

LAPORAN PRAKTIKUM

“TUGAS 2”



Dosen Pengampu :

I Gede Agung Sri Sidhiamantra, S.Kom., M.Kom.

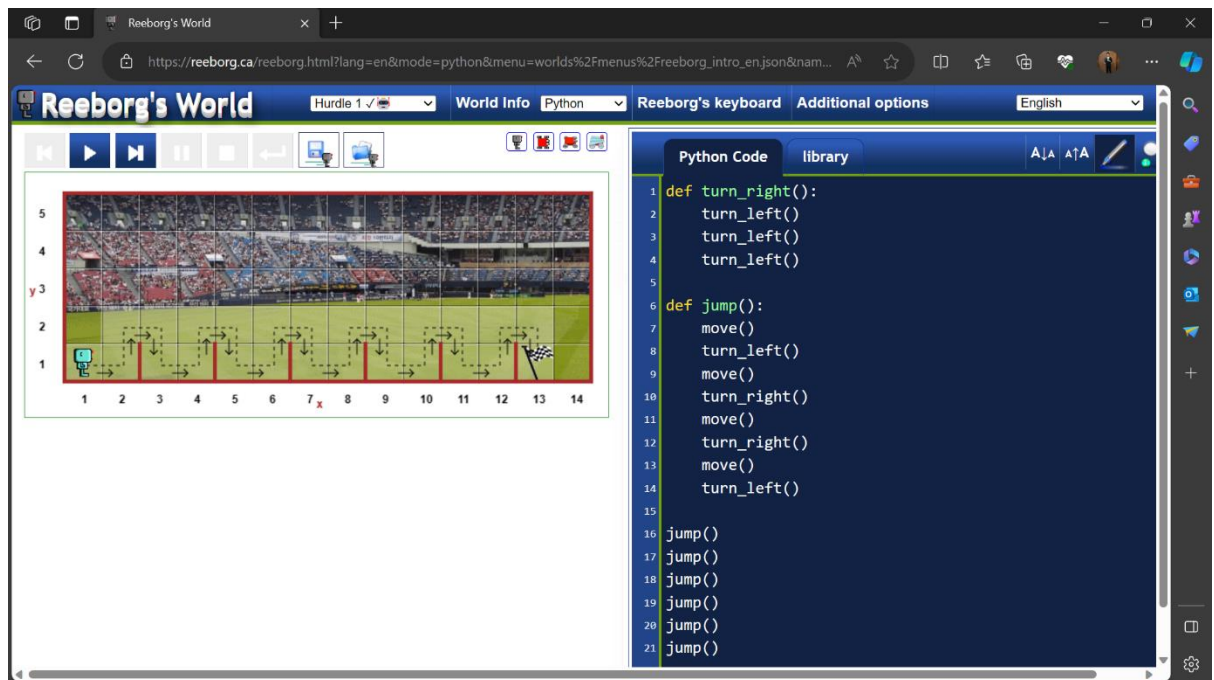
DISUSUN OLEH :

KELOMPOK 3
2023F



MADA PERMATA (23091397179)
ERLINDA MEILANA (23091397184)
WILDAN HABIBI (23091397212)

**PROGRAM STUDI D4 MANAJEMEN INFORMATIKA
FAKULTAS VOKASI
UNIVERSITAS NEGERI SURABAYA
2023**

Hurdle 1



Source Code

```
Python Code library A⌵ A⌴  
```

```
1 def turn_right():
2     turn_left()
3     turn_left()
4     turn_left()
5
6 def jump():
7     move()
8     turn_left()
9     move()
10    turn_right()
11    move()
12    turn_right()
13    move()
14    turn_left()
15
16 jump()
17 jump()
18 jump()
19 jump()
20 jump()
21 jump()
```

Penjelasan Step By Step

1. Definisi Fungsi “turn_right”:

```
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()
```

Fungsi ini berisi tiga pemanggilan fungsi “turn_left()”, yang menunjukkan bahwa “turn_right()” sebenarnya adalah operasi “turn_left()” yang dilakukan tiga kali. Ini digunakan agar robot dapat membuat pergerakan 90 derajat ke kanan (belok ke kanan).

2. Definisi Fungsi “jump”:

```
def jump():  
    move()  
    turn_left()  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move()  
    turn_left()
```

Fungsi ini mendefinisikan cara objek "melompat". Untuk melakukannya, fungsi memanggil fungsi “move()” untuk maju, “turn_left()” untuk belok ke kiri, dan turn_right() untuk berputar belok ke kanan.

3. Pemanggilan Fungsi “jump()” Sebanyak 6 kali:

```
jump()  
jump()  
jump()  
jump()  
jump()  
jump()
```

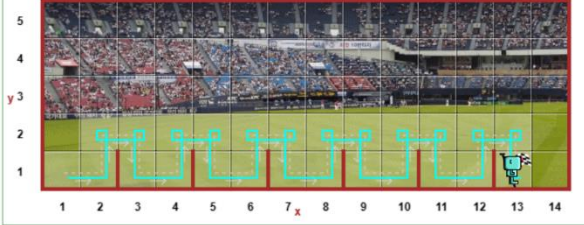
Pada bagian ini, fungsi “jump()” dipanggil secara berulang enam kali. Oleh karena itu, objek akan menjalankan pola pergerakan yang didefinisikan dalam fungsi “jump()” sebanyak enam kali.

Output

Reeborg's World

Hurdle 1 ✓ World Info Python Reeborg's keyboard Additional options English

92/92



Python Code library A|A A

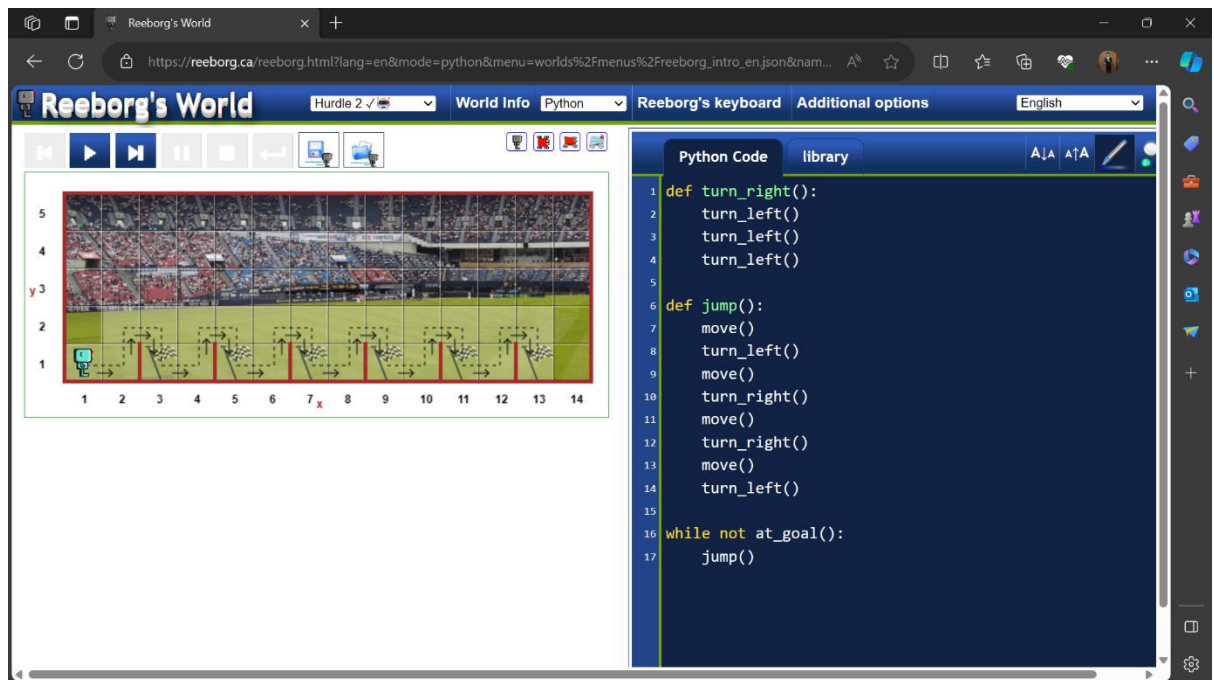
```
1 def turn_right():
2     turn_left()
3     turn_left()
4     turn_left()
5
6 def jump():
7     move()
```

Reeborg says: I'm done!

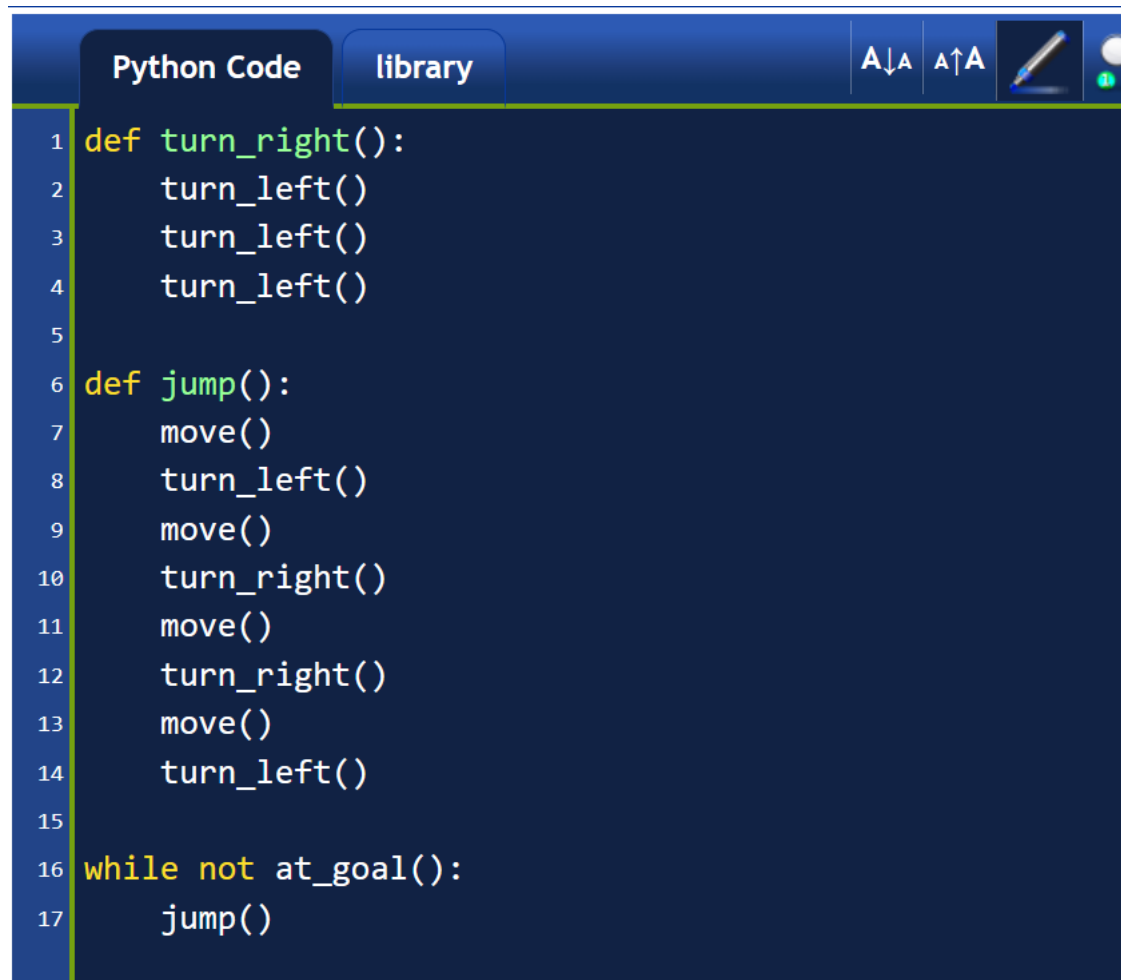
Reeborg is at the correct x position. ✓
Reeborg is at the correct y position. ✓

```
14 turn_left()
15
16 jump()
17 jump()
18 jump()
19 jump()
20 jump()
21 jump()
```

Hurdle 2



Source Code



Penjelasan Step By Step

1. Definisi Fungsi “turn_right”:

```
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()
```

Fungsi ini berisi tiga pemanggilan fungsi “turn_left()”, yang menunjukkan bahwa “turn_right()” sebenarnya adalah operasi “turn_left()” yang dilakukan tiga kali. Ini digunakan agar robot dapat membuat pergerakan 90 derajat ke kanan (belok ke kanan).

2. Definisi Fungsi “jump”:

```
def jump():  
    move()  
    turn_left()  
    move()  
    turn_right()  
    move()  
    turn_right()  
    move()  
    turn_left()
```

Fungsi ini mendefinisikan cara objek "melompat". Untuk melakukannya, fungsi memanggil fungsi “move()” untuk maju, “turn_left()” untuk belok ke kiri, dan turn_right() untuk berputar belok ke kanan.

3. Perulangan Utama:

```
while not at_goal():  
    jump()
```

Ini adalah perulangan utama yang berjalan selama robot belum mencapai garis finish. Di dalamnya ada fungsi “jump()” yang di panggil, maka proses definisi fungsi “jump” yang di atas akan diulang sampai robot mencapai garis finish.

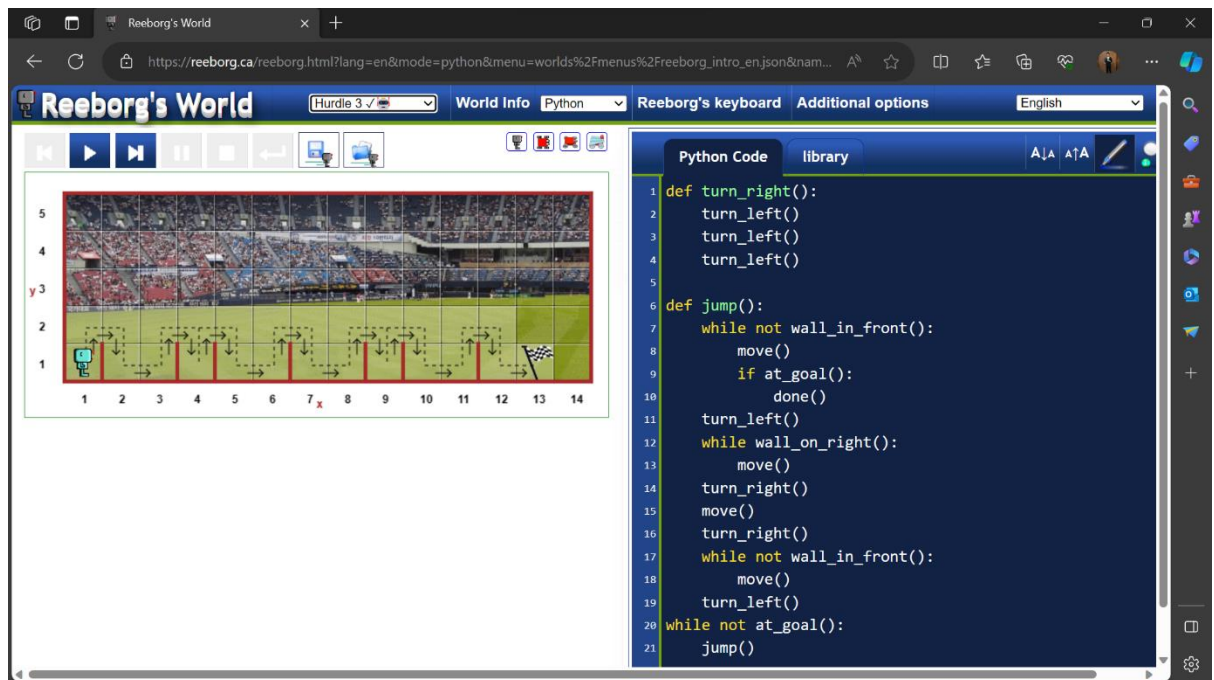
Output

The screenshot displays the Reeborg's World Python environment. On the left, a 14x5 grid represents the arena. A small robot is positioned at (5, 1). A light blue path indicates the robot's movement: it moves left from (5, 1) to (2, 1), then up to (2, 3), then right to (4, 3), then down to (4, 1), and finally right to (5, 1). The arena is filled with a crowd of spectators. The top navigation bar includes 'Hurdle 2', 'World Info', 'Python', 'Reeborg's keyboard', 'Additional options', and 'English'. The right panel shows the Python code editor with the following code:

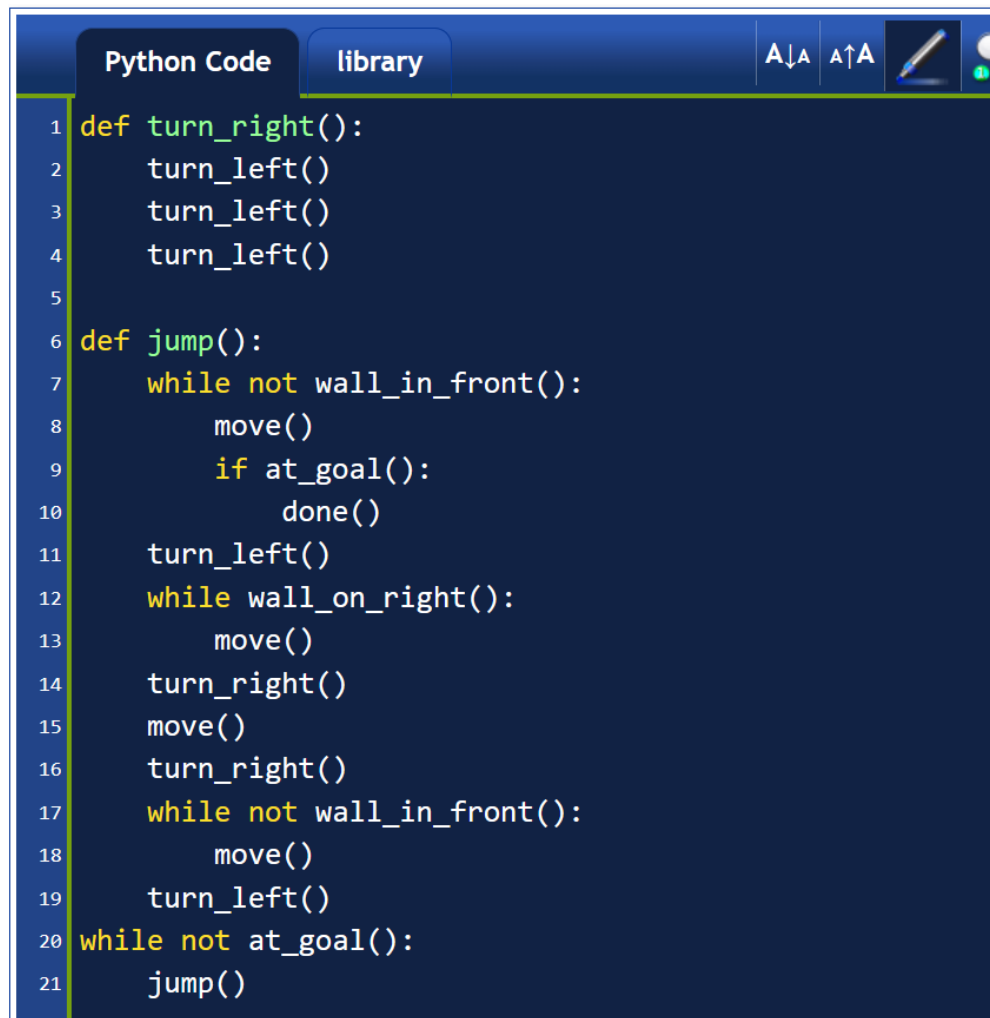
```
1 def turn_right():
2     turn_left()
3     turn_left()
4     turn_left()
5
6 def jump():
7     move()
8
9
10
11
12
13
14 turn_left()
15
16 while not at_goal():
17     jump()
```

A green notification box in the center-right of the code editor displays the message: "Reeborg says: I'm done!". Below this, two green checkmarks confirm the robot's position: "Reeborg is at the correct x position." and "Reeborg is at the correct y position."

Hurdle 3



Source Code



Penjelasan Step By Step

1. Definisi Fungsi “turn_right”:

```
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()
```

Fungsi ini berisi tiga pemanggilan fungsi “turn_left()”, yang menunjukkan bahwa “turn_right()” sebenarnya adalah operasi “turn_left()” yang dilakukan tiga kali. Ini digunakan agar robot dapat membuat pergerakan 90 derajat ke kanan (belok ke kanan).

2. Definisi Fungsi “jump”:

```
def jump():  
    while not wall_in_front():  
        move()  
        if at_goal():  
            done()  
    turn_left()  
    while wall_on_right():  
        move()  
    turn_right()  
    move()  
    turn_right()  
    while not wall_in_front():  
        move()  
    turn_left()
```

Fungsi di atas memiliki beberapa bagian:

1. Perulangan pertama (“while not wall_in_front()”)

Perulangan tersebut digunakan agar robot terus bergerak maju selama tidak ada dinding di depan dengan memanggil fungsi “move()”. Jika robot sudah mencapai tujuan (“at_goal()”), maka akan memanggil fungsi “done()” dan robot atau program akan berhenti secara otomatis.

2. Belok ke kiri (“turn_left()”)

Setelah keluar dari perulangan pertama, panggil fungsi “turn_left()” satu kali. Fungsi tersebut digunakan agar robot melakukan belok ke kiri jika dinding tepat di depan, jika dinding belum tepat di depan maka akan kembali melakukan perulangan pertama dengan memanggil fungsi “move()”.

3. Perulangan kedua (“while wall_on_right()”)

Perulangan tersebut digunakan agar robot terus bergerak maju selama dinding ada di sebelah kanan dengan memanggil fungsi “move()”.

4. Putaran kembali (“turn_right()”, “move()”, “turn_right()”)

Setelah keluar dari perulangan kedua, panggil fungsi “turn_right()” satu kali untuk belok ke kanan, “move()” satu kali untuk maju ke depan, dan “turn_right()” satu

kali lagi untuk belok ke kanan. Fungsi tersebut digunakan jika robot sudah melewati dinding yang berada di sebelah kanan dan robot akan melakukan putar balik.

5. Perulangan ketiga (“while not wall_in_front()”)

Perulangan tersebut digunakan agar robot terus bergerak maju selama tidak ada dinding di depan dengan memanggil fungsi “move()”.

6. Belok ke kiri terakhir (“turn_left()”)

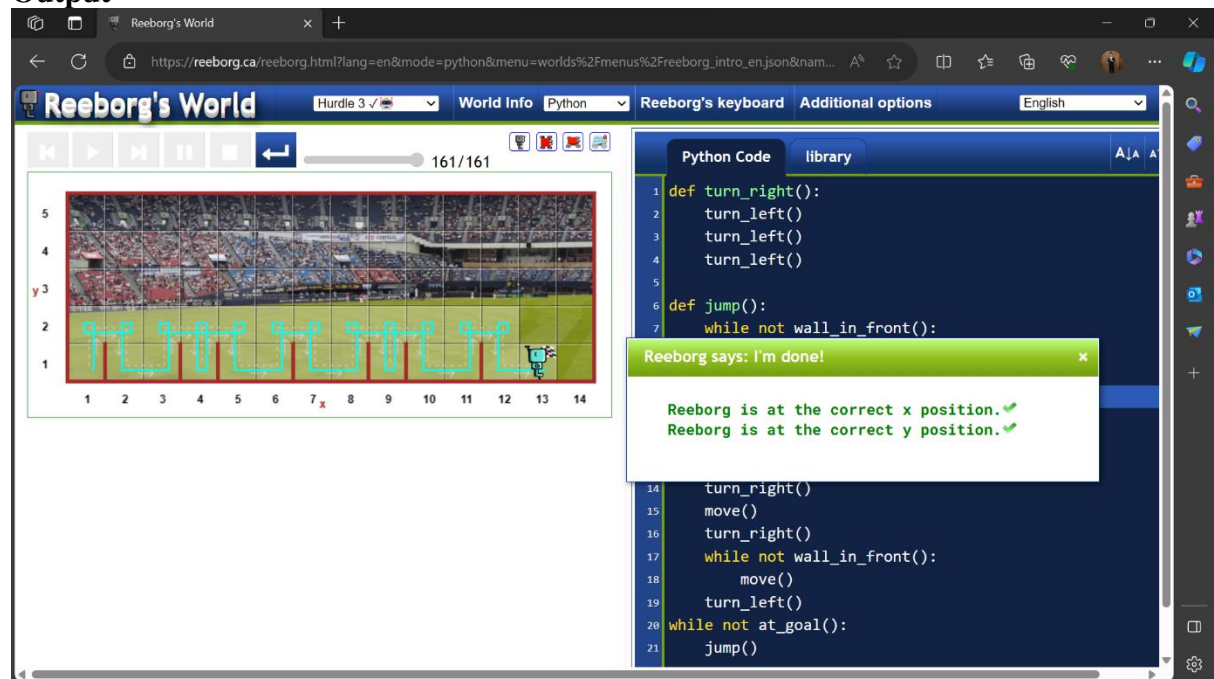
Setelah keluar dari perulangan pertama, panggil fungsi “turn_left()” satu kali. Fungsi tersebut digunakan agar robot melakukan belok ke kiri jika dinding tepat di depan, jika dinding belum tepat di depan maka akan kembali melakukan perulangan ketiga dengan memanggil fungsi “move()”.

3. Perulangan Utama:

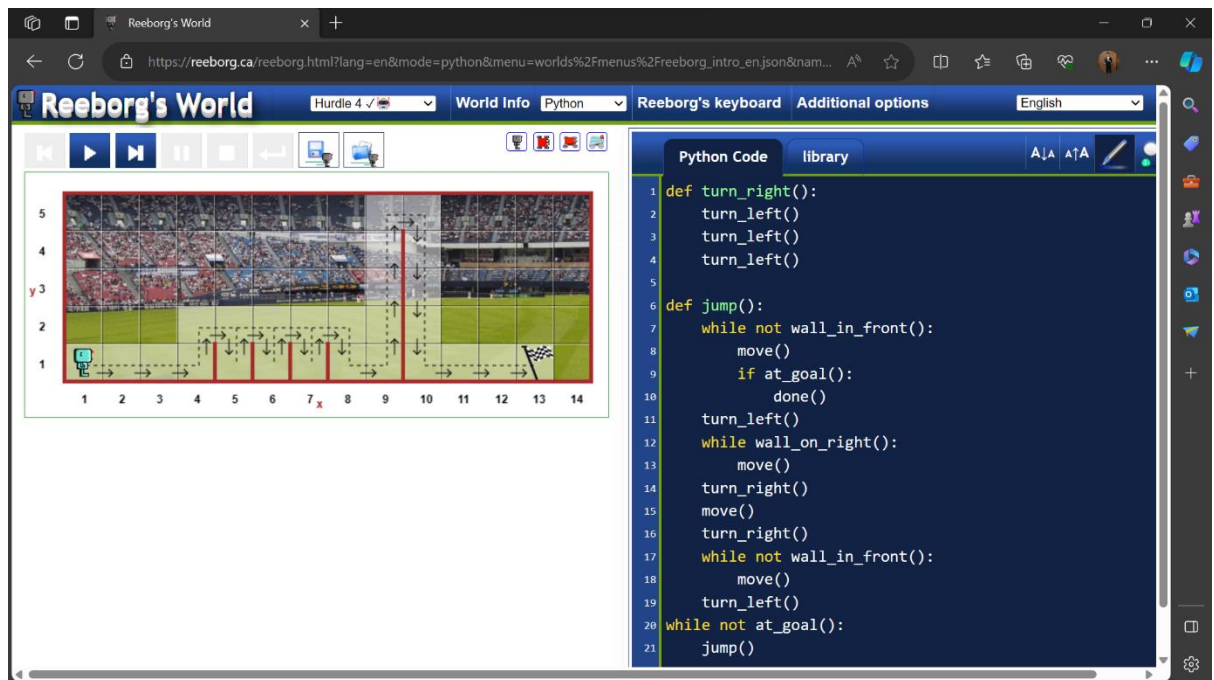
```
while not at_goal():  
    jump()
```

Ini adalah perulangan utama yang berjalan selama robot belum mencapai garis finish. Di dalamnya ada fungsi “jump()” yang di panggil, maka proses definisi fungsi “jump” yang di atas akan diulang.

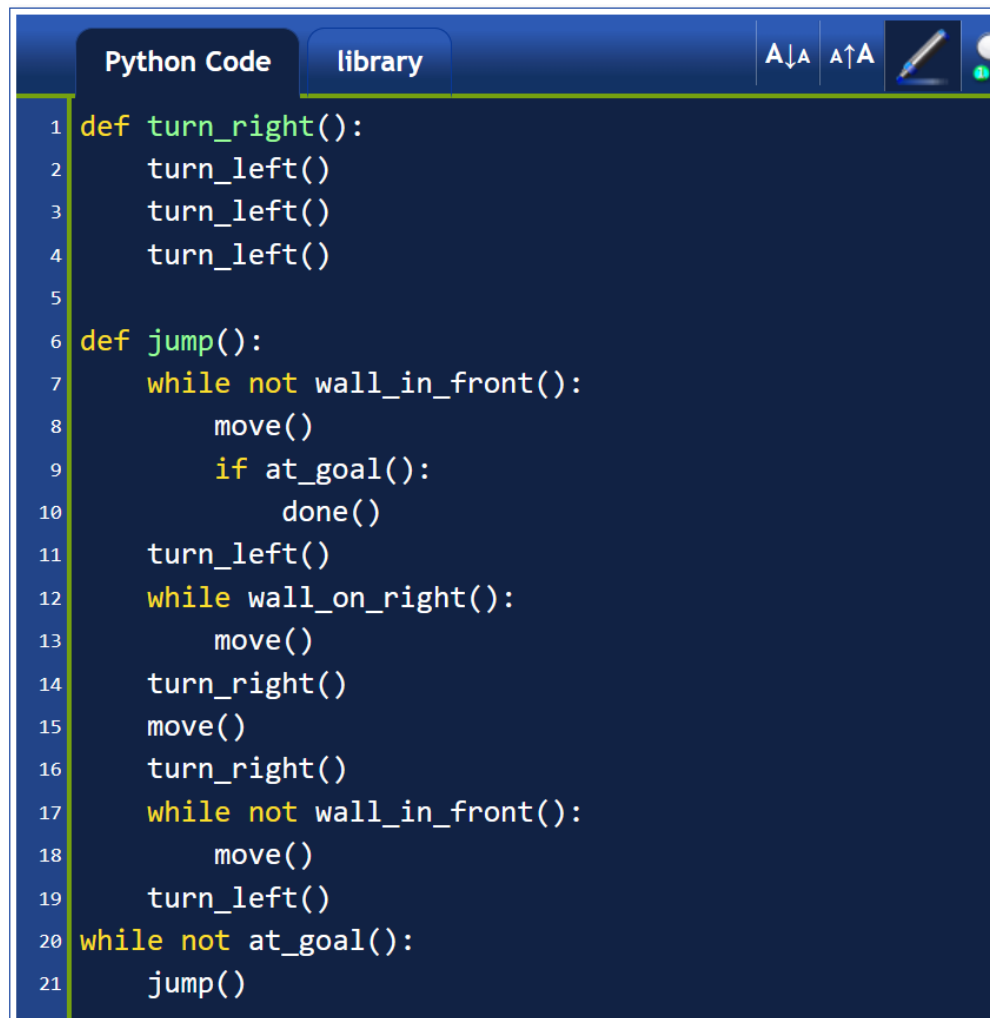
Output



Hurdle 4



Source Code



Penjelasan Step By Step

1. Definisi Fungsi “turn_right”:

```
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()
```

Fungsi ini berisi tiga pemanggilan fungsi “turn_left()”, yang menunjukkan bahwa “turn_right()” sebenarnya adalah operasi “turn_left()” yang dilakukan tiga kali. Ini digunakan agar robot dapat membuat pergerakan 90 derajat ke kanan (belok ke kanan).

2. Definisi Fungsi “jump”:

```
def jump():  
    while not wall_in_front():  
        move()  
        if at_goal():  
            done()  
    turn_left()  
    while wall_on_right():  
        move()  
    turn_right()  
    move()  
    turn_right()  
    while not wall_in_front():  
        move()  
    turn_left()
```

Fungsi di atas memiliki beberapa bagian:

1. Perulangan pertama (“while not wall_in_front()”)

Perulangan tersebut digunakan agar robot terus bergerak maju selama tidak ada dinding di depan dengan memanggil fungsi “move()”. Jika robot sudah mencapai tujuan (“at_goal()”), maka akan memanggil fungsi “done()” dan robot atau program akan berhenti secara otomatis.

2. Belok ke kiri (“turn_left()”)

Setelah keluar dari perulangan pertama, panggil fungsi “turn_left()” satu kali. Fungsi tersebut digunakan agar robot melakukan belok ke kiri jika dinding tepat di depan, jika dinding belum tepat di depan maka akan kembali melakukan perulangan pertama dengan memanggil fungsi “move()”.

3. Perulangan kedua (“while wall_on_right()”)

Perulangan tersebut digunakan agar robot terus bergerak maju selama dinding ada di sebelah kanan dengan memanggil fungsi “move()”.

4. Putaran kembali (“turn_right()”, “move()”, “turn_right()”)

Setelah keluar dari perulangan kedua, panggil fungsi “turn_right()” satu kali untuk belok ke kanan, “move()” satu kali untuk maju ke depan, dan “turn_right()” satu

kali lagi untuk belok ke kanan. Fungsi tersebut digunakan jika robot sudah melewati dinding yang berada di sebelah kanan dan robot akan melakukan putar balik.

5. Perulangan ketiga (“while not wall_in_front()”)

Perulangan tersebut digunakan agar robot terus bergerak maju selama tidak ada dinding di depan dengan memanggil fungsi “move()”.

6. Belok ke kiri terakhir (“turn_left()”)

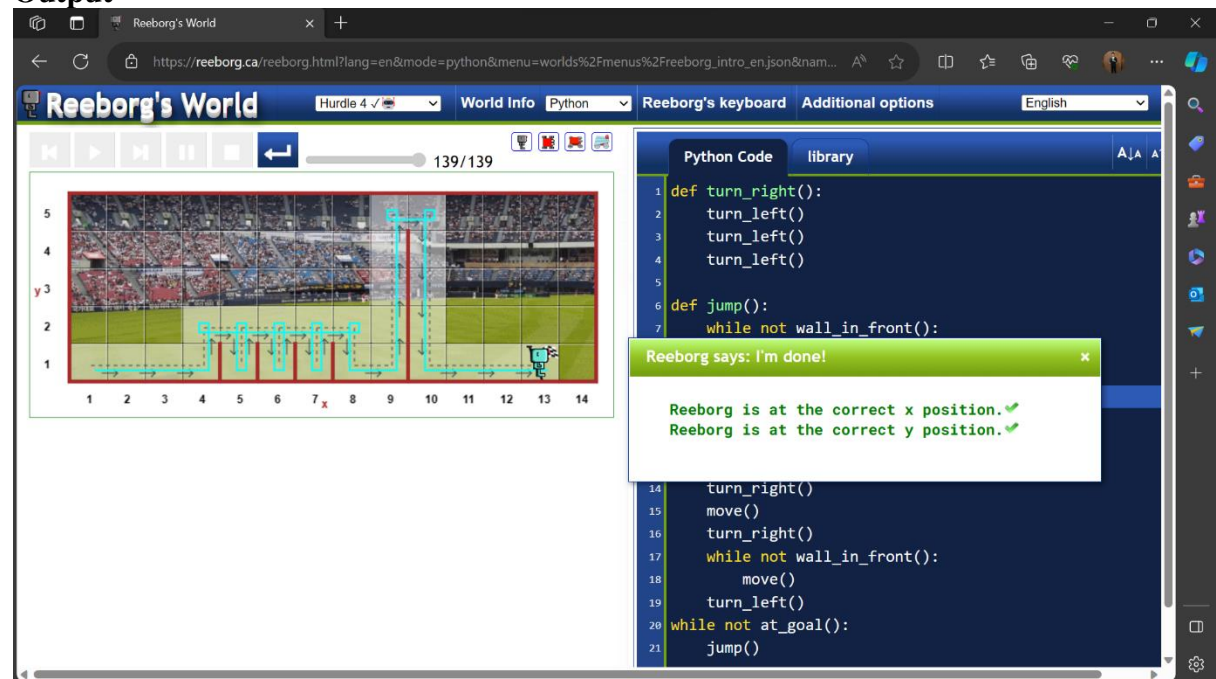
Setelah keluar dari perulangan pertama, panggil fungsi “turn_left()” satu kali. Fungsi tersebut digunakan agar robot melakukan belok ke kiri jika dinding tepat di depan, jika dinding belum tepat di depan maka akan kembali melakukan perulangan ketiga dengan memanggil fungsi “move()”.

3. Perulangan Utama:

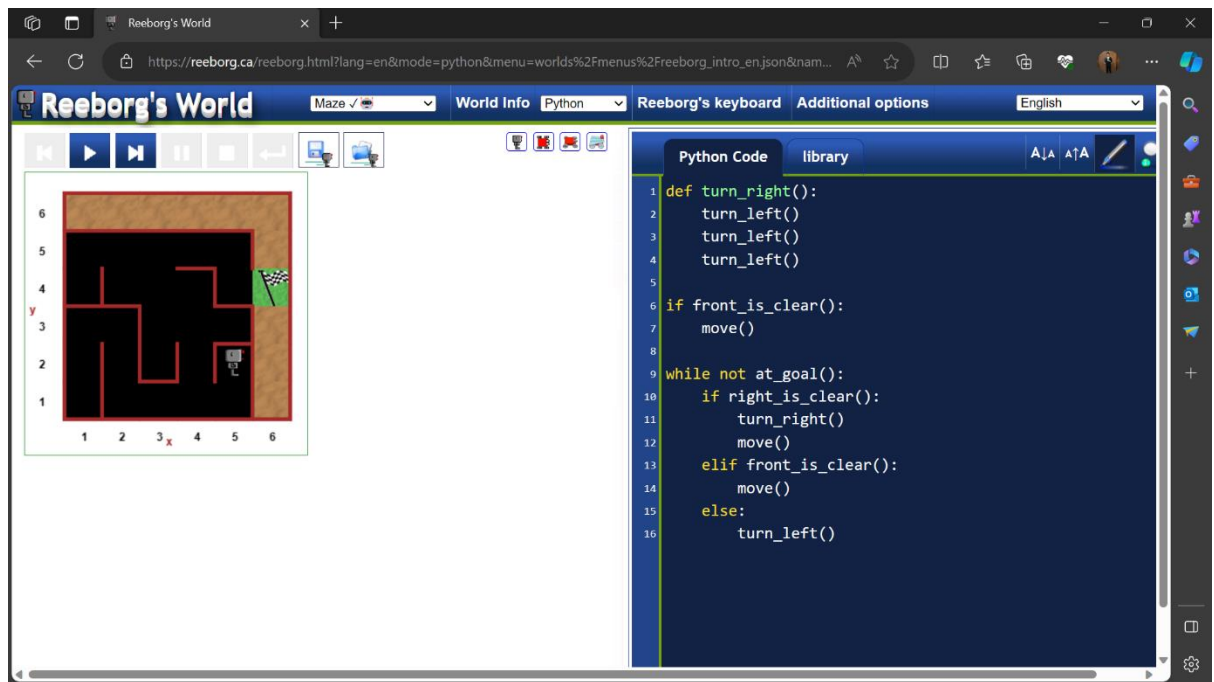
```
while not at_goal():  
    jump()
```

Ini adalah perulangan utama yang berjalan selama robot belum mencapai garis finish. Di dalamnya ada fungsi “jump()” yang di panggil, maka proses definisi fungsi “jump” yang di atas akan diulang.

Output



Maze



Source Code

```
Python Code library A↓A A↑A
1 def turn_right():
2     turn_left()
3     turn_left()
4     turn_left()
5
6 if front_is_clear():
7     move()
8
9 while not at_goal():
10     if right_is_clear():
11         turn_right()
12         move()
13     elif front_is_clear():
14         move()
15     else:
16         turn_left()
```

Penjelasan Step By Step

1. Definisi Fungsi “turn_right”:

```
def turn_right():  
    turn_left()  
    turn_left()  
    turn_left()
```

Fungsi ini berisi tiga pemanggilan fungsi “turn_left()”, yang menunjukkan bahwa “turn_right()” sebenarnya adalah operasi “turn_left()” yang dilakukan tiga kali. Ini digunakan agar robot dapat membuat pergerakan 90 derajat ke kanan (belok ke kanan).

2. Cek Depan Apakah Clear

```
if front_is_clear():  
    move()
```

Program pertama-tama akan memeriksa apakah bagian depan robot tidak terhalang. Jika tidak ada halangan (front_is_clear()), maka fungsi “move()” dipanggil untuk memindahkan robot satu langkah ke depan.

3. Loop Sampai Mencapai Tujuan (“at_goal()”):

```
while not at_goal():  
    if right_is_clear():  
        turn_right()  
        move()  
    elif front_is_clear():  
        move()  
    else:  
        turn_left()
```

Ini mengatur sebuah loop yang terus berjalan sampai robot mencapai tujuannya. Di dalam loop, terdapat tiga pernyataan kondisional:

1. Pengecekan Kondisi Kanan Clear (“right is clear”):

```
if right_is_clear():  
    turn_right()  
    move()
```

Di dalam loop, pertama-tama, program memeriksa apakah sebelah kanan robot tidak terhalang. Jika benar “right_is_clear()”, maka robot akan memanggil fungsi “turn_right()” untuk berputar ke kanan dan kemudian memanggil “move()” untuk maju satu langkah.

2. Pengecekan Kondisi Depan Clear (“front_is_clear()”):

```
elif front_is_clear():  
    move()
```

Jika sebelah kanan terhalang, program memeriksa apakah bagian depan robot tidak terhalang. Jika benar “front_is_clear()”, agen akan memanggil “move()” untuk maju satu langkah.

3. Jika Kondisi Tidak Terpenuhi:

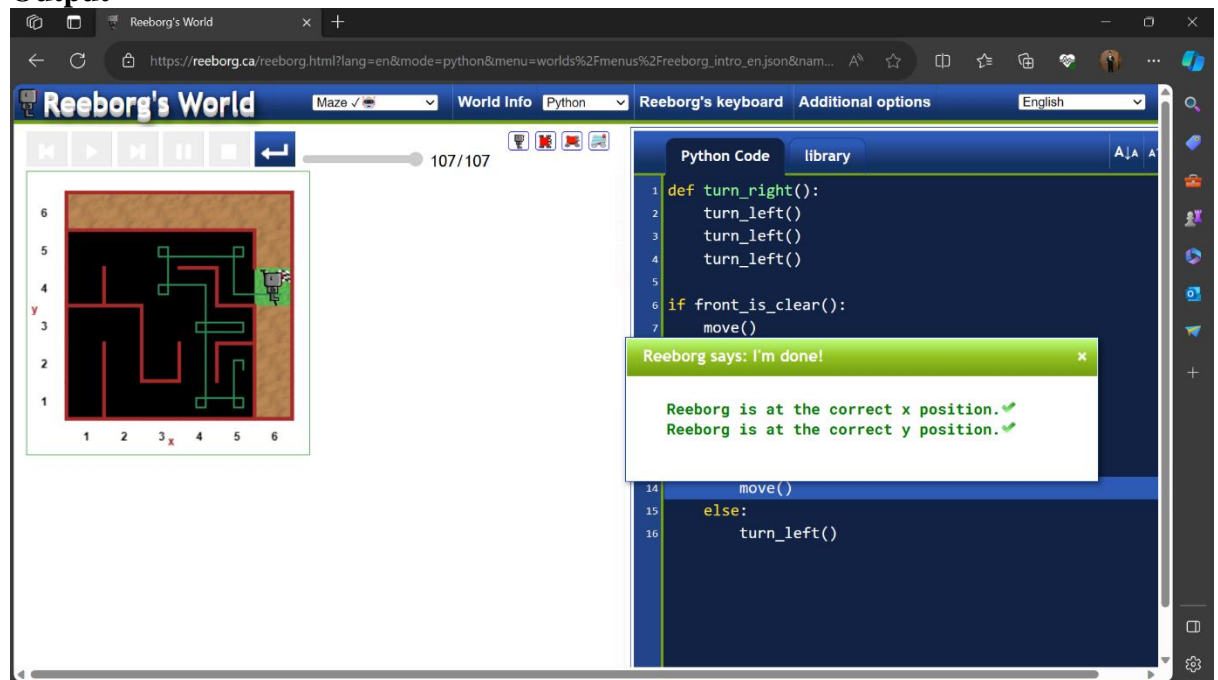
```
else:  
    turn_left()
```

Jika kondisi di atas tidak ada yang terpenuhi (sebelah kanan dan depan terhalang), maka robot akan memanggil fungsi “turn_left()” untuk belok ke kiri.

Note:

Jika robot mengecek semua jalan yang ada di labirin, itu merupakan hal yang wajar. Karena jalanan yang ada di labirin sangat gelap dan baterai di senternya habis. Maka robot akan mengecek semua jalan sesuai dengan code program di atas.

Output



Link GitHub:

<https://github.com/MadaPermata/robot-kelompok-3/tree/main>