

# SENSOR NETWORK

V Pavani:21b01a12i4:IT

B Reshma:21b01a1216:IT

P Sri Vyshnavi:21b01a5484:AIDS

L Parimala Prasanna:22b05a5408:AIDS

S Kalyani:21b01a0251:EEE

SVECW

March 25,2023

# Introduction about the project

- ▶ A subset of sensors where each pair of sensors can communicate directly with each other (distance at most  $d$ ). The goal is to choose as many sensors as possible, given their current locations represented as points in a two-dimensional plane, using Euclidean distance to measure the distance between two points.

# Approach

- ▶ We used command line arguments to provide input
- ▶ math and sys modules are imported
- ▶ User defined functions are used
- ▶ We used Euclidean distance to measure the distance between two sensors

# Learnings

We have learnt about how to pass command line arguments in VS code and ability to work collaboratively and productively with others. How to manage our time effectively and meet deadlines. We have learnt the usage of GitLab and LaTeX.

# Challenges

- ▶ Integration of code is difficult.
- ▶ Indentation errors,data type errors.
- ▶ We faced difficulty while using command line arguments in visual studio code

# STATISTICS

- ▶ The Python code has overall 49 lines.
- ▶ The code has three user-defined functions.

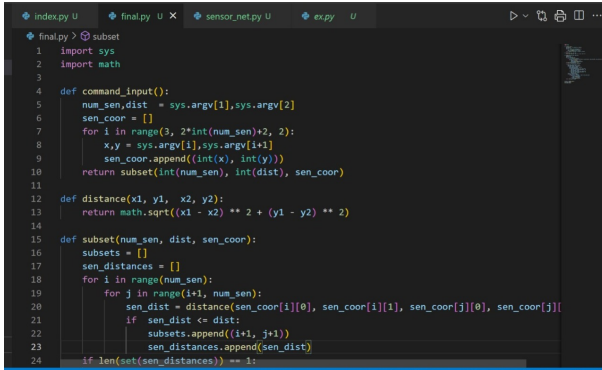
They are:

- 1.command-input()
- 2.distance()
- 3.subset()

- ▶ Built-in Modules:

- 1.sys
- 2.math

# Demo/Screenshots



The screenshot shows a code editor with four tabs: 'index.py U', 'final.py U X', 'sensor\_net.py U', and 'ex.py U'. The 'final.py' tab is active, displaying Python code. The code defines a 'command\_input()' function that takes command-line arguments for the number of sensors and distance, and a 'distance()' function that calculates the Euclidean distance between two points. A 'subset()' function is also defined, which iterates through the sensors to find pairs that are within a specified distance. The code is as follows:

```
1 import sys
2 import math
3
4 def command_input():
5     num_sen,dist = sys.argv[1],sys.argv[2]
6     sen_coor = []
7     for i in range(3, 2*int(num_sen)+2, 2):
8         x,y = sys.argv[i],sys.argv[i+1]
9         sen_coor.append((int(x), int(y)))
10    return subset(int(num_sen), int(dist), sen_coor)
11
12 def distance(x1, y1, x2, y2):
13     return math.sqrt((x1 - x2) ** 2 + (y1 - y2) ** 2)
14
15 def subset(num_sen, dist, sen_coor):
16     subsets = []
17     sen_distances = []
18     for i in range(num_sen):
19         for j in range(i+1, num_sen):
20             sen_dist = distance(sen_coor[i][0], sen_coor[i][1], sen_coor[j][0], sen_coor[j][1])
21             if sen_dist <= dist:
22                 subsets.append((i+1, j+1))
23                 sen_distances.append(sen_dist)
24     if len(set(sen_distances)) == 1:
```

Figure 1: Code of the project

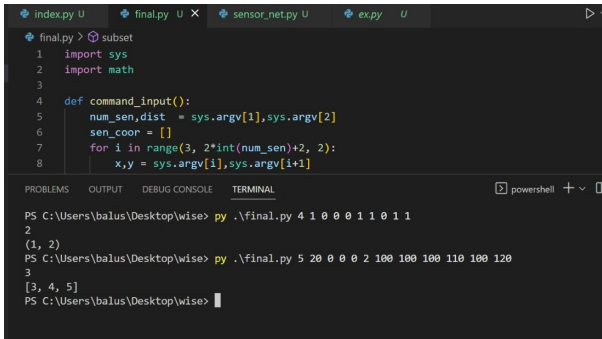
# Demo/Screenshots

```
final.py > ...
24     if len(set(sen_distances)) == 1:
25         return len(subsets[0]),subsets[0]
26     else :
27         max_conn, max_conn_sens = 0, 0
28         sens_connections = [0] * num_sen
29         for sens1, sens2 in subsets:
30             sens_connections[sens1 - 1] += 1
31             sens_connections[sens2 - 1] += 1
32             if sens_connections[sens1 - 1] > max_conn:
33                 max_conn = sens_connections[sens1 - 1]
34                 max_conn_sens = sens1
35             if sens_connections[sens2 - 1] > max_conn:
36                 max_conn = sens_connections[sens2 - 1]
37                 max_conn_sens = sens2
38         result = [max_conn_sens]
39         for sens1, sens2 in subsets:
40             if sens1 == max_conn_sens or sens2 == max_conn_sens:
41                 if sens1 not in result:
42                     result.append(sens1)
43                 if sens2 not in result:
44                     result.append(sens2)
45         return len(result),result
46
47 max_sub = command_input()
48 print(max_sub[0])
49 print(max_sub[1])
```

Figure 2: Code of the project



# Demo/Screenshots



The screenshot shows a code editor with four tabs: `index.py U`, `final.py U X`, `sensor_net.py U`, and `ex.py U`. The `final.py` tab is active, displaying a Python script. Below the editor is a terminal window with the `TERMINAL` tab selected, showing the execution of the script from a PowerShell prompt.

```
final.py > subset
1  import sys
2  import math
3
4  def command_input():
5      num_sen,dist = sys.argv[1],sys.argv[2]
6      sen_coor = []
7      for i in range(3, 2*int(num_sen)+2, 2):
8          x,y = sys.argv[i],sys.argv[i+1]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL powershell + -

```
PS C:\Users\balus\Desktop\wise> py .\final.py 4 1 0 0 0 1 1 0 1 1
2
(1, 2)
PS C:\Users\balus\Desktop\wise> py .\final.py 5 20 0 0 0 2 100 100 100 110 100 120
3
[3, 4, 5]
PS C:\Users\balus\Desktop\wise> 
```

Figure 3: demo output

Thank You