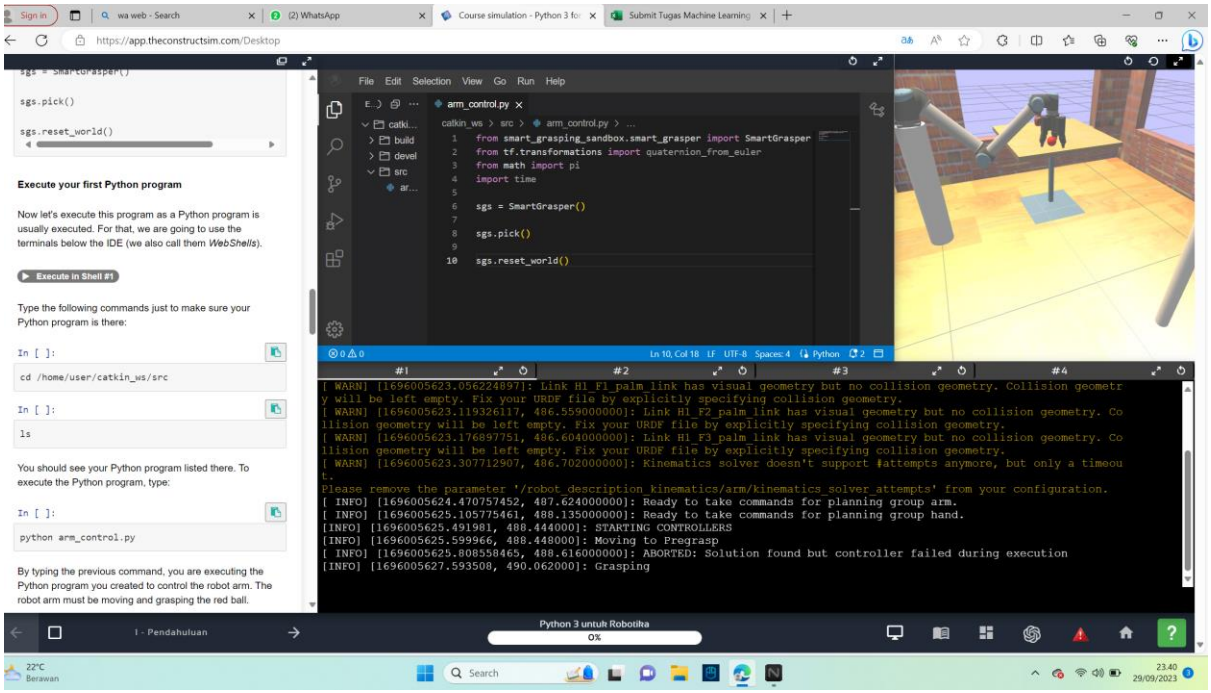
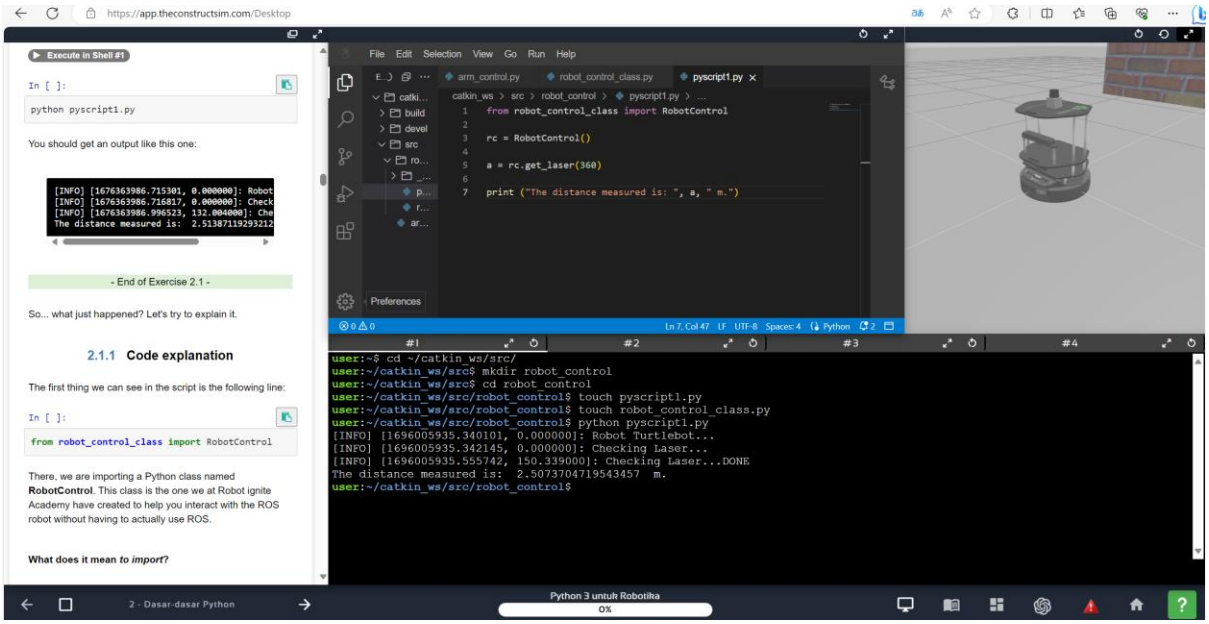
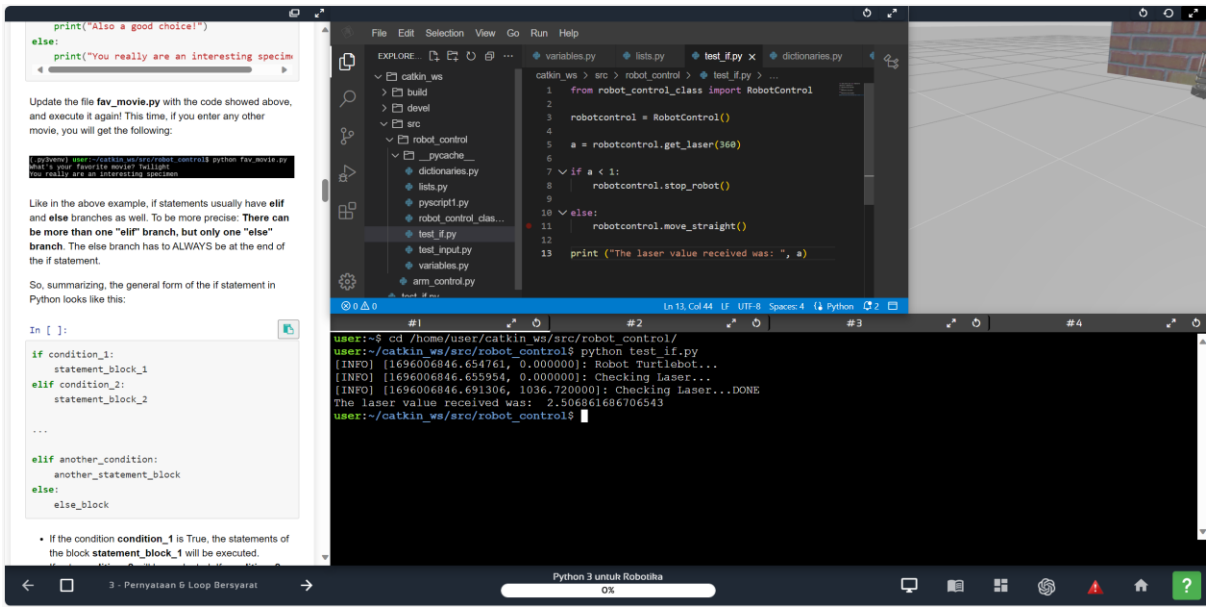
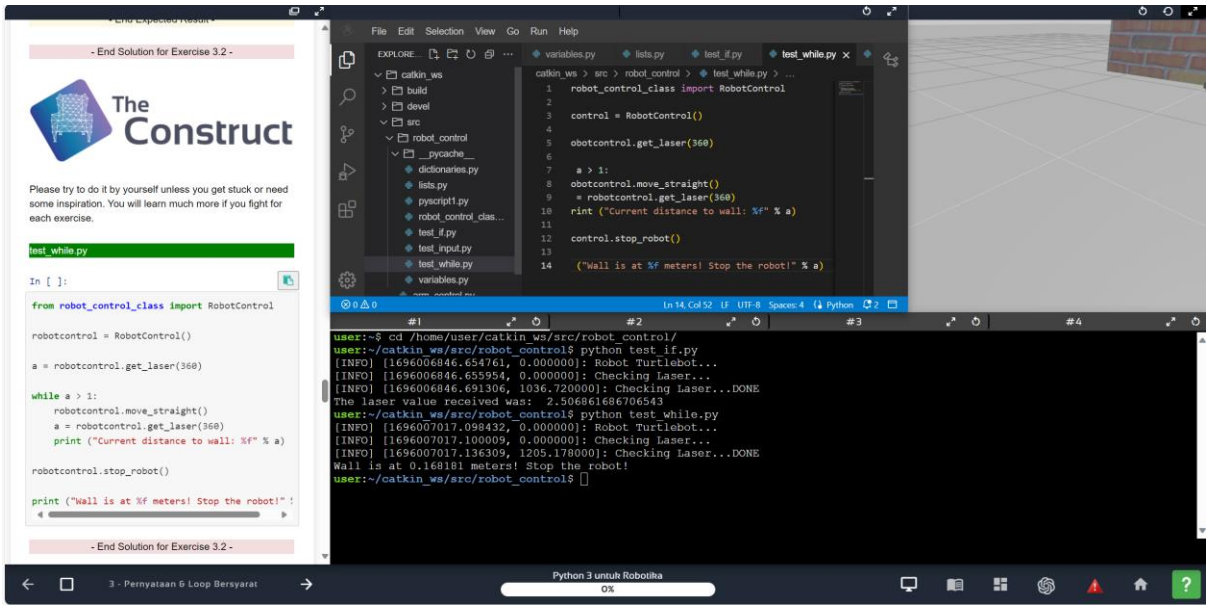
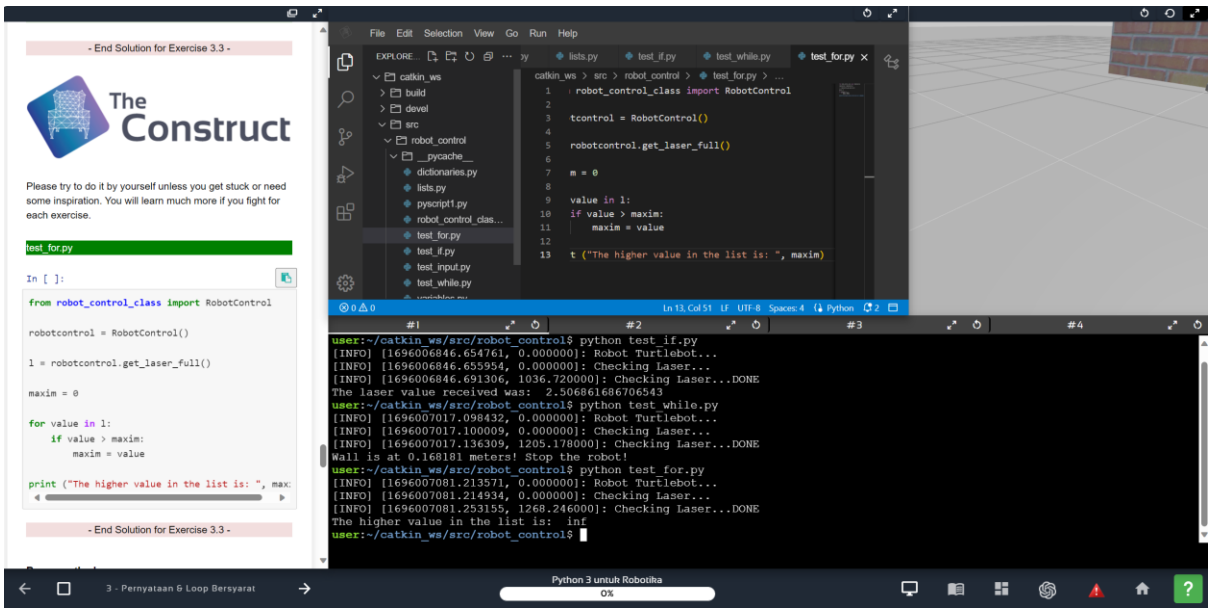
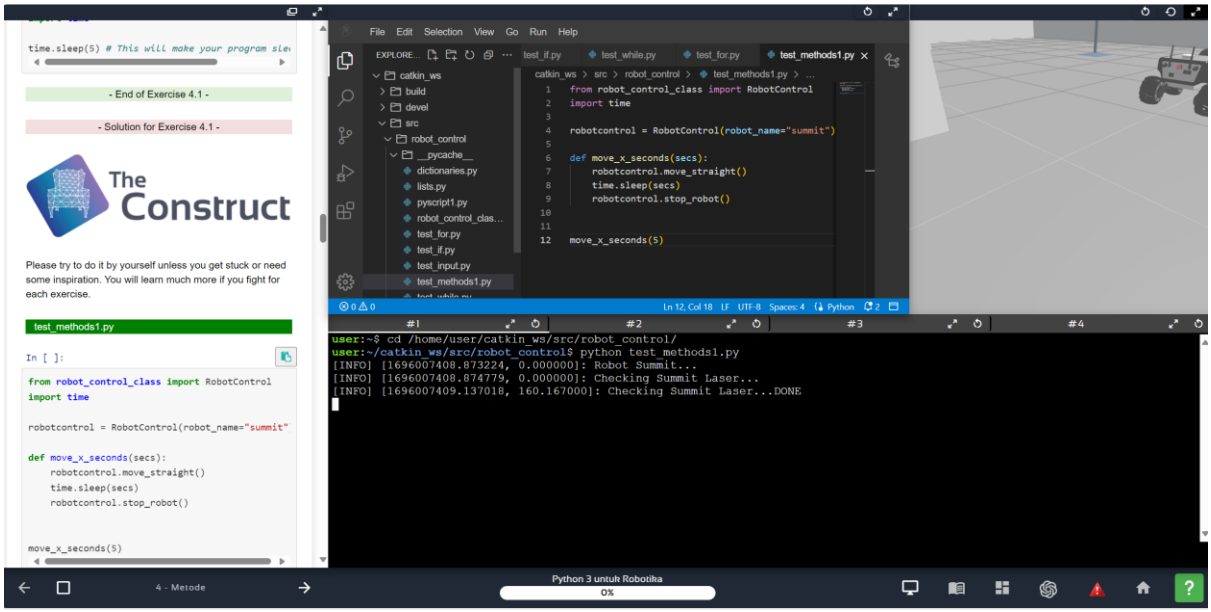

sgs.pick()
sgs.reset_world()
Execute your first Python program
Now let's execute this program as a Python program is usually executed. For that, we are going to use the terminals below the IDE (we also call them WebShells).
Execute in Shell #1
Type the following commands just to make sure your Python program is there:
In [ ]:
cd /home/user/catkin_ws/src
In [ ]:
ls
You should see your Python program listed there. To execute the Python program, type:
In [ ]:
python arm_control.py
By typing the previous command, you are executing the Python program you created to control the robot arm. The robot arm must be moving and grasping the red ball.
File Edit Selection View Go Run Help
arm_control.py
catkin_ws > src > arm_control.py
from smart_grasping_sandbox.smart_grasper import SmartGrasper
from tf.transformations import quaternion_from_euler
from math import pi
import time
sgs = SmartGrasper()
sgs.pick()
sgs.reset_world()
[WARN] [1696005623.056224897]: Link H1_F1_palm_link has visual geometry but no collision geometry. Collision geometry will be left empty. Fix your URDF file by explicitly specifying collision geometry.
[WARN] [1696005623.119326117, 486.559000000]: Link H1_F2_palm_link has visual geometry but no collision geometry. Collision geometry will be left empty. Fix your URDF file by explicitly specifying collision geometry.
[WARN] [1696005623.176897751, 486.604000000]: Link H1_F3_palm_link has visual geometry but no collision geometry. Collision geometry will be left empty. Fix your URDF file by explicitly specifying collision geometry.
[WARN] [1696005623.307712907, 486.702000000]: Kinematics solver doesn't support #attempts anymore, but only a timeout.
Please remove the parameter '/robot_description_kinematics/arm/kinematics_solver_attempts' from your configuration.
[INFO] [1696005624.470757452, 487.624000000]: Ready to take commands for planning group arm.
[INFO] [1696005625.105775461, 488.135000000]: Ready to take commands for planning group hand.
[INFO] [1696005625.491981, 488.444000]: STARTING CONTROLLERS
[INFO] [1696005625.599966, 488.448000]: Moving to Pregrasp
[INFO] [1696005625.808558465, 488.616000000]: ABORTED: Solution found but controller failed during execution
[INFO] [1696005627.593508, 490.062000]: Grasping
I - Pendahuluan
Python 3 untuk Robotika
0%
22°C
Berawan
Search
23.40
29/09/2023

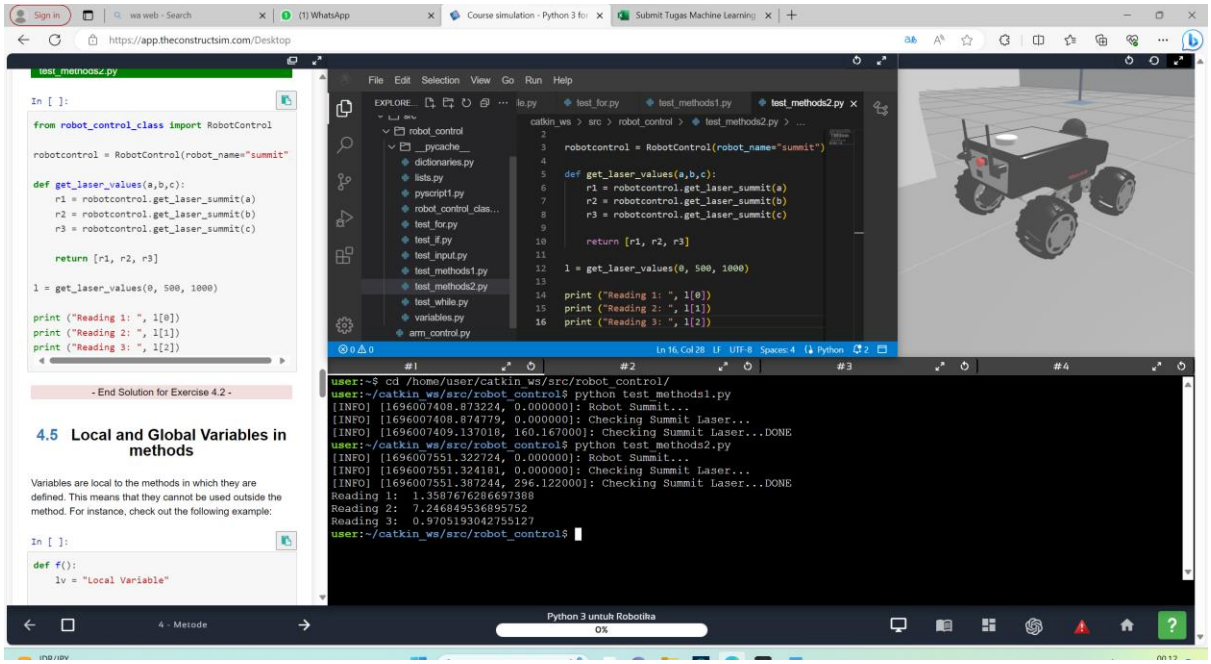

https://app.theconstructsim.com/Desktop
Execute in Shell #1
In [ ]:
python pyscript1.py
You should get an output like this one:
[INFO] [1676363986.715301, 0.000000]: Robot
[INFO] [1676363986.716817, 0.000000]: Check
[INFO] [1676363986.996523, 132.004000]: Che
The distance measured is: 2.513871192932125
- End of Exercise 2.1 -
So... what just happened? Let's try to explain it.
2.1.1 Code explanation
The first thing we can see in the script is the following line:
In [ ]:
from robot_control_class import RobotControl
There, we are importing a Python class named RobotControl. This class is the one we at Robot ignite Academy have created to help you interact with the ROS robot without having to actually use ROS.
What does it mean to import?
File Edit Selection View Go Run Help
arm_control.py
robot_control_class.py
pyscript1.py
catkin_ws > src > robot_control > pyscript1.py
from robot_control_class import RobotControl
rc = RobotControl()
a = rc.get_laser(360)
print ("The distance measured is: ", a, " m.")
Preferences
user:~$ cd ~/catkin_ws/src/
user:~/catkin_ws/src$ mkdir robot_control
user:~/catkin_ws/src$ cd robot_control
user:~/catkin_ws/src/robot_control$ touch pyscript1.py
user:~/catkin_ws/src/robot_control$ touch robot_control_class.py
user:~/catkin_ws/src/robot_control$ python pyscript1.py
[INFO] [1696005935.340101, 0.000000]: Robot Turtlebot...
[INFO] [1696005935.342145, 0.000000]: Checking Laser...
[INFO] [1696005935.555742, 150.339000]: Checking Laser...DONE
The distance measured is: 2.5073704719543457 m.
user:~/catkin_ws/src/robot_control$
2 - Dasar-dasar Python
Python 3 untuk Robotika
0%

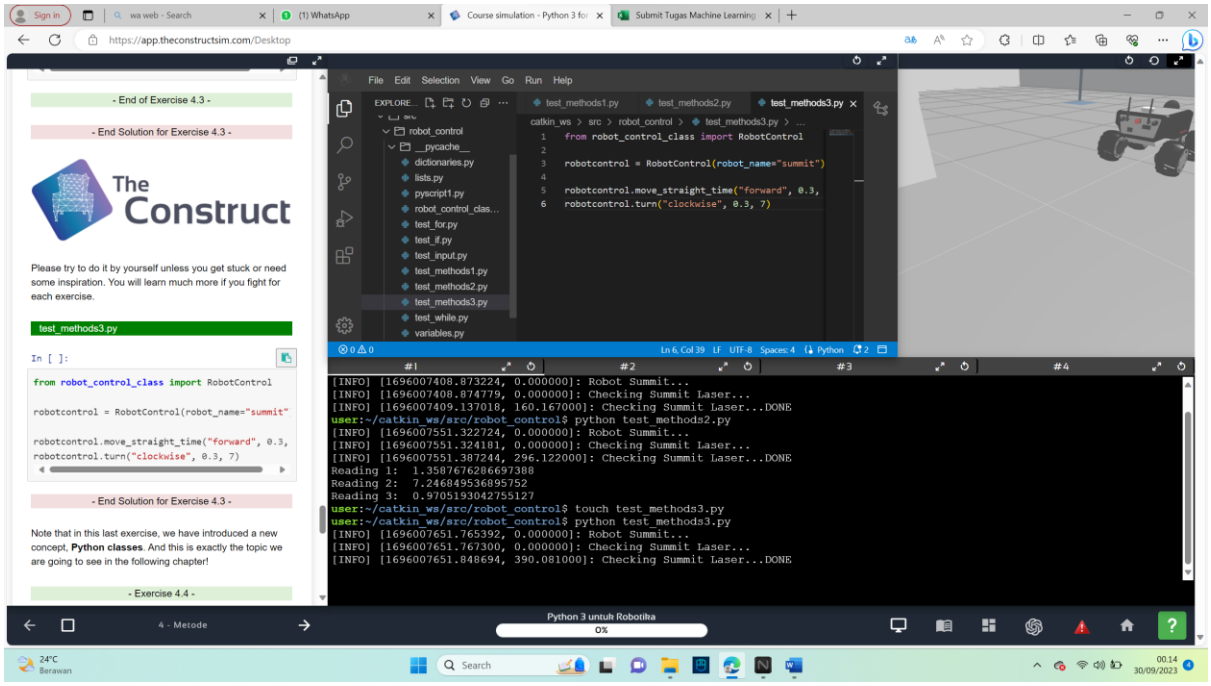

print("Also a good choice!")
else:
print("You really are an interesting specim
Update the file fav_movie.py with the code showed above, and execute it again! This time, if you enter any other movie, you will get the following:
(.py3venv) user:~/catkin_ws/src/robot_control$ python fav_movie.py
What's your favorite movie? Twilight
You really are an interesting specimen
Like in the above example, if statements usually have elif and else branches as well. To be more precise: There can be more than one "elif" branch, but only one "else" branch. The else branch has to ALWAYS be at the end of the if statement.
So, summarizing, the general form of the if statement in Python looks like this:
In [ ]:
if condition_1:
statement_block_1
elif condition_2:
statement_block_2
...
elif another_condition:
another_statement_block
else:
else_block
If the condition condition_1 is True, the statements of the block statement_block_1 will be executed.
File Edit Selection View Go Run Help
EXPLORE...
variables.py
lists.py
test_if.py
dictionaries.py
catkin_ws
build
devel
src
robot_control
__pycache__
dictionaries.py
lists.py
pyscript1.py
robot_control_clas...
test_if.py
test_input.py
variables.py
arm_control.py
catkin_ws > src > robot_control > test_if.py > ...
from robot_control_class import RobotControl
robotcontrol = RobotControl()
a = robotcontrol.get_laser(360)
if a < 1:
robotcontrol.stop_robot()
else:
robotcontrol.move_straight()
print ("The laser value received was: ", a)
Ln 13, Col 44 LF UTF-8 Spaces: 4 Python
#1
#2
#3
#4
user:~$ cd /home/user/catkin_ws/src/robot_control/
user:~/catkin_ws/src/robot_control$ python test_if.py
[INFO] [1696006846.654761, 0.000000]: Robot Turtlebot...
[INFO] [1696006846.655954, 0.000000]: Checking Laser...
[INFO] [1696006846.691306, 1036.720000]: Checking Laser...DONE
The laser value received was: 2.506861686706543
user:~/catkin_ws/src/robot_control$
3 - Pernyataan & Loop Bersyarat
Python 3 untuk Robotika
0%

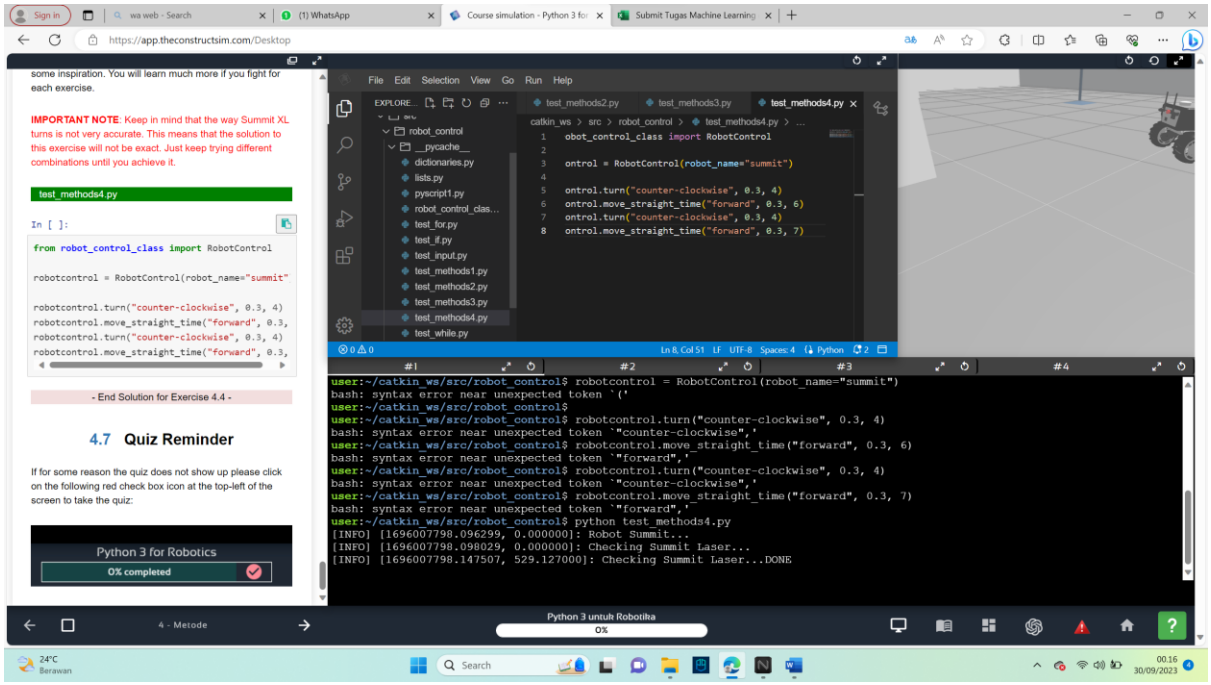

- End Solution for Exercise 3.2 -
The Construct
Please try to do it by yourself unless you get stuck or need some inspiration. You will learn much more if you fight for each exercise.
test_while.py
In [ ]:
from robot_control_class import RobotControl
robotcontrol = RobotControl()
a = robotcontrol.get_laser(360)
while a > 1:
robotcontrol.move_straight()
a = robotcontrol.get_laser(360)
print ("Current distance to wall: %f" % a)
robotcontrol.stop_robot()
print ("Wall is at %f meters! Stop the robot!" %
- End Solution for Exercise 3.2 -
File Edit Selection View Go Run Help
EXPLORE...
variables.py
lists.py
test_if.py
test_while.py
catkin_ws
build
devel
src
robot_control
__pycache__
dictionaries.py
lists.py
pyscript1.py
robot_control_clas...
test_if.py
test_input.py
test_while.py
variables.py
catkin_ws > src > robot_control > test_while.py > ...
robot_control_class import RobotControl
control = RobotControl()
obotcontrol.get_laser(360)
a > 1:
obotcontrol.move_straight()
= robotcontrol.get_laser(360)
rint ("Current distance to wall: %f" % a)
control.stop_robot()
("Wall is at %f meters! Stop the robot!" % a)
Ln 14, Col 52 LF UTF-8 Spaces: 4 Python
#1
#2
#3
#4
user:~$ cd /home/user/catkin_ws/src/robot_control/
user:~/catkin_ws/src/robot_control$ python test_if.py
[INFO] [1696006846.654761, 0.000000]: Robot Turtlebot...
[INFO] [1696006846.655954, 0.000000]: Checking Laser...
[INFO] [1696006846.691306, 1036.720000]: Checking Laser...DONE
The laser value received was: 2.506861686706543
user:~/catkin_ws/src/robot_control$ python test_while.py
[INFO] [1696007017.098432, 0.000000]: Robot Turtlebot...
[INFO] [1696007017.100009, 0.000000]: Checking Laser...
[INFO] [1696007017.136309, 1205.178000]: Checking Laser...DONE
Wall is at 0.168181 meters! Stop the robot!
user:~/catkin_ws/src/robot_control$
3 - Pernyataan & Loop Bersyarat
Python 3 untuk Robotika
0%


- End Solution for Exercise 3.3 -
The Construct
Please try to do it by yourself unless you get stuck or need some inspiration. You will learn much more if you fight for each exercise.
test_for.py
In [ ]:
from robot_control_class import RobotControl
robotcontrol = RobotControl()
l = robotcontrol.get_laser_full()
maxim = 0
for value in l:
if value > maxim:
maxim = value
print ("The higher value in the list is: ", max
- End Solution for Exercise 3.3 -
File Edit Selection View Go Run Help
EXPLORE...
lists.py
test_if.py
test_while.py
test_for.py
catkin_ws
build
devel
src
robot_control
__pycache__
dictionaries.py
lists.py
pyscript1.py
robot_control_clas...
test_for.py
test_if.py
test_input.py
test_while.py
catkin_ws > src > robot_control > test_for.py > ...
robot_control_class import RobotControl
tcontrol = RobotControl()
robotcontrol.get_laser_full()
m = 0
value in l:
if value > maxim:
maxim = value
t ("The higher value in the list is: ", maxim)
Ln 13, Col 51 LF UTF-8 Spaces: 4 Python
#1
#2
#3
#4
user:~/catkin_ws/src/robot_control$ python test_if.py
[INFO] [1696006846.654761, 0.000000]: Robot Turtlebot...
[INFO] [1696006846.655954, 0.000000]: Checking Laser...
[INFO] [1696006846.691306, 1036.720000]: Checking Laser...DONE
The laser value received was: 2.506861686706543
user:~/catkin_ws/src/robot_control$ python test_while.py
[INFO] [1696007017.098432, 0.000000]: Robot Turtlebot...
[INFO] [1696007017.100009, 0.000000]: Checking Laser...
[INFO] [1696007017.136309, 1205.178000]: Checking Laser...DONE
Wall is at 0.168181 meters! Stop the robot!
user:~/catkin_ws/src/robot_control$ python test_for.py
[INFO] [1696007081.213571, 0.000000]: Robot Turtlebot...
[INFO] [1696007081.214934, 0.000000]: Checking Laser...
[INFO] [1696007081.253155, 1268.246000]: Checking Laser...DONE
The higher value in the list is: inf
user:~/catkin_ws/src/robot_control$
3 - Pernyataan & Loop Bersyarat
Python 3 untuk Robotika
0%


time.sleep(5) # This will make your program sle
- End of Exercise 4.1 -
- Solution for Exercise 4.1 -
The Construct
Please try to do it by yourself unless you get stuck or need some inspiration. You will learn much more if you fight for each exercise.
test_methods1.py
In [ ]:
from robot_control_class import RobotControl
import time
robotcontrol = RobotControl(robot_name="summit"
def move_x_seconds(secs):
robotcontrol.move_straight()
time.sleep(secs)
robotcontrol.stop_robot()
move_x_seconds(5)
File Edit Selection View Go Run Help
EXPLORE...
catkin_ws
build
devel
src
robot_control
__pycache__
dictionaries.py
lists.py
pyscript1.py
robot_control_clas...
test_for.py
test_if.py
test_input.py
test_methods1.py
test_while.py
test_for.py
test_methods1.py
catkin_ws > src > robot_control > test_methods1.py > ...
from robot_control_class import RobotControl
import time
robotcontrol = RobotControl(robot_name="summit")
def move_x_seconds(secs):
robotcontrol.move_straight()
time.sleep(secs)
robotcontrol.stop_robot()
move_x_seconds(5)
Ln 12, Col 18 LF UTF-8 Spaces: 4 Python
#1
#2
#3
#4
user:~$ cd /home/user/catkin_ws/src/robot_control/
user:~/catkin_ws/src/robot_control$ python test_methods1.py
[INFO] [1696007408.873224, 0.000000]: Robot Summit...
[INFO] [1696007408.874779, 0.000000]: Checking Summit Laser...
[INFO] [1696007409.137018, 160.167000]: Checking Summit Laser...DONE
4 - Metode
Python 3 untuk Robotika
0%


Sign in
wa web - Search
(1) WhatsApp
Course simulation - Python 3 for
Submit Tugas Machine Learning
https://app.theconstructsim.com/Desktop
test_methods2.py
In [ ]:
from robot_control_class import RobotControl
robotcontrol = RobotControl(robot_name="summit"
def get_laser_values(a,b,c):
r1 = robotcontrol.get_laser_summit(a)
r2 = robotcontrol.get_laser_summit(b)
r3 = robotcontrol.get_laser_summit(c)
return [r1, r2, r3]
l = get_laser_values(0, 500, 1000)
print ("Reading 1: ", l[0])
print ("Reading 2: ", l[1])
print ("Reading 3: ", l[2])
- End Solution for Exercise 4.2 -
4.5 Local and Global Variables in methods
Variables are local to the methods in which they are defined. This means that they cannot be used outside the method. For instance, check out the following example:
In [ ]:
def f():
lv = "Local Variable"
File Edit Selection View Go Run Help
EXPLORE...
robot_control
__pycache__
dictionaries.py
lists.py
pyscript1.py
robot_control_clas...
test_for.py
test_if.py
test_input.py
test_methods1.py
test_methods2.py
test_while.py
variables.py
arm_control.py
test_for.py
test_methods1.py
test_methods2.py
catkin_ws > src > robot_control > test_methods2.py > ...
robotcontrol = RobotControl(robot_name="summit")
def get_laser_values(a,b,c):
r1 = robotcontrol.get_laser_summit(a)
r2 = robotcontrol.get_laser_summit(b)
r3 = robotcontrol.get_laser_summit(c)
return [r1, r2, r3]
l = get_laser_values(0, 500, 1000)
print ("Reading 1: ", l[0])
print ("Reading 2: ", l[1])
print ("Reading 3: ", l[2])
Ln 16, Col 28 LF UTF-8 Spaces: 4 Python
#1
#2
#3
#4
user:~$ cd /home/user/catkin_ws/src/robot_control/
user:~/catkin_ws/src/robot_control$ python test_methods1.py
[INFO] [1696007408.873224, 0.000000]: Robot Summit...
[INFO] [1696007408.874779, 0.000000]: Checking Summit Laser...
[INFO] [1696007409.137018, 160.167000]: Checking Summit Laser...DONE
user:~/catkin_ws/src/robot_control$ python test_methods2.py
[INFO] [1696007551.322724, 0.000000]: Robot Summit...
[INFO] [1696007551.324181, 0.000000]: Checking Summit Laser...
[INFO] [1696007551.387244, 296.122000]: Checking Summit Laser...DONE
Reading 1: 1.3587676286697388
Reading 2: 7.246849536895752
Reading 3: 0.9705193042755127
user:~/catkin_ws/src/robot_control$
4 - Metode
Python 3 untuk Robotika
0%


- End of Exercise 4.3 -
- End Solution for Exercise 4.3 -
The Construct
Please try to do it by yourself unless you get stuck or need some inspiration. You will learn much more if you fight for each exercise.
test_methods3.py
In [ ]:
from robot_control_class import RobotControl
robotcontrol = RobotControl(robot_name="summit"
robotcontrol.move_straight_time("forward", 0.3,
robotcontrol.turn("clockwise", 0.3, 7)
- End Solution for Exercise 4.3 -
Note that in this last exercise, we have introduced a new concept, Python classes. And this is exactly the topic we are going to see in the following chapter!
- Exercise 4.4 -
File Edit Selection View Go Run Help
test_methods1.py
test_methods2.py
test_methods3.py
from robot_control_class import RobotControl
robotcontrol = RobotControl(robot_name="summit")
robotcontrol.move_straight_time("forward", 0.3,
robotcontrol.turn("clockwise", 0.3, 7)
[INFO] [1696007408.873224, 0.000000]: Robot Summit...
[INFO] [1696007408.874779, 0.000000]: Checking Summit Laser...
[INFO] [1696007409.137018, 160.167000]: Checking Summit Laser...DONE
user:~/catkin_ws/src/robot_control$ python test_methods2.py
[INFO] [1696007551.322724, 0.000000]: Robot Summit...
[INFO] [1696007551.324181, 0.000000]: Checking Summit Laser...
[INFO] [1696007551.387244, 296.122000]: Checking Summit Laser...DONE
Reading 1: 1.3587676286697388
Reading 2: 7.246849536895752
Reading 3: 0.9705193042755127
user:~/catkin_ws/src/robot_control$ touch test_methods3.py
user:~/catkin_ws/src/robot_control$ python test_methods3.py
[INFO] [1696007651.765392, 0.000000]: Robot Summit...
[INFO] [1696007651.767300, 0.000000]: Checking Summit Laser...
[INFO] [1696007651.848694, 390.081000]: Checking Summit Laser...DONE
4 - Metode
Python 3 untuk Robotika
0%


some inspiration. You will learn much more if you fight for each exercise.
IMPORTANT NOTE: Keep in mind that the way Summit XL turns is not very accurate. This means that the solution to this exercise will not be exact. Just keep trying different combinations until you achieve it.
test_methods4.py
In [ ]:
from robot_control_class import RobotControl
robotcontrol = RobotControl(robot_name="summit"
robotcontrol.turn("counter-clockwise", 0.3, 4)
robotcontrol.move_straight_time("forward", 0.3,
robotcontrol.turn("counter-clockwise", 0.3, 4)
robotcontrol.move_straight_time("forward", 0.3,
- End Solution for Exercise 4.4 -
4.7 Quiz Reminder
If for some reason the quiz does not show up please click on the following red check box icon at the top-left of the screen to take the quiz:
Python 3 for Robotics
0% completed
File Edit Selection View Go Run Help
test_methods2.py
test_methods3.py
test_methods4.py
obot_control_class import RobotControl
ontrol = RobotControl(robot_name="summit")
ontrol.turn("counter-clockwise", 0.3, 4)
ontrol.move_straight_time("forward", 0.3, 6)
ontrol.turn("counter-clockwise", 0.3, 4)
ontrol.move_straight_time("forward", 0.3, 7)
user:~/catkin_ws/src/robot_control$ robotcontrol = RobotControl(robot_name="summit")
bash: syntax error near unexpected token `('
user:~/catkin_ws/src/robot_control$
user:~/catkin_ws/src/robot_control$ robotcontrol.turn("counter-clockwise", 0.3, 4)
bash: syntax error near unexpected token `"counter-clockwise",'
user:~/catkin_ws/src/robot_control$ robotcontrol.move_straight_time("forward", 0.3, 6)
bash: syntax error near unexpected token `"forward",'
user:~/catkin_ws/src/robot_control$ robotcontrol.turn("counter-clockwise", 0.3, 4)
bash: syntax error near unexpected token `"counter-clockwise",'
user:~/catkin_ws/src/robot_control$ robotcontrol.move_straight_time("forward", 0.3, 7)
bash: syntax error near unexpected token `"forward",'
user:~/catkin_ws/src/robot_control$ python test_methods4.py
[INFO] [1696007798.096299, 0.000000]: Robot Summit...
[INFO] [1696007798.098029, 0.000000]: Checking Summit Laser...
[INFO] [1696007798.147507, 529.127000]: Checking Summit Laser...DONE
4 - Metode
Python 3 untuk Robotika
0%

programs we provide you that has not been explained in this course, go to the internet and find out what it means. You will be able to understand it very quickly now that you have a basic knowledge of the language.

ROS Basics in 5 Days

FINAL RECOMMENDATION: If you go to the ROS Basics course now, you will see that you will have to do a lot of exercises writing Python code for ROS. You have two options there: write the code using a procedural method, or using Python classes. Our recommendation is that you always try to do the exercises using classes, since it is the method of professional ROS programmers.

Keep pushing your Python learning!

5.6 Quiz Reminder

If for some reason the quiz does not show up please click on the following red check box icon at the top-left of the screen to take the quiz:

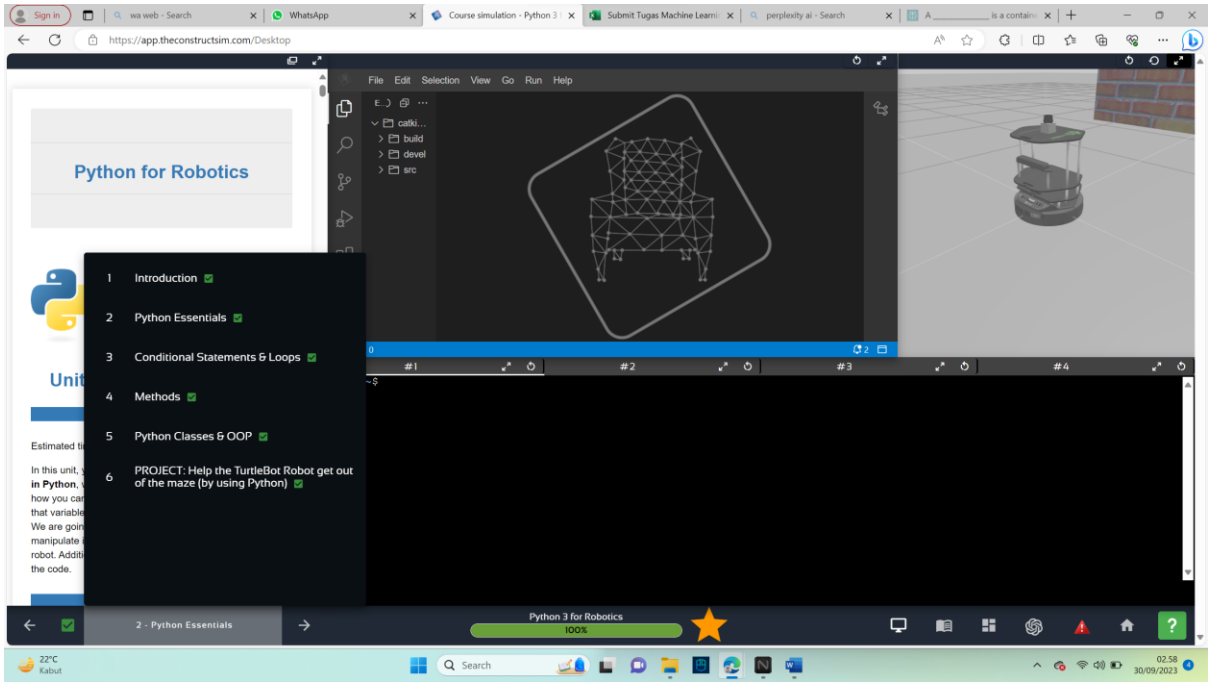

```
bash: syntax error near unexpected token `"counter-clockwise",'
user:~/catkin_ws/src/robot_control$ robotcontrol.move_straight_time("forward", 0.3, 7)
bash: syntax error near unexpected token `"forward",'
user:~/catkin_ws/src/robot_control$ python test_methods4.py
[INFO] [1696007798.096299, 0.000000]: Robot Summit...
[INFO] [1696007798.098029, 0.000000]: Checking Summit Laser...
[INFO] [1696007798.147507, 529.127000]: Checking Summit Laser...DONE
user:~/catkin_ws/src/robot_control$ touch test_class.py
user:~/catkin_ws/src/robot_control$ touch robot_control_class.py
user:~/catkin_ws/src/robot_control$ python est_class.py
python: can't open file 'est_class.py': [Errno 2] No such file or directory
user:~/catkin_ws/src/robot_control$ python test_class.py
[INFO] [1696008193.623111, 0.000000]: Robot Summit...
[INFO] [1696008193.624549, 0.000000]: Checking Summit Laser...
[INFO] [1696008193.682510, 906.970000]: Checking Summit Laser...DONE
```

direction of the laser reading.

- **get_laser_full()**: As the name itself says, this method will allow you to get all the data from ALL the laser beams of the robot. As I've said before, this is a total of 719 different readings. So, when called, this method will return a **LIST** containing all the 719 different readings from the laser beams.
- **move_straight()**: As the name says, this method will allow you to start moving the robot in a straight line.
- **stop_robot()**: As the name says, this method will allow you to stop the robot from moving.
- **move_straight_time(motion, speed, time)**: As the name itself says, this method will allow you to move the robot in a straight line. You will need to pass three parameters to it.
  - **motion**: Specify here if you want your robot to move forward (**"forward"**) or backward (**"backward"**).
  - **speed**: Specify here the speed at which you want your robot to move (in m/s).
  - **time**: Specify here how long you want your robot to keep moving (in seconds).
- **turn (clockwise, speed, time)**: As the name itself says, this method will allow you to turn the robot. You will need to pass three parameters to it.
  - **clockwise**: Specify here whether you want your robot to turn clockwise (**"clockwise"**) or counter-clockwise (**"counter-clockwise"**).
  - **speed**: Specify here the speed at which you want your robot to turn (in rad/s).
  - **time**: Specify here how long you want your robot to keep turning (in seconds).

Finally, and for this project, you are going to be able to use

```
user:~$ cd home/user/catkin_ws/src/robot_control
bash: cd: home/user/catkin_ws/src/robot_control: No such file or directory
user:~$ cd /home/user/catkin_ws/src/robot_control
user:~/catkin_ws/src/robot_control$ python tobot_control_class.py
python: can't open file 'tobot_control_class.py': [Errno 2] No such file or directory
user:~/catkin_ws/src/robot_control$ python robot_control_class.py
[INFO] [1696008435.683998, 0.000000]: Robot Turtlebot...
[INFO] [1696008435.685581, 0.000000]: Checking Laser...
[INFO] [1696008435.941941, 241.436000]: Checking Laser...DONE
user:~/catkin_ws/src/robot_control$
```


Python for Robotics
1 Introduction
2 Python Essentials
3 Conditional Statements & Loops
4 Methods
5 Python Classes & OOP
6 PROJECT: Help the TurtleBot Robot get out of the maze (by using Python)
2 - Python Essentials
Python 3 for Robotics
100%