wat er is gegooid.
Beide variabelen geven we als startwaarde 0.
(Je kunt natuurlijk ook andere namen kiezen)
Voeg het sjabloon toe voor **while** ... met Menu > Built-ins > Control.
Achter while komt de voorwaarde te staan.
Dat is in dit geval: uitkomst is ongelijk aan 6
In Python noteer je dat als: **uitkomst != 6**. (dit kun je intypen, maar je kunt ook in het menu kijken bij Built-ins en dan Ops)



In het herhalingsblok komen twee opdrachten te staan. De eerste is het kiezen van een randomgetal tussen 1 en 6 en de tweede is het aantal met 1 verhogen.
Als de uitslag gelijk is aan 6 wordt de lus verlaten en volgt nog de opdracht om het aantal worpen af te drukken.

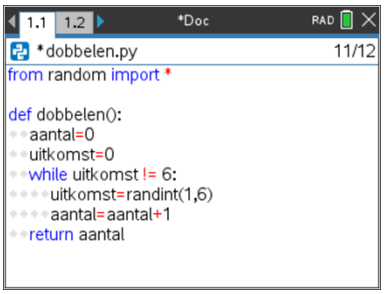
Als je het programma een aantal keren uitvoert (Ctrl+R) dan zie je waarschijnlijk verschillende waarden voor aantal.



Tip voor de docent: Je kunt in de shell met Ctrl+R het programma opnieuw uitvoeren.

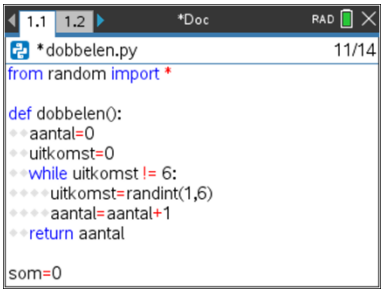
We gaan het programma wijzigen zodat we het experiment niet één keer maar honderd keer uitvoeren en dan het gemiddeld aantal worpen berekenen.

Verander eerst het vorige programma zo dat het werpen met de dobbelsteen totdat zes is gegooid in een functie wordt gezet (noem die functie bijvoorbeeld dobbelen()).
Let op het gebruik van de spaties.
Je kunt het hele functieblok in een keer ‘opschuiven’ door het met de shift toets en de pijltoetsen te selecteren en dan de tab toets in te drukken.

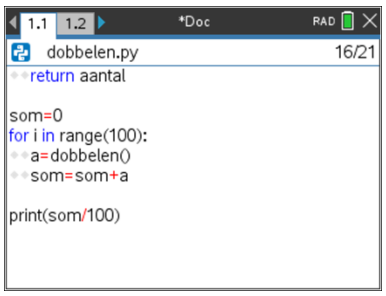


Als je deze functie 100 keer uitvoert, de aantallen optelt en het totaal deelt door 100 krijg je het gemiddeld aantal worpen.

Begin met een variabele som en zet deze op nul.



Maak nu een for-lus waarin de dobbel-functie 100 keer wordt uitgevoerd en tel telkens het aantal worpen op bij de som.



Als laatste komt de opdracht om som/100 af te drukken.

Om het gemiddelde nauwkeuriger te bepalen kun je bijvoorbeeld het getal 100 in het programma wijzigen in 1000 (op twee plekken).
Je berekent dan het gemiddelde van 1000 experimenten.