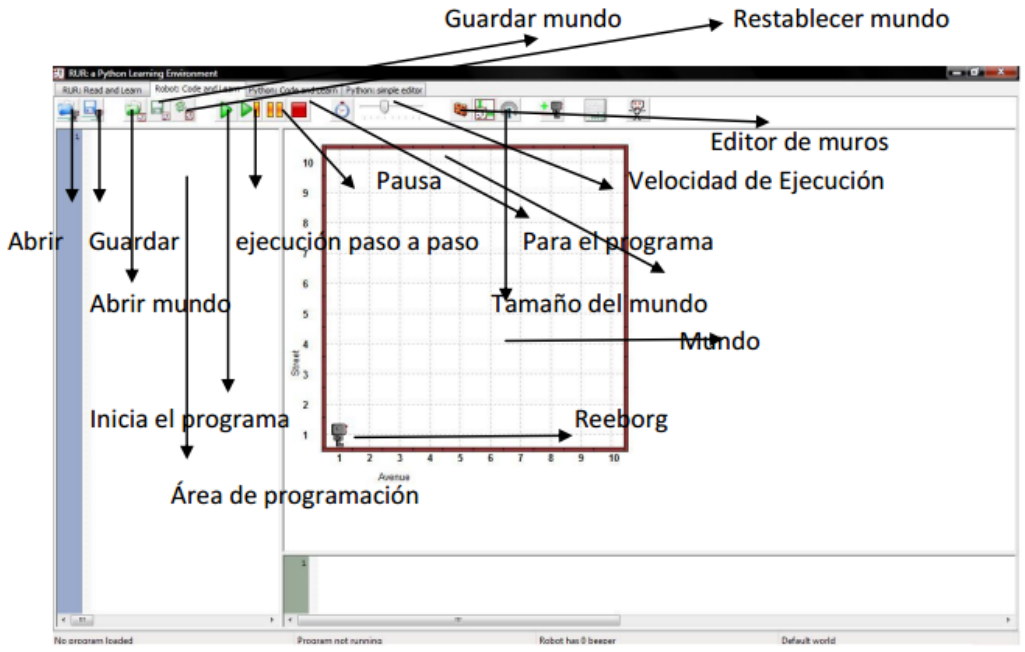
Pràctica grup: rur-ple

RUR-PLE és un entorn per facilitar el aprenentatge per programar utilitzant el llenguatge de programació Python. compta amb una interfície dinàmica que consisteix en un rectangle envoltat per parets. les línies horitzontals es diuen avingudes i les línies verticals es diuen carrers. Per aquests carrers i avingudes es desplaça un robot anomenat Reeborg. Reeborg és l'encarregat d'executar totes les instruccions que se li indiquin, com moure a través del món, recollir beepers que són indicadors que Reeborg pot detectar, els pot agafar i els pot tornar a posar, esquivar parets, no pot xocar amb les parets i una infinitat de activitats més.

Reeborg es mou

Reeborg compta amb 3 comandaments que pot executar. La resta és qüestió de la teva imaginació per combinar aquestes comandes amb condicions, repeticions i cicles per aconseguir el que sigui. Aquests tres ordres són:

- **move ()** serveix perquè Reeborga avanç una sola vegada.
- **turn_left ()** Reeborg només pot girar a l'esquerra, així que hauràs de jugar amb aquesta comanda perquè Reeborg avanç.
- **turn_off()** Reeborg necessita apagar-se quan acabi les seves instruccions. aquest comandament s'apaga a Reeborg.



La instrucció **while** ens serveix per "mentre faci alguna cosa passi o faci això" és una condició, i és molt comú a l'hora de topar-nos amb murs. Reeborg desplega un missatge d'error a l'hora de xocar amb un mur, per la qual cosa col·loca el while i la instrucció de quan no hi hagi mur que avanci. per exemple:

```
while front_is_clear ():
    move ()
```

Això defineix que Reeborg es va a moure al capdavant sempre que no hi hagi mur. I ha la contrapart el **while not** que és el contrari al while.
La següent condició és el **if** és la típica condicional, si tenim una situació i és veritable es donar un procés però si és falsa serà un altre procés. per aquesta condicional hi les seves contraparts **elif** i **else**. Que reflecteixen "en cas contrari ..."
Si volem creuar a la dreta:

turn_left ()

turn_left ()

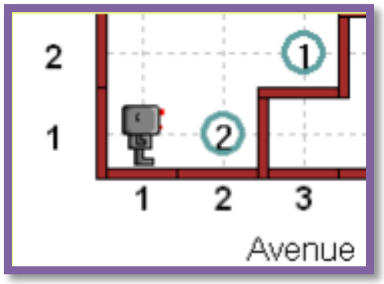
turn_left ()

Les repeticions són molt útils en casos que tinguem la mateixa instrucció diverses vegades. Aquestes es poden simplificar en una sola línia i no importa quantes repeticions hagin de tenir. Per exemple en l'exemple anterior tenim 3 turn_left () això es pot simplificar mitjançant: *repeat ("la instrucció", # de vegades)*. Exemple:

repeat (turn_left, 3)

Creació d'un mon:

Al món vostè podrà construir parets com a obstacles i el robot pot a més posar i treure petits **beepers** quan vostè li indiqui.



Els petits cercles blaus són beepers El nombre indica quants n'hi ha. Les parets són les corintas.

INSTRUCCIONS:

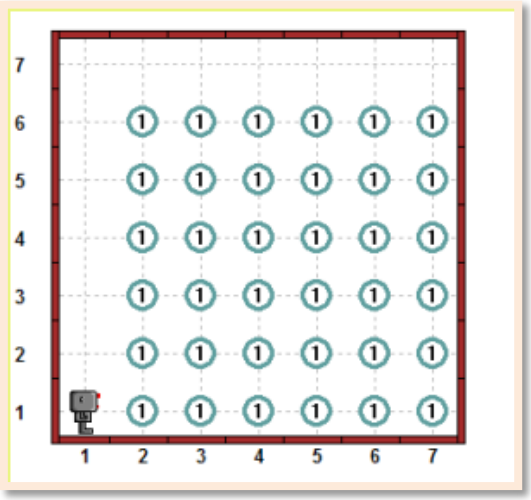
INSTRUCCIONS LINUX	SIGNIFICAT
#	Poseu aquest signe per fer comentaris que no comptin com una instrucció. Ex .: #Programa Un
move( )	Mourà un pas endavant al robot
turn_off()	Apaga a robot quan acaba un procés. Aquesta instrucció és totalment necessària posar-la sempre.
turn_left( )	Li indicarà el robot a girar cap a l'esquerra
put_beeper( )	Indica al robot que deixi un beeper en la posició en on està parat.
pick_beeper( )	Indica al robot que aixequi un beeper de la posició on està parat.
on_beeper()	Equivalent to: next_to_a_beeper()
front_is_clear() **left_is_clear()** **rigth_is_clear()**	Indica que no hi ha obstacles en la direcció escollida
repeat (procedimiento, # de veces)	Serveix per a que una instrucció es repeteix el número de vegades que s'indiquin. Ex.: repeat (move ( ) , 5) .
def nombreprocedimiento( ):	Serveix per definir una nova instrucció, per a evitar repeticions.
if situacio( ):	if es una condició que significa "Si", l indiquem al robot fer algo sempre i quan es compleix alguna situació.
else :	L'indicarà al robot una nova instrucció SI NO es pot complir la condició o situació anterior.
elif situacio ( ):	És una instrucció per a encadenar varies condicions.
pass	Quan es treballa amb condicions, serveix per ignorar alguna.
not	S'utilitza després d'un if o una altra instrucció per indicar que aquesta situació no s'està complint o és negativa.
while	És una instrucció que indica que se segueixi fent una acció fins que es compleixi una condició.

INSTRUCCIONS WINDOWS:

- **move()**
- **turn_left()**
- **front_is_clear()**
- **left_is_clear()**
- **right_is_clear()**
- **facing_north()**
- **pick_stone()**

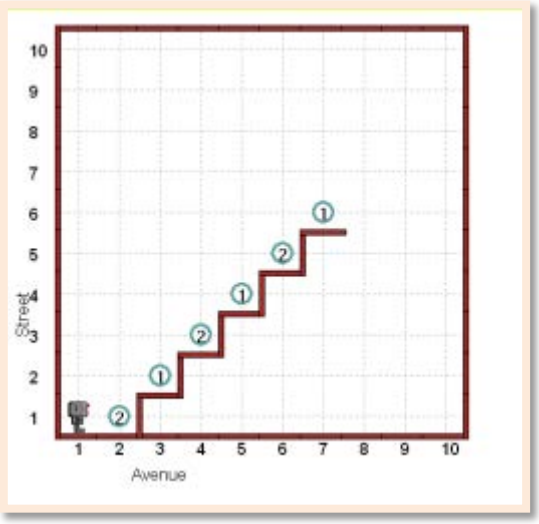
- **put_stone()**
- **on_stone()**
- **got_stone()**
- **roll_dice()**
- **input_string()**
- **input_int()**
- **print()**

Exercici 1:



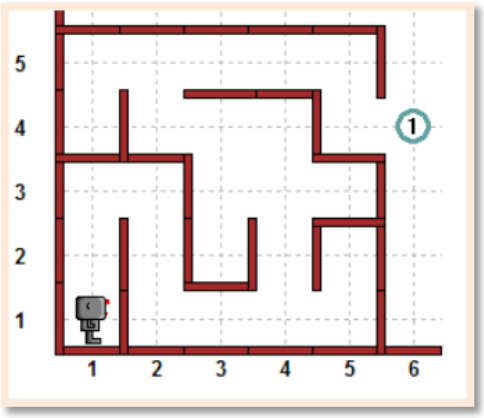
Suposem aquest món, suposi que la seva missió és fer que Reeborg reculli tots els beepers, amb les instruccions que ja sap, com procediria?
Mostreu el resultat per pantalla.

EXERCICI 2:



Suposem que se li presenta aquest món i el seu repte és recollir tots els beepers i tornar al robot a la posició original, quin és el següent pas?
Mostreu el resultat per pantalla.

EXERCICI 3:



Suposeu que teniu aquest món, la dificultat va augmentant. La seva missió és recollir el beeper que està al final. Com avançaria pel laberint?

EXERCICI 4:

Creeu un mon i programeu el resultat d'aquest mon. Sabent que:

1. Cada vegada que agafi un beeper, la puntuació del robot augmentarà en 1 i mostrarà un missatge d'èxit en el programa.
2. Poseu alguna regla