Modélisation et robotique

Exercices 7

1 Révisions du cours

Question A. Types. Quel est le type de chacune des variables suivantes ?

```
xs = [1, 2, 3]
t = (1, 2, 3)
e = {1, 2, 3}
d = {'un': 1, 'deux': 2, 'trois': 3}
```

Lesquels sont des types de données *mutables* ? Pour chacune des variables d'un type mutable dire comment ajouter un élément à ses données. Puis dire comment supprimer un élément.

Question B. Ambiguïté de la notation. Quel est le type de `{}` ?

Question C. Tests. Donner la valeur des booléens.

```
xs = [1, 2, 3]
t = (1, 2, 3)
e = {1, 2, 3}
d = {'un': 1, 'deux': 2, 'trois': 3}
2 in xs
2 in t
2 in e
2 in d
'un' in d
```

2 Programmer Thymio

Question D. Préparatifs. Ouvrir la machine virtuelle, double cliquer sur Mise À Jour, puis, lorsque ce nouveau bouton apparaîtra, sur Thymio Installation (le mot de passe demandé est rien0rien0). Quitter et relancer la session. Connecter le Thymio, et lancer la commande `asebamedulla 'ser:device=/dev/ttyACM0'` (il vous suffit d'ouvrir un terminal et de rappeler cette commande depuis l'historique, avec la touche flèche haut).

Question E. Premiers pas. Pour votre premier programme avec Thymio, nous allons exploiter le capteur de température. Vous pouvez en lire plus sur les capteurs sur cette page : `https://www.thymio.org/fr:thymioapi`

Les évènements n'iront que dans un sens, du Thymio vers votre programme. Créer un nouveau programme Python.

Pour communiquer avec le Thymio, il vous faut importer pythymio (`import pythymio as pt`), et créer un bloc `with pt.thymio(evts1, evts2) as Thym:` où `evts1` est une liste d'événements internes au Thymio que l'on souhaite écouter (pour cet exercice, choisissez juste `['temperature']`) et `evts2` est une liste que vous choisirez vide pour le moment. Dans cette instruction `Thym` est un surnom choisi librement pour parler du Thymio. À l'intérieur de ce bloc il faut créer une fonction qui traitera les événements en provenance du Thymio. Appelons cette fonction `progression`. Nous attacherons ensuite cette fonction à la boucle événementielle du Thymio en utilisant l'instruction : `Thym.loop(progression)`.

La fonction progression va être appelée à chaque événement du Thymio jusqu'à ce qu'on décide d'arrêter le programme avec Ctrl-C.

Cette fonction progression recevra en entrée trois arguments : un identifiant d'événement (que nous n'utiliserons pas), le nom de l'événement prefixé par 'fwd.', et une liste contenant les arguments de l'événement. Dans le cas de l'événement interne 'temperature' nous recevrons une liste avec une seule valeur, la température actuelle en degrés Kelvin. Cet événement est déclenché à une fréquence fixe de 1Hz, c'est parfait pour effectuer un affichage. Affichez la température lue par le Thymio à chaque réception de l'événement `fwd.temperature`.

Question F. Filtrer par événement. La liste evts1 ne contient qu'un seul événement. Que se passe t'il si vous ajoutez à cette liste d'autres événements, par exemple `button.center` ? Faites en sorte que seul l'événement `fwd.temperature` déclenche un affichage. Affichez également 'bouton centre' à chaque fois que votre progression reçoit l'événement `fwd.button.center`. Quel est le problème avec cet événement ?

Question G. Accéléromètre 1. Modifier votre programme précédent pour n'écouter que l'événement interne `acc` avec lequel nous recevrons la liste des trois valeurs lues par l'accéléromètre. Cet événement est déclenché à une fréquence fixe de 16Hz.

C'est un peu trop rapide pour un affichage. Nous allons donc utiliser un compte à rebours pour ne faire l'affichage des valeurs que toutes les secondes. Il vous faudra utiliser un compteur de délai de façon à attendre 16 fois que l'événement ait été déclenché avant un nouvel affichage.

Vous aurez besoin d'une variable globale, que la fonction progression devra pouvoir modifier. Vous rappelez vous comment faire ?

Comme cette situation est appelée à se reproduire, plutôt que de multiplier les variables globales, créez en une seule qui contiendra tout l'état du programme entre deux appels à `progression`. Pour cela créez un dictionnaire `state` et utilisez des clés différentes pour les différentes valeurs que vous voulez stocker. Par exemple, `state['delay']` pourra contenir un entier valant initialement 15, decrémenté à chaque événement `fwd.acc` et qui, lorsqu'il atteindra zéro, déclenchera l'affichage puis se remettra à 15.

Question H. Gravité. Quelle est la valeur de l'attraction terrestre selon Thymio ?

Question I. Accéléromètre 2. Utilisez Thymio comme télécommande pour vous déplacer dans gardenworld. Tenir Thymio droit vous fera avancer d'une case par seconde, le basculer vers l'avant le stoppera, le tourner comme un volant un peu incliné vers la gauche ou la droite le fera tourner dans la direction correspondante.

Question J. Bouton(s) quitter. L'événement interne `buttons` a une fréquence de 100Hz (de quoi bien occuper votre machine virtuelle si vous faites un affichage à chaque réception). Il retourne une liste de 5 valeurs, 0 ou 1, qui encode le fait que les boutons sont pressés ou non. Utilisez la technique du délai employée précédemment pour ne faire un affichage que tous les 80 centièmes de seconde. Trouvez à quoi correspond chaque bouton, puis faites en sorte de quitter la boucle événementielle avec `Thym.stop()` lorsque on appuie simultanément sur le bouton gauche et le bouton droite.

Question K. Simon. Le jeu Simon consiste en observer une séquence de boutons tirée au hasard puis reproduire la même séquence. La difficulté progressive se traduit par la longueur de la séquence. Le but du jeu est d'atteindre la plus haute difficulté possible. Pour que le jeu soit agréable la séquence de bouton s'accompagne de sons et de couleurs, un son et une couleur différente par bouton. Le bouton central n'entrera pas dans la composition des séquences. Commencez par écrire une fonction qui prend en entrée un entier et retourne une séquence de lettres tirées parmi U, D, L et R (up/haut, down/bas, left/gauche, right/droite) de longueur cet entier. Testez votre fonction sur un programme ne nécessitant pas Thymio.

Question L. Simon 2. Vous allez utiliser Thymio pour jouer une séquence de boutons, en allumant des leds en fonction de la direction des boutons (haut, bas, gauche, droite). Pour cela vous aurez besoin de créer des événements qui iront du programme au Thymio et pour lesquels le Thymio saura quel code exécuter dans son propre programme. Nous avons préparé cela pour vous. Dans un nouveau programme pour Thymio, affichez simplement la valeur retournée par `pt.customEvents('circle')`. Ce que vous voyez est une liste de paires consitutées d'un événément et du code qu'exécutera le Thymio au moment où il recevra cet événement. Passez cette liste comme second argument de la fonction `pt.thymio()`. Les événements qui vous intéressent sont : `circle.right`, `circle.left`, `circle.front`, `circle.back` et `circle.off`. Pour envoyer un événement au Thymio, par exemple `circle.off`, vous pouvez utiliser `Thym.send_event('circle.off')`.

Question M. Son et lumière. Avec `pt.customEvents('circle', 'music', 'colors')` vous aurez le son et la lumière !

Question N. Simon 3. Une fois que votre programme a choisi une séquence et l'a joué à une personne, cette personne doit reproduire la séquence à l'identique (sans toutefois en respecter le rythme). Cette personne peut se tromper ou réussir, issue qui ajustera la difficulté pour la fois suivante, après quoi le programme se met en attente. Avant de commencer la lecture d'une nouvelle séquence le programme attend que la personne appuie sur le bouton central. Imaginez un automate à cinq états "ready", "read", "compare", "won", "lost" et les transitions que vous aurez à faire entre ces états. Pensez à introduire des délais entre les étapes du programme. Complétez le programme.

Question O. Variante. Plutôt que de tirer une nouvelle séquence à chaque succès, rallongez la séquence précédente.