

Egyptian Russian University		Faculty of Engineering
Mechatronics & Robotics Dept.		Course Title: Computer Programing and Introduction to AI
Max time: 60 min	Mid Term Exam. – V2 Fall 2025-2026	Course Code: BSN 106 Dr. Omar Abdelaziz

Question 1 (5 marks)

- Write a Python program that takes an even number "n" of inputs from the user.
 - Add the first half of the inputs to a list called `desired_path`.
 - Add the second half of the inputs to a list called `measured_path`.
 - Make code to calculate the Mean Squared Error (MSE) between the two lists.
- Formula:** $MSE = \frac{1}{(n/2)} \sum_{i=0}^{(n/2)-1} (desired_path[i] - measured_path[i])^2$
- Print both lists and the final MSE value.

Test Example: If n = 6

`desired_path = [1.0, 2.0, 3.0],`

`measured_path = [1.1, 1.9, 3.0]`

```
Enter value: 1.0
Enter value: 2.0
Enter value: 3.0
Enter value: 1.1
Enter value: 1.9
Enter value: 3.0
```

- Results → $MSE = ((1.0 - 1.1)^2 + (2.0 - 1.9)^2 + (3.0 - 3.0)^2) / 3 = 0.00666$

Question 2 (3 marks)

- Make a list named `voltages` that contains elements from -5.0 to +5.0 (inclusive) with a step of 0.1. Print the list.

results → : [-5.0, -4.9, -4.8, ..., 4.9, 5.0]

Question 3 (5 marks)

In mechatronics, actuators have limits. Make a function `saturate_actuator` that takes a list of `motor_commands`, a `min_val`, and a `max_val`.

- The function must return a new list where:
 - Any value below `min_val` is set to `min_val`.
 - Any value above `max_val` is set to `max_val`.
 - Values within the range are unchanged.
 - Print the new saturated list.

Test Example:

- `commands = [-20, 300, 120, 256, 0]` , `min_val = 0` , `max_val = 255`
- Result → : [0, 255, 120, 255, 0]

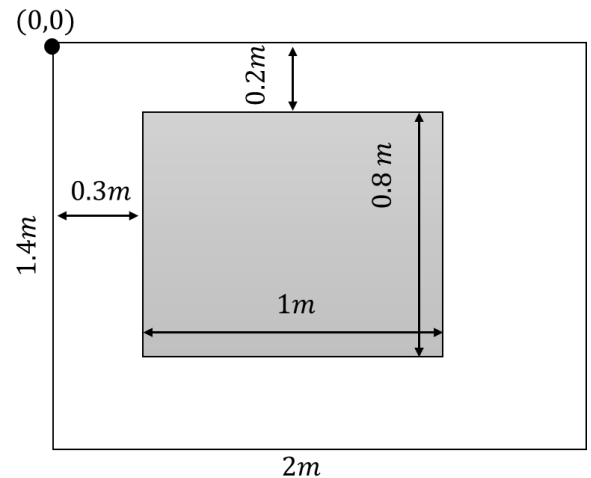
Question 4 (7 marks)

- Make a function to build an occupancy map that takes inputs
 - map width and map height
 - list for the square object has four items as follows: `obj_list=[h0,h,w0,w]`
 1. object initial height (`h0`)
 2. object height (`h`)
 3. object initial width (`w0`)
 4. object width (`w`)
 - map resolution/pixel accuracy (`acur`)
- the function should print the following
 - Number of rows/ grid Height
 - Number of columns / grid width
 - occupancy grid/map `occ_grid` (a list of lists)

Test Example:

shown map is (1.4 m × 2 m, 10 cm resolution) showing the objects in gray (ones) and free areas in white (zeros).

- Knowing Pixel accuracy (resolution): 0.1 m (10 cm):
- object initial height(`h0`)=0.2m ,
- object height (`h`)=0.8m,
- object initial width (`w0`)=0.3m,
- object width(`w`)=1m



(notice you need to check the results from any float division for example sometimes $1.9/0.1$ give you 18.9999 use `math.ceil(1.9/0.1)` gives 19, you need to (import math) in the beginning of the code

results

- Number of rows = 14,
- Number of columns =20

```
1 for row in occ_grid:
2     print(row)
```

```
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
```