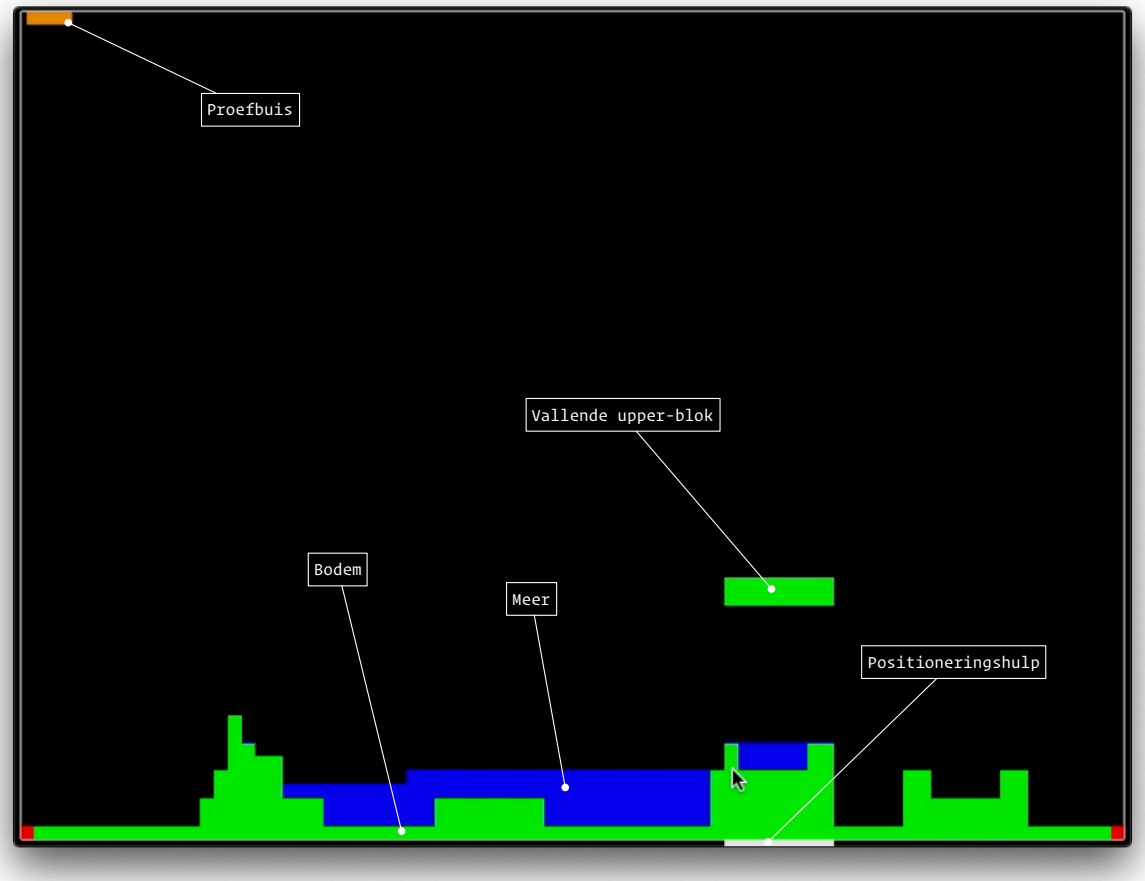
Inleiding tot de Computerwetenschappen
Taak “Wetrix” – Deel 2

Inleiding

Het doel van deze taak is om een eigen versie van het spel “Wetrix” te maken. Wetrix lijkt op het klassieke tetris. Blokken vallen vallen naar beneden en je moet ze positioneren op de bodem van het speelveld. Het doel van het spel is dijken bouwen die het speelveld omsluiten. Het speelveld stelt een stuk van de aardbodem voor, dat in het begin vlak is. Er zijn verschillende soorten blokken, met elk een eigen functie. De basisblokken zijn “uppers”, “downers” en waterblokken. Als een upper op het speelveld beland wordt de bodem op die plaats opgehoogd. Een downer doet het omgekeerde en verlaagd de bodem. Een waterblok laat water vallen op de bodem. Waar geen dijken voorzien zijn, stroomt het water van het speelveld af. Tussen dijken worden het water vastgehouden. Het spel eindigt wanneer te veel water verloren is gegaan.

Via volgende links kan je een tutorial en de gameplay van het spel bekijken. Meer video’s kan je uiteraard terugvinden via Google. Het originele spel werkt met een driedimensionaal speelveld. Wij bouwen een vereenvoudigde versie in twee dimensies, waarbij het water links en rechts van het speelveld wegstroomt. Hieronder zie je een voorbeeld van een eenvoudige implementatie, met aanduiding van de belangrijkste componenten in het spel.



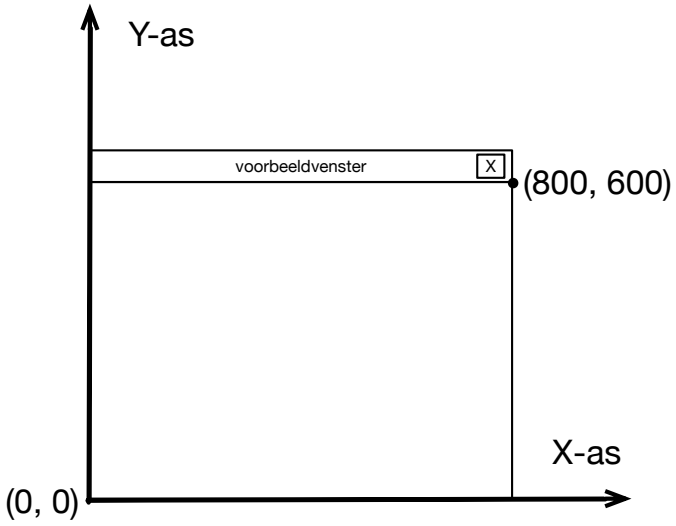
Figuur 1: Voorbeeldimplementatie van Wetrix in 2D

De taak bestaat uit twee delen. In dit deel van de taak bouwen we het echte spel, **maak hierbij zoveel als mogelijk gebruik van objectgerichte programmeertechnieken**! De functies uit deel 1 kan je hergebruiken in deel 2 of helpen je om deel 2 te bouwen. De grafische voorstelling bouw je met behulp van de VPython module. In wat volgt beschrijven we de componenten van het spel.

HELPERS

Gegeven is de module `helpers`. Deze module bevat een functie `open_display` die gebruikt kan worden om een VPython venster te openen met een bepaalde hoogte en breedte. Dit venster toont dan steeds het eerste kwadrant van het assenstelsel. Onderstaande code opent dus een venster met breedte 800 en hoogte 600 zoals aangegeven op de bijhorende figuur.

```
venster = open_display(title = "voorbeeldvenster", width = 800, height = 600)
```

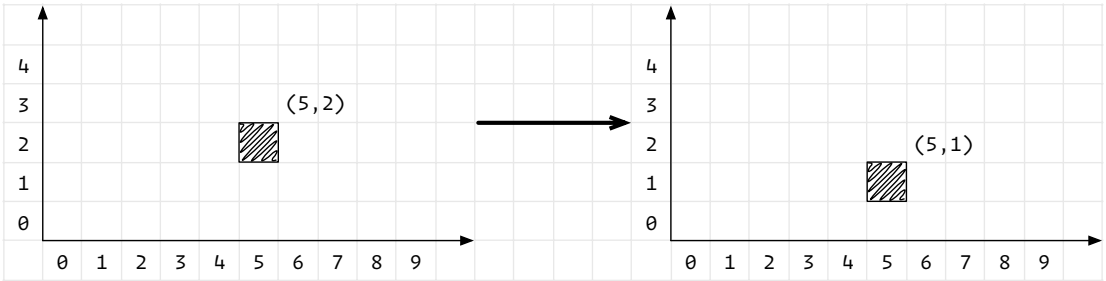


Je bent vrij om de dimensies van het speelveld te kiezen.

ROOSTER

Om het spel te bouwen gebruiken we een roostervoorstelling. Het rooster laat ons toe op eenvoudige wijze blokken te plaatsen in het spel en ze te laten bewegen. De grootte van de vakken in het rooster is de basiseenheid waarmee we werken. Elke component in het spel heeft een positie in het rooster, t.t.z. een aantal vakken horizontaal en verticaal. Componenten laten bewegen doen we door ze één positie (één vak) op te schuiven naar links, rechts, boven of onder in het rooster.

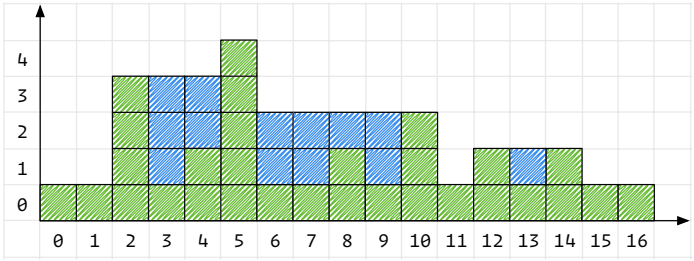
In de afbeelding hieronder hebben we een vierkant getekend (bijvoorbeeld door middel van Box). In het rooster staat het vierkant op positie (5,2). Het vierkant naar beneden verplaatsen doen we eenvoudig weg door de verticale positie in het rooster met 1 te verminderen. De positie wordt dan (5,1) in het rooster.



Om het vierkant effectief te tekenen (d.m.v. VPython) moet je de roosterpositie omrekenen naar de positie in het carthesisch coördinatenstelsel. Je kan daarvoor gebruik maken van de functies uit deel 1. Merk op dat je het rooster zelf niet in Python hoeft voor te stellen en niet moet visualiseren. Het is voldoende dat de componenten hun plaats in het rooster kennen.

Bodem

Onderaan het speelveld bevindt zich de bodem. Op elke horizontale positie in het rooster heeft de bodem een grondhoogte en een waterniveau. De grondhoogte en het waterniveau zijn gegeven als een aantal vakken in het rooster. Bijvoorbeeld, in de afbeelding hieronder heeft het stuk bodem op horizontale positie 4 grondhoogte 2 en waterniveau 2. Dat komt overeen met 2 vakken van het rooster die groen gekleurd zijn met daarboven 2 vakken blauw gekleurd.



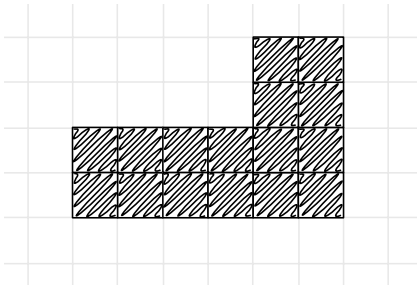
Proefbuis

Ergens op het speelveld wordt een zgn. “proefbuis” afgebeeld. De proefbuis geeft visueel weer hoeveel water van de bodem is afgestroomd (verloren gegaan). De proefbuis heeft een maximumcapaciteit die je vrij mag kiezen. Wanneer de proefbuis overloopt en er dus teveel water verloren is gegaan eindigt het spel. Je bent vrij om te kiezen hoe je de proefbuis visueel voorstelt. In figuur 1 is de proefbuis getekend als een horizontale balk die zich van links naar rechts vult.

Blokken

Tijdens het spel valt er telkens één willekeurig spelblok naar beneden. Zo’n spelblok bestaat uit verschillende kleine blokjes (de grootte van een vak in het rooster) die samen bewegen. Een blok valt volgens een regelmatig ritme en beweegt dus op vaste intervallen één vak naar beneden in het rooster. Een blok kan horizontaal verplaatst worden door de cursor (de muis) te bewegen over het speelveld. Dat wordt later verder besproken.

Hieronder zie je een voorbeeldblok in de vorm van een “liggende L” opgebouwd uit 16 kleine blokjes.



Wanneer een blok in aanraking komt met de bodem of een meer heeft dat een bepaald effect. Het effect is afhankelijk van het type blok. Volgende blokken zijn zeker aanwezig in het spel:

- **“Uppers”** verhogen de bodemhoogte op de plaats(en) waar de individuele blokken de bodem raken. De verhoging per horizontale positie in het rooster is gelijk aan het aantal kleine blokjes die op die plaats de bodem raken. Voorzie minstens 4 verschillende vormen van upper-blokken (bv. in de vorm van een L, T, I, en U).
- **“Downers”** verlagen de bodemhoogte op de plaats(en) waar de individuele blokken de bodem raken. De verhoging per horizontale positie in het rooster is gelijk aan het aantal kleine blokjes die op die plaats de bodem raken. Voorzie minstens 2 verschillende vormen van downer-blokken.
- **Waterblokken** verhogen het waterniveau op de plaats(en) waar de individuele blokken de bodem raken. Waterblokken hebben een vaste rechthoekige vorm.

- **Vuurblokken** verdampen al het water in het meer waarin ze terecht komen. Als een vuurblok op het land terecht komt creëert het een krater. Op de plaats waar een vuurblok het land raakt wordt de bodemhoogte herleid naar 1. De bodem errond wordt zodanig afgebroken dat ze schuin oploopt naar links en rechts.

Je bent vrij om kleuren en vormen te kiezen om blokken visueel weer te geven. Zorg dat alle blokken goed van elkaar te onderscheiden zijn door verschillende kleuren of vormen te gebruiken. Bijvoorbeeld, uppers zijn groene vierkantjes, downers rode vierkantjes, water blauwe bollen en vuur is geel.

De effecten van bovenstaande blokken kunnen tot gevolg hebben dat de waterniveaus in het spel niet langer genivelleerd zijn. Bijvoorbeeld omdat extra water op de bodem is terecht gekomen, of omdat een downer-blok een stuk van een dijk heeft afgebroken. Nadat een blok op de bodem is terechtgekomen en het effect ervan is toegepast moet je dus telkens het water nivelleren.

Input van de muis en het keyboard

Een vallende blok moet horizontaal naar de muis bewegen, zodat je kan sturen waar een blok op de bodem terecht komt. Bovendien kan je op een toets duwen om een blok onmiddellijk naar beneden te laten vallen, net zoals in Tetris (bv. de spatiebalk).

Op de website van VPython kan je de documentatie m.b.t. keyboard interactie en muis interactie bestuderen. Hieronder vind je alvast een klein voorbeeld dat een stuk tekst print telkens een toets wordt ingedrukt, en vervolgens een stuk code dat telkens de positie van de muis print. Beide stukken code vind je ook op Canvas.

```
venster = open_display()
while True:
    rate(30)
    if venster.kb.keys:
     toets = venster.kb.getkey()
     print "toets \"" + toets + "\" werd ingedrukt"
```

```
venster = open_display()
while True:
    rate(30)
    positie_muis = venster.mouse.pos
    print positie_muis
```

Verloop van het spel

Het spel begint met een leeg speelveld. De bodemhoogte is overal 1 en de proefbuis is leeg. Na invoer van de speler door op een toets te duwen (of eventueel te klikken met de muis) start het spel.

Tijdens het spel moet je volgende stappen ondernemen:

- laat een willekeurige blok verschijnen indien er geen ander blok in het spel is
- beweeg het blok telkens een stap naar beneden en links/rechts afhankelijk van de positie van de cursor
- controleer of het blok de bodem raakt en verwerk het effect
- nivelleer het water (indien nodig)
- controleer of het spel gedaan is

Uitbreidingen

Kies minstens twee uitbreidingen uit de lijst hieronder en voeg deze toe aan je spel.

- De **positioneringshulp** is een visuele hulp die aangeeft waar een blok op de bodem terecht zal komen. In figuur 1 wordt de positioneringshulp weergegeven als een wit balkje onder de bodem van het speelveld. Zo kan je gemakkelijk zien waar een blok op de bodem zal vallen.
- Een **bom**-spelblok heeft hetzelfde effect als een vuurblok op het land (nl. het ontstaan van een krater) met als bijkomend effect dat er een gat in de bodem ontstaat (bodemhoogte 0). Een bom heeft geen effect op het water. De bom beweegt gewoon door het water en creëert een krater in de bodem onder het water zoals hierboven omschreven. Merk op dat er door het gat in de bodem ook water kan wegstromen.
- Tijdens het spel kan het op een willekeurig moment beginnen **regenen**. Op willekeurige plaatsen vallen dan individuele waterblokjes naar beneden. Regen is beperkt in de tijd, dus na een zelfgekozen tijdsinterval stopt de regenval.
- Telkens je een upper-blok plaatst gaat een **aardbevingsmeter** omhoog. Wanneer de meter zijn maximum bereikt doet zich een aardbeving voor. Je verliest een willekeurige hoeveelheid grond van de bodem, vooral langs de zijkanten van het spel. Een aardbeving kan geen gat in de bodem creëren.
- Zorg voor een **scorebord**. Je mag vrij kiezen hoe je scores telt, maar je moet ergens het aantal meren als een "multiplier" gebruiken. T.t.z. telkens je punten scoort worden die vermenigvuldigd met het aantal meren die op dat moment op het speelveld aanwezig zijn. Je wordt dus beloond voor het aantal meren dat je bouwt.
  Een eenvoudig voorbeeld is:
  - Je scoort 5 punten per blok die je op het spelbord plaatst.
  - Je krijgt $w \times m$ punten voor het verdampen van water d.m.v. een vuurblok. w is de hoeveelheid water in het meer dat je verdampt. m is het aantal meren in het spel. Hier gebruiken we het aantal meren als multiplier.
- **Rubberen eendjes** verschijnen in diepe meren. Hoe diep een meer moet zijn eer er een eendje in verschijnt mag je zelf kiezen. Elk meer kan slechts één eendje bevatten. Elk eendje verdubbelt alle punten die gescoord worden. Dus voor 1 eendje $\times 2$, voor 2 eendjes $\times 4$, etc.

Extra informatie

Op Canvas kan je extra documentatie vinden in de vorm van:

- Een document met de volledige documentatie van het originele spel. Hier staan alle blokken nogmaals uitgelegd alsook de verschillende uitbreidingen, weliswaar voor de originele 3D-versie van Wetrix.
- Een filmpje van een eenvoudige voorbeeldimplementatie.
- De introductiepresentatie van de taak met o.a. extra voorbeelden van het effect van de verschillende blokken.

Af en toe moeten dingen "willekeurig"gebeuren in de taak. Neem een kijkje op de documentatiepagina van de random module van Python. Met de functies uit deze module kan je bv. een willekeurig getal kiezen, een willekeurig element uit een lijst, etc.

Praktische informatie

Tenslotte volgt hier nog de praktische informatie over de taak.

- Je moet de taak **individueel** oplossen! Samenwerken is **niet** toegelaten. Uitwisselen van code wordt als **plagiaat** beschouwd!
- Er is geen aparte deadline om deel 1 van de taak in te dienen. Dien deze samen in met deel 2.
- De deadline is **zondag 6 januari 2019 om 23u59**.
- Indienen doe je door je code in te zenden via "Canvas > Inleiding tot de Computerwetenschappen > Opdrachten > Taak Wetrix".
- Nuttige commentaar[1] in je code is niet verplicht maar wordt aangemoedigd.
- De keuzes die gemaakt werden tijdens de uitvoering van de taak zullen mondeling verdedigd worden tijdens de zittijd in januari.
- De assistenten zijn bereikbaar voor vragen via e-mail: icw@dinf.vub.ac.be
- Jullie kunnen ook steeds terecht bij de studiebegeleider van de computerwetenschappen, Marjon Blondeel. Via e-mail: mblondee@vub.ac.be
- Wacht niet te lang om aan je taak te beginnen!

[1] Als je commentaar toevoegt aan je code, zorg dan dat ze een toegevoegde waarde heeft. Commentaar is dus **geen tekstuele vertaling** van je code.