# Ast x Oru Robotics Summer School

Andreas Persson, David Caceres Dominguez Pedro Zuidberg Dos Martires

Örebro University

github.com/pedrozudo/astxoru-roboticssummerschool

# Robot = Forced Labour (in Czech)

# Robot = Forced Labour (in Czech)

https://www.youtube.com/watch?v=GmU7SimFkpU

# Robot = Forced Labour (in Czech)

# Challenges in Robotics

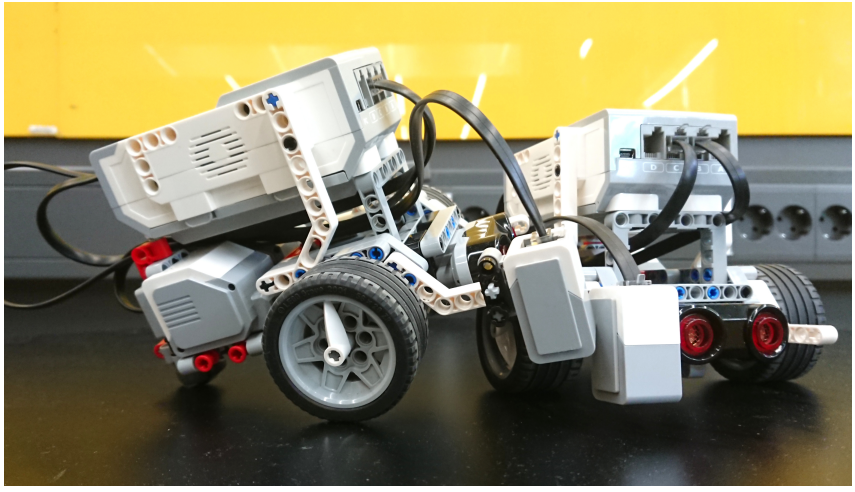
- building robots
- simulating robots
- robot control
- sensing
- inter-robot collaboration
- robot automation
- human-robot interaction
- ...

# Challenges in Robotics

- building robots
- simulating robots
- robot control
- sensing
- inter-robot collaboration
- robot automation
- human-robot interaction
- ...

**Even bigger problem:** solve all at the same time

# Our Robots



# Coding in Python

# Built-In Types

```
# This is a comment
n1 = 1 # an integer
n2 = 3 # an other integer

f1 = 0.5 # a float
f2 = 5. # an other float

b1, b2 = true, false # Booleans

s1 = "I am a string"
```

# Collections

```
l = [1, 2, 10, "some-string"] #list

t = (2, 4, 6.3) #tuple

d = {"key1" : 5, 2 : "string value"} #dictionary/hashmap
```

# Collections

```
l = [1, 2, 10, "some-string"] #list

t = (2, 4, 6.3) #tuple

d = {"key1" : 5, 2 : "string value"} #dictionary/hashmap

l[0] #gives 1
l[1] #gives 2
l[-2] #gives 10
l[3] = 5 # mutates the list to [1,2,10,5]
```

similar for tuples

# Collections

```
d = {"key1" : 5, 2 : "string value"} #dictionary/hashmap

d["key1"] # gives 5
d[2] # gives "string value"

d[2] = 10 # mutates dictionary into {"key1" : 5, 2 : 10}
```

# For Loops

```
l = [1, 2, 10, "some-string"]
for i in l:
    print(i)
```

This will return the following:

```
1
2
10
"some-string"
```

# For Loops

```
s = "some-string"
for i in s:
    print(i)
```

# For Loops

```
s = "some-string"
for i in s:
    print(i)
```

This will return the following:

```
s
o
m
e
-
...
```

# Conditional Statements

```
b1 = True
b2 = False
if b1 and b2:
    print("okay")
else if not (b1 and b2) :
    print("not okay")
else:
    print("somewhat okay")
```

# Functions

```
def is_pair(x):
    if x % 2:
        return True
    else:
        return False
```

# Classes

```
class Rectangle:
    def __init__(self, length, width):
        self.length = length
        self.width = width

    def area(self):
        return self.length * self.width

    def perimeter(self):
        return 2 * self.length + 2 * self.width

class Square(Rectangle):
    def __init__(self, length):
        super().__init__(length, length)
```

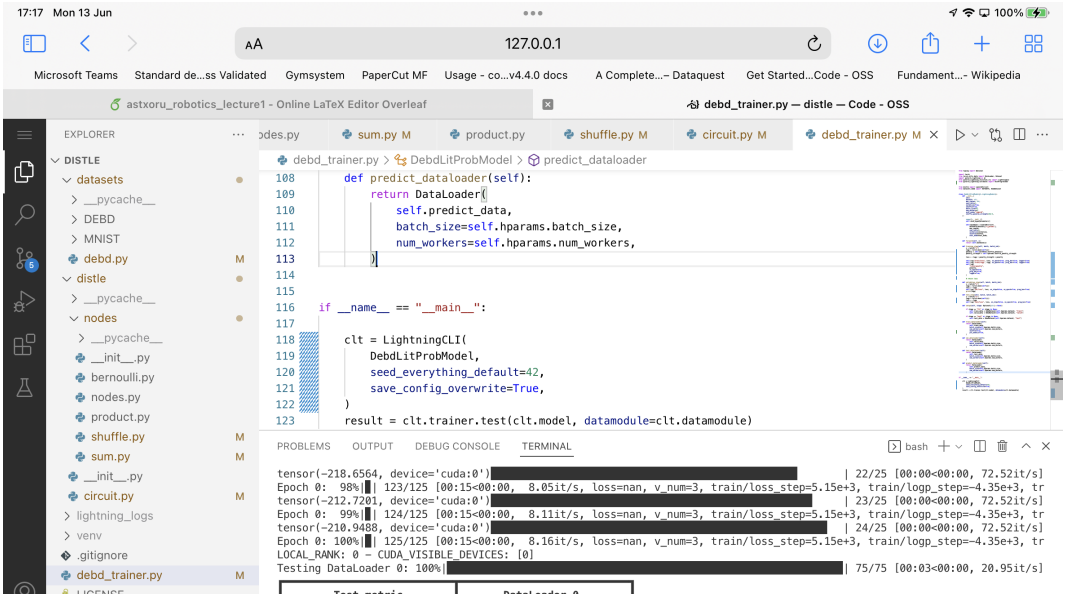
# Classes

```
r = Rectanglge(3,4)
s = Square(4)

print(r.area())
print(s.perimeter())
print(r.length)
```

# The VS Code IDE

# code0server: VS Code in the Browser



# The Command Line

# Programming Robots

## Robot Operating System

The Robot Operating System (ROS) is a flexible framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.

ros.org (general information)
wiki.ros.org (practical information)

# Robot Operating System

- A standard (more or less) for writing robot software.
- Encourages modular solutions.
- Targets different levels of abstraction:

**Low:** Communications infrastructure
**Mid:** Robot-Specific components (e.g. specific sensors)
**High:** Tooling (graphical interfaces, command-line tools, . . . )

## Robot Specific Components

- Standard robot messages,
- Robot geometry library,
- Robot description language,
- Diagnostic tools,
- Pose estimation, localization, and Navigation,

# Robot Specific Components

- Standard robot messages,
- Robot geometry library,
- Robot description language,
- Diagnostic tools,
- Pose estimation, localization, and Navigation,
- ...support for, e.g.:

# ROS Nodes

Nodes are program that communicate with each other.

- Message Passing (through "topics")
- Publisher (talker)
- Subscriber (listener)
- Uses ROS 'msg'-files