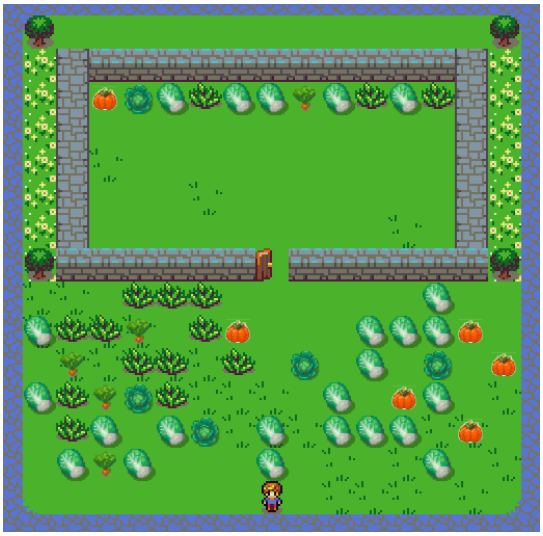
TP SEMAINE 2 - PROGRAMMATION ROBOTIQUE

YANN CHEVALEYRE

L'objectif de ce TP est de vous permettre de prendre rapidement l'environnement `gardenworld`, en écrivant de petits programmes permettant de réaliser différentes tâches dans ce "micro-monde".

Dans ce micro-monde, il y a un jardin entouré de murs de pierre, qui occupe la moitié du terrain. Le personnage principal est un jardinier. Au tout début, le micro-monde ressemble à cela:



1. Introduction

Le programme python utilisant ce micro-monde devra commencer ainsi:

```
from gardenworld import *
init('info2_1')
```

Ensuite, on a plusieurs instructions à disposition (en plus de l'ensemble des instructions python). Le jardinier peut pivoter sur lui-même, avancer tout droit, ramasser des légumes, et les déposer. Le jardinier peut ramasser autant de légumes qu'il veut avant de les déposer (il a un grand panier)

```
avance()        # avance d'une case
tournegauche()  # pivote d'un quart de tour
tournedroite()  # pivote d'un quart de tour (autre sens)
cherche()       # renvoie le nom du légume sous les pieds du jardinier
ramasse()       # ramasse le légume sous les pieds du jardinier
depose()        # dépose l'un des légumes ramassés
obstacle()      # teste s'il y a un obstacle en face du jardinier
```

Ces instructions ont des abbréviations:

```
av(); tg(); td(); ra(); ch(); dp(); ob()
```

Le jardinier peut tester s'il est face à un obstacle avec le code suivant:

```
if obstacle():
    print('je suis bloqué !!!')
else:
    print('la voie est libre')
```

De même, il peut tester s'il y a un légume sous ses pieds avec le code suivant:

```
if cherche():
    print('je sens un truc sous mes pieds !!!')
else:
    print('rien que de la terre...')
```

Si le jardiner a ramassé plusieurs légumes différents, il peut appeler la fonction *depose* avec en paramètre le nom du légume à déposer. Par exemple *depose('citrouille')*

2. Prise en Main

- Ouvrez un éditeur python
- faites un programme qui déplace le jardinier vers des légumes
- essayez *ramasse(), cherche()* lorsque le personnage se trouve sur des légumes. Que font et que renvoient ces fonctions ?
- essayez *avance()* lorsque le jardinier est bloqué contre un mur. Que se passe-t-il ?
- Ecrivez un programme qui fasse ramasser un légume au personnage, et qui le fasse déposer ce légume un peu plus loin
- Ramassez un chou et une carotte, puis déposez la carotte à la place du chou et vis-versa.

3. Exercices sur les Fonctions, Boucles, IF

Créez une fichier *exo1.py* dans l'environnement python Pyzo (ou Geany) pour cet exercice.

- Ecrivez une fonction *av10()* qui appelle 10 fois la fonction *avance()*, en utilisant une boucle. Testez-la
- Ecrivez une fonction *cours(x)* qui appelle x fois la fonction *avance()*. Testez-la
- Ecrivez une fonction *demitour()* et testez la
- Ecrivez une fonction *fonce()* qui appelle la fonction *av()* tant qu'il n'y a pas d'obstacle.
  Note: Cette fonction devra utiliser l'instruction if dans une boucle for, ainsi que l'instruction break qui a pour effet d'arrêter immédiatement une boucle for. Vous pouvez aussi utiliser l'instruction while à la place de if+break.
- Ecrire une fonction *troisTours()* qui rentre dans l'enclos et en fasse le tour 3 fois.
- Ecrire une fonction *afficheLegu()* qui affiche à l'écran le nom du légume qu'il y a sous les pieds du jardiner

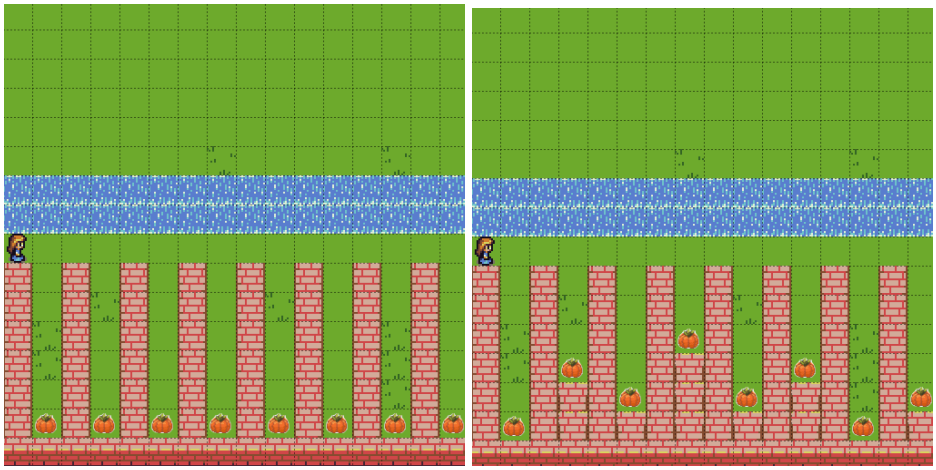
- Ecrivez une fonction *positionne_haut_gauche_du_jardin()* qui positionne le jardinier en haut à gauche de l'enclos.
- Ecrivez une fonction *cherchetout()* qui rentre dans l'enclos et le parcoure entièrement de haut en bas et de droite à gauche. La fonction devra par ailleurs afficher le nom de tous les légumes sur lesquels le jardinier passe
- Ecrivez une fonction *compteLeg()* qui renvoie 1 s'il y a un légume sous les pieds du joeur et 0 sinon.
  Attention, cette fonction n'affiche rien, et ne doit donc pas contenir d'instruction print.
- Ecrivez une fonction *compteFonce()* qui aille tout droit jusqu'au prochain obstacle (comme la fonction *fonce()*), et qui renvoie le nombre de légumes rencontrés sur son trajet.
  Note: Cette fonction appellera compteLeg().
- Ecrivez une fonction *cherchetout2()* en modifiant *cherchetout*, de telle sorte que cette fonction affiche le nombre total de légumes vu pendant son parcours.
  Note: Cette fonction appellera la fonction compteFonce().

4. Déplacements

- Creez des fonction *haut(),bas(),gauche(),droite()* qui permette de diriger le jardinier plus facilement, en le déplaçant d'une case dans la direction voulue.
  Ces fonctions supposerons que le jardinier est orienté vers l'Est, et à l'issue de ces fonctions, il devra encore être orienté vers l'Est
- Ecrivez une fonction *translate(x,y)* qui déplace le jardinier de x case horizontalement (vers la droite) et de y cases verticalement (vers le haut). Si x est négatif, le jardinier se déplacera vers la gauche (un déplacement de -1 case vers la droite est similaire à 1 case à gauche). Idem pour y. Donc par exemple, *translate(1,-1)* déplacera le jardinier d'une case à droite et d'une case vers le bas.
- Optionnel: Ecrire une fonction *aller_a(x,y)* qui déplace le joueur à la colonne x et à la ligne y. Pour cela, la fonction devra stocker dans des variables globales la position courante du joueur.

5. Jardins Labyrinthiques

Il y a deux jardins labyrinthiques. Celui de gauche (couloirs à profondeur fixe) et celui de droite (couloirs à profondeur variable) sur l'image ci-dessous:



Pour charger ces micro-mondes, faite

```
from gardenworld import *
init('laby_fixe')
```

et

```
from gardenworld import *
init('laby_variable')
```

- Sur le labyrinthe à profondeur fixe, faites un programme qui ramasse toutes les citrouilles.
  Attention: tout contact avec l'eau est mortel !!! Et l'eau n'est pas considérée comme un obstacle !!!!
- Faites de même sur le labyrinthe à profondeur variable. Pour éviter de tomber dans l'eau, le jardinier devra compter le nombre de pas qu'il fait avant d'atteindre le fond du couloir (qui est un obstacle), puis refaire le même nombre de pas en sens inverse.