Day - 6

**Functions**

Code Blocks (indentation), Functions, While loops

**6.1 Function**

* Defining: Use "**def**" key word **function\_name** with "**()**" and "**:**", create block by indenting.
* Call: Similar to other languages. Just **function\_name()**.
* Exercise 6.1 Reebog (Weird project):

|  |  |  |
| --- | --- | --- |
| def right():  left()  left()  left() | def jump():  move()  left()  move()  right()  move()  right()  move()  left() | for I in range(1, 7) :  jump() |

**6.2 Indentation**

Indentation is very important in Python. It plays the rule of {}.

* You can use "**spaces**" or "**tabs**". Usually "*tabs*" are used.
* But Python prefers "**spaces**". For minimize the file.
* Don’t mix use "**spaces**" or "**tabs**" in same block.
* Commonly **4-Spaces** are used for indent.
* Code editor setting: Hitting "tab" makes **4-Spaces** indent. (default in some editors)

**6.3 While loop**

**while** condition :

block

* **for loop** is good for iteration but **while loop** is good for checking against a condition

|  |  |
| --- | --- |
| * Exercise 6.2: Reeborg Random Hurdle Challenge. Use Following conditions along with two defined function **left()** and **move()**.      * Notice the **redefined** **jump** function. |  |

M=move, L=left, R=right, J=jump, **def** R(): L L L, def J(): L M R M R M L

**while** ! at\_goal() :

**if** wall\_at\_front(): J()

**else**: M

* Exercise 6.3: Reeborg Random Hurdle Challenge with *variable* *height*. Redefine **J = jump**

M=move, L=left, R=right, J=jump,

**def** up():

**while** wall\_on\_right() : M

**def** down():

**while** front\_is\_clear() : M

**def** R(): L L L,

**def** J(): L up() R M R down() L

**while** ! at\_goal() :

**if** wall\_at\_front(): J()

**else**: M

* Exercise 6.4: Reeborg Maze solution. Escape the maze. Hint: Track the "*Right-side-wall*"

|  |  |
| --- | --- |
| Web solution | Instructor Solution |
| **def** turn\_right():  turn\_left()  turn\_left()  turn\_left()  *#To avoid infinite loop:*  **while** front\_is\_clear():  move()  turn\_left()  **while** **not** at\_goal():  **if** wall\_on\_right():  **if** wall\_in\_front():  turn\_left()  **else**:  move()  **else**:  turn\_right()  move() | **def** turn\_right():  turn\_left()  turn\_left()  turn\_left()  *#To avoid infinite loop:*  **while** front\_is\_clear():  move()  turn\_left()  **while** **not** at\_goal():  **if** right\_is\_clear():  turn\_right()  move()  **elif** front\_is\_clear():  move()  **else**:  turn\_left() |

* Debugging: Test your code with given 3-cases.

**Appendix Keras REEBORGS WORLD solutions**

**6.4 Reeborg's World**

Reeborg's World is a free "Karel the robot" type of environment used to teach programming, using either Python or Javascript.

* **Exercise 6.1 Hurdles with constant-height**

def turn\_right():

turn\_left()

turn\_left()

turn\_left()

while not at\_goal():

move()

turn\_left()

move()

turn\_right()

move()

turn\_right()

move()

turn\_left()

* **Exercise 6.2 Hurdles with constant-height and variable-distance**

def turn\_right():

turn\_left()

turn\_left()

turn\_left()

def jump():

turn\_left()

move()

turn\_right()

move()

turn\_right()

move()

turn\_left()

while not at\_goal():

if wall\_in\_front():

jump()

else:

move()

* **Exercise 6.3 Hurdles with height**

#<mycode>

def turn\_right():

turn\_left()

turn\_left()

turn\_left()

def jump():

turn\_right()

move()

turn\_right()

move()

while not at\_goal():

while wall\_on\_right() and wall\_in\_front:

if wall\_in\_front():

turn\_left()

elif wall\_on\_right:

if at\_goal():

done()

else:

move()

jump()

#<angela's code>

|  |  |
| --- | --- |
| def turn\_right():  turn\_left()  turn\_left()  turn\_left()  def jump():  turn\_left()  while wall\_on\_right():  move()  turn\_right()  move()  turn\_right()  while front\_is\_clear():  move()  turn\_left()    while not at\_goal():  if wall\_in\_front():  jump()  else:  move() | def turn\_right():  turn\_left()  turn\_left()  turn\_left()  def up():  while wall\_on\_right():  move()  def down():  while front\_is\_clear():  move()    def jump():  turn\_left()  up()  turn\_right()  move()  turn\_right()  down()  turn\_left()  while not at\_goal():  if wall\_in\_front():  jump()  else:  move() |

* **Final Project : Escaping the Maze**

Algorithm: following the right edge of the wall.

def turn\_right():

turn\_left()

turn\_left()

turn\_left()

#<mycode>

while not at\_goal():

if wall\_on\_right():

if wall\_in\_front():

turn\_left()

elif front\_is\_clear():

move()

else:

turn\_right()

move()

#<angela's code>

def turn\_right():

turn\_left()

turn\_left()

turn\_left()

while not at\_goal():

if right\_is\_clear():

turn\_right()

move()

elif front\_is\_clear():

move()

else:

turn\_left()

Special case: if frint and right is clear. Middle of space.

def turn\_right():

turn\_left()

turn\_left()

turn\_left()

while front\_is\_clear():

move()

turn\_left()

while not at\_goal():

if wall\_on\_right():

if wall\_in\_front():

turn\_left()

else:

move()

else:

turn\_right()

move()