

DSA3 Lab session 1

Administrivia

- Lab time Fridays 14:00-18:00
- Lab focus will not be helping with assignments, but you can come and ask for help
- We'll use 1-2 hours of lab time to review the week's topics, do some exercises, talk about the new assignment

Coding assignments

- 10 assignments total, done in Python
- To pass the course:
 - All assignments completed
 - At least 60% completed on time (by assigned deadline)
 - Completed means passing all of the unit tests, not just writing some code that does some things
- There will also be one project, bigger than DSA3 assignments, not as big as the DSA2 project, details later

Coding assignments (cont.)

- You're encouraged to work together in groups of two
 - You may not repeat partners over multiple assignments
 - **ASK** before you join someone's repo, we don't want any more Kdrama
- Working on an assignment alone is also fine
- You're also allowed to pair up for the project
 - You can pair with anyone, including prior partners from assignments

Coding assignments (cont.)

- Assignments released Fridays just before tutorial starts
- Assignments due the next NEXT Monday 14:00 just before class time
- This means you have 10 days per assignment
- This also means assignments will overlap

Last bit of administrivia

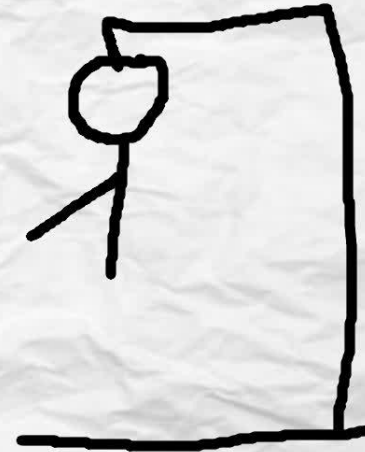
- If you are not yet in the DSACL3 Github team

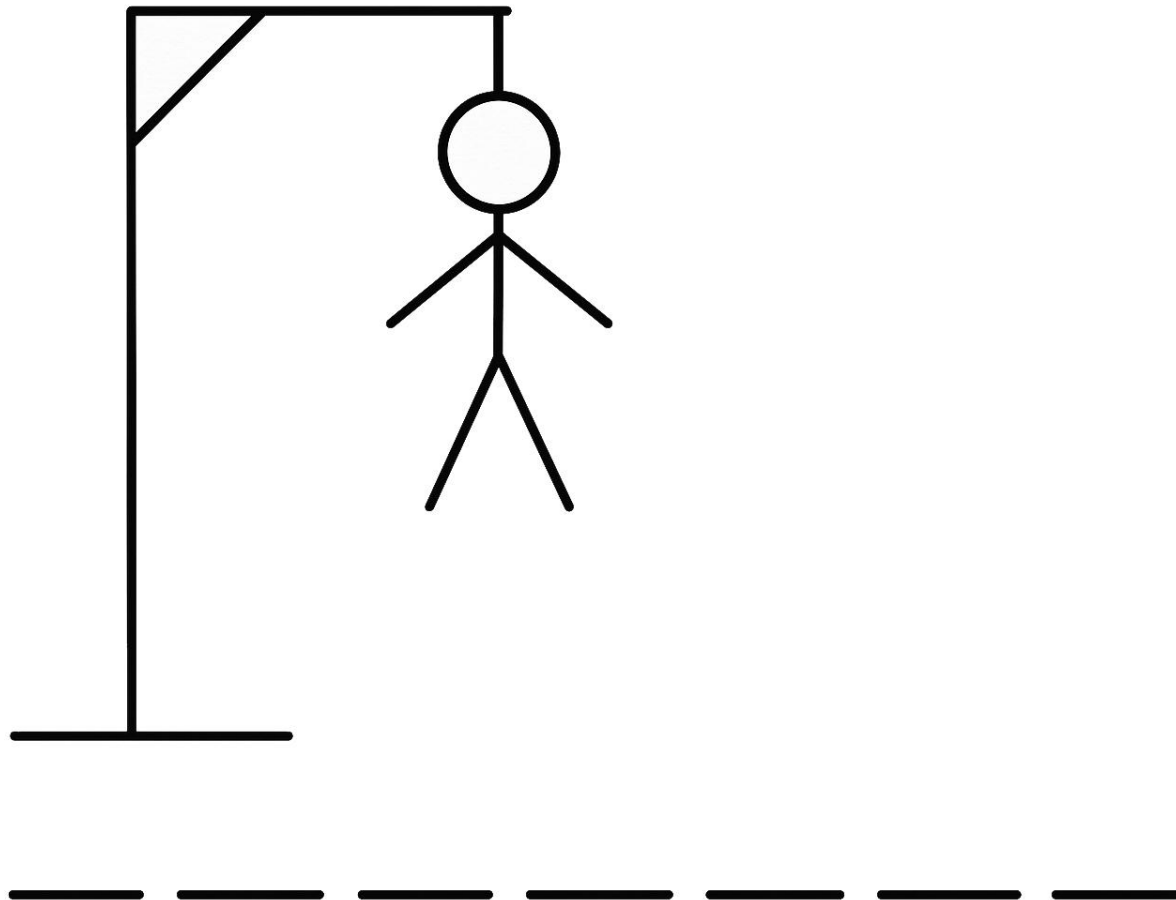
GET IN IT

Let's talk about assignment 1

guess the word

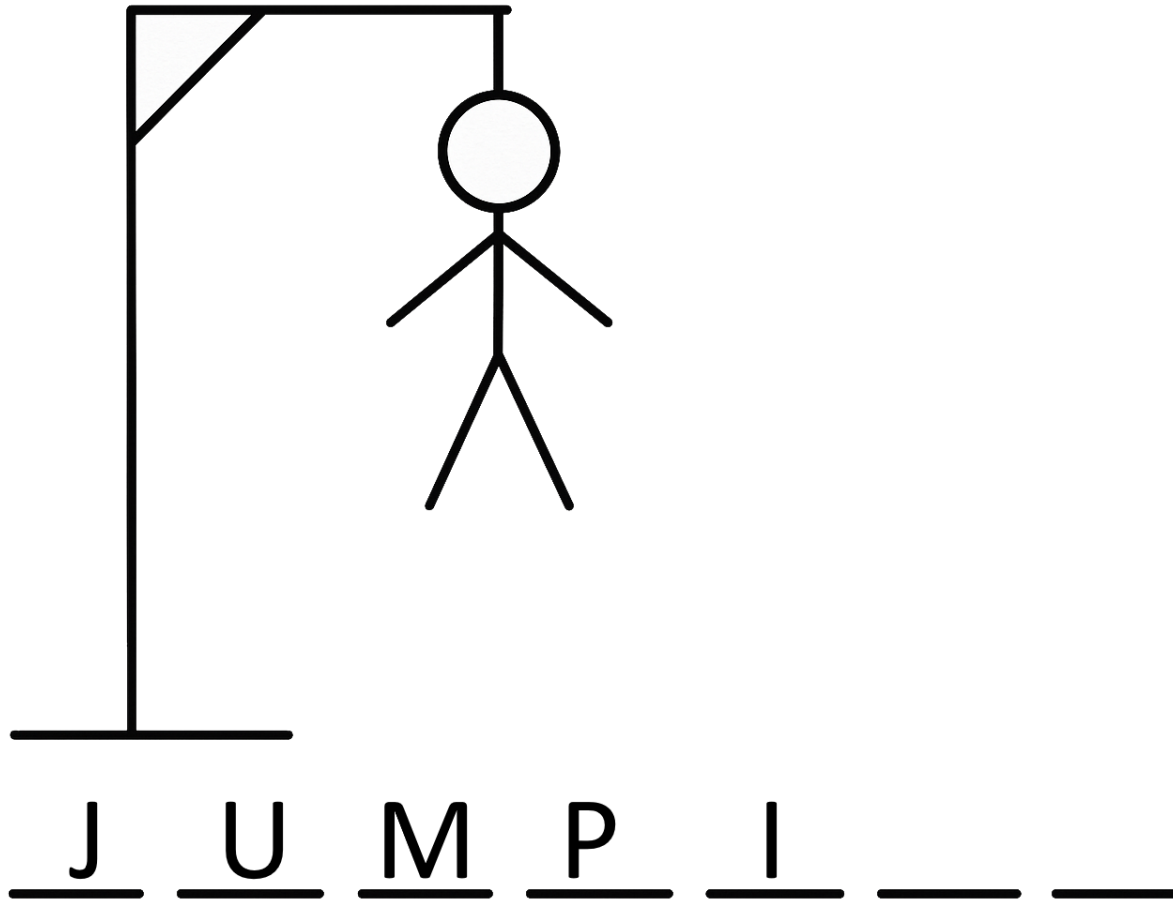
H A N G M A N





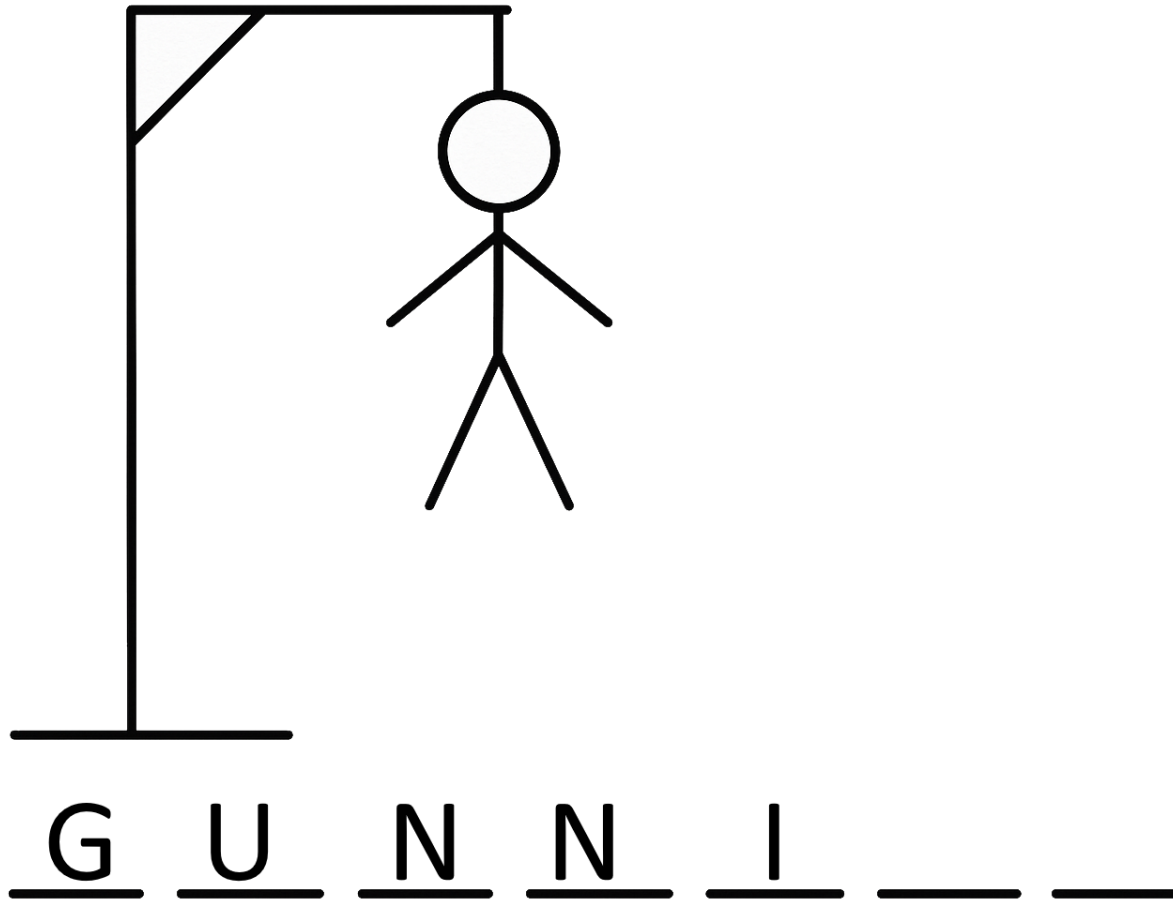
What letter would you guess?

Why?



What letter would you guess?

Why?



What letter would you guess?

Why?

Assignment 1

- We want to make guesses based on probability
- The probability distribution across letters changes depending on context
- But how do we establish a probability distribution?

a0.py

lexicon.txt

lexicon.txt

325

blow

326

blue

327

blur

328

boat

329

body

330

boil

331

bold

332

bomb

333

bond

334

bone

335

boob

336

book

337

boom

338

boot

339

bore

340

born

341

boss

342

bowl

343

boys

344

buck

345

bugs

346

bulb

347

bulk

348

bull

349

bump

350

burn

351

bush

352

bust

353

busy

354

butt

355

cafe

356

cage

357

cake

358

calf

359

calm

360

camp

361

cans

362

cant

363

caps

Ln 7078, Col 10 (8 selected)

Spaces: 4

UTF-8

CRLF

Plain Text

Signed out

A quick review of Python for
proficient Java coders like
yourselves

These two do exactly the same thing...

```
J Main.java X test2.py
J Main.java > ...
1 public class Main {
  Run | Debug
2   public static void main(String[] args) {
3       int x = 0;
4
5       for (int i = 0 ; i < 3 ; i++) {
6           for (int j = 0 ; j < 3 ; j++) {
7               x += 1;
8               x += 2;
9           }
10      }
11      System.out.print(x);
12  }
13 }
14
```

```
J Main.java X test2.py
J Main.java > Main > main(String[])
1 public class Main {
  Run | Debug
2   public static void main(String[] args) {
3       int x = 0;
4
5       for (int i = 0 ; i < 3 ; i++) {
6           for (int j = 0 ; j < 3 ; j++) {
7               x += 1;
8               x += 2;
9           }
10      }
11      System.out.print(x);
12  }
13 }
14
```

These two do not...

```

J Main.java      test2.py
test2.py > ...
1  x = 0
2
3  for i in range(3):
4      for j in range(3):
5          x += 1
6          x += 2
7  print(x)

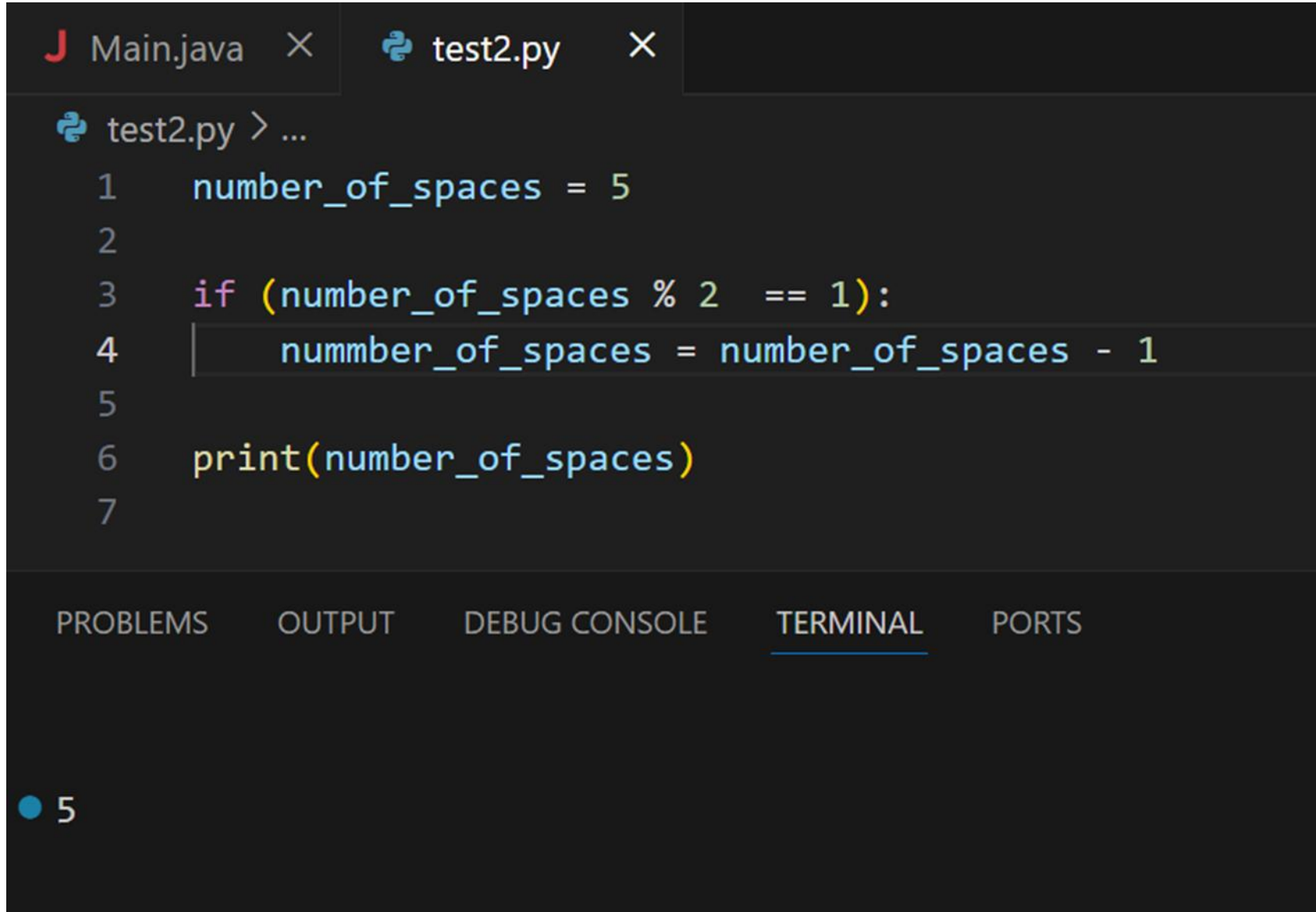
```

```

J Main.java      test2.py
test2.py > ...
1  x = 0
2
3  ✓ for i in range(3):
4  ✓   for j in range(3):
5       x += 1
6       x += 2
7  print(x)

```

What's happening here?



The image shows a screenshot of an IDE with two tabs: 'Main.java' and 'test2.py'. The 'test2.py' tab is active, displaying a Python script. The script initializes 'number_of_spaces' to 5, checks if it is odd, and if so, decrements it by 1. It then prints the value. The output at the bottom shows the number 5.

```
test2.py > ...  
1  number_of_spaces = 5  
2  
3  if (number_of_spaces % 2 == 1):  
4      number_of_spaces = number_of_spaces - 1  
5  
6  print(number_of_spaces)  
7
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

● 5

Dynamically typed variables

test2.py > ...

```
1  some_variable = 10
2  print(some_variable)
3  some_variable = "Norwegian Reggaeton is my spirit animal"
4  print(some_variable)
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

● 10

Norwegian Reggaeton is my spirit animal

Lists, tuples, and sets

test.py > ...

```
1 list = [1, 2, 2, 3]      # ordered, allows duplicates, mutable
2 tuple = (1, 2, 2, 3)    # ordered, allows duplicates, immutable
3 set   = {1, 2, 2, 3}    # unordered, removes duplicates, mutable
4
5 print("List: ", list)
6 print("Tuple:", tuple)
7 print("Set:  ", set)
8
```

PROBLEMS 1

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

```
• List: [1, 2, 2, 3]
  Tuple: (1, 2, 2, 3)
  Set:  {1, 2, 3}
```

And one more...

test2.py > ...

```
1 SNLP = {"Aida": "tutor", "Kyle": "(T_T)", "Miriam": "So lucky she took it with Hinrichs"}
2
3 for key, value in SNLP.items():
4     print(key, ":", value)
5
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
● Aida : tutor
  Kyle : (T_T)
  Miriam : So lucky she took it with Hinrichs
```

And one more...

test2.py > ...

```
1  name_counts = {}
2  # name_counts['Aida'] += 1  this will error
3  name_counts['Aida'] = 1
4  print(name_counts)
5  name_counts['Aida'] = name_counts.get('Aida', 0) + 1
6  name_counts['Miriam'] = name_counts.get('Miriam', 0) + 1
7  print(name_counts)
```

PROBLEMS

OUTPUT

DEBUG CONSOLE

TERMINAL

PORTS

```
● {'Aida': 1}
  {'Aida': 2, 'Miriam': 1}
```

Comprehensions

List comprehension:

[expression for item in iterable (if condition)]

Dictionary comprehension:

{k: v for item in iterable (if condition)}

test.py > ...

```
1  list = [1, 2, 3, 4]
2
3  print([x * 2 for x in list])
4  print(['^_^' for x in list])
5  print([x * 2 for x in list if x % 2 == 0])
6  print([x * 2 if x % 2 == 0 else x for x in list])
```

PROBLEMS

OUTPUT

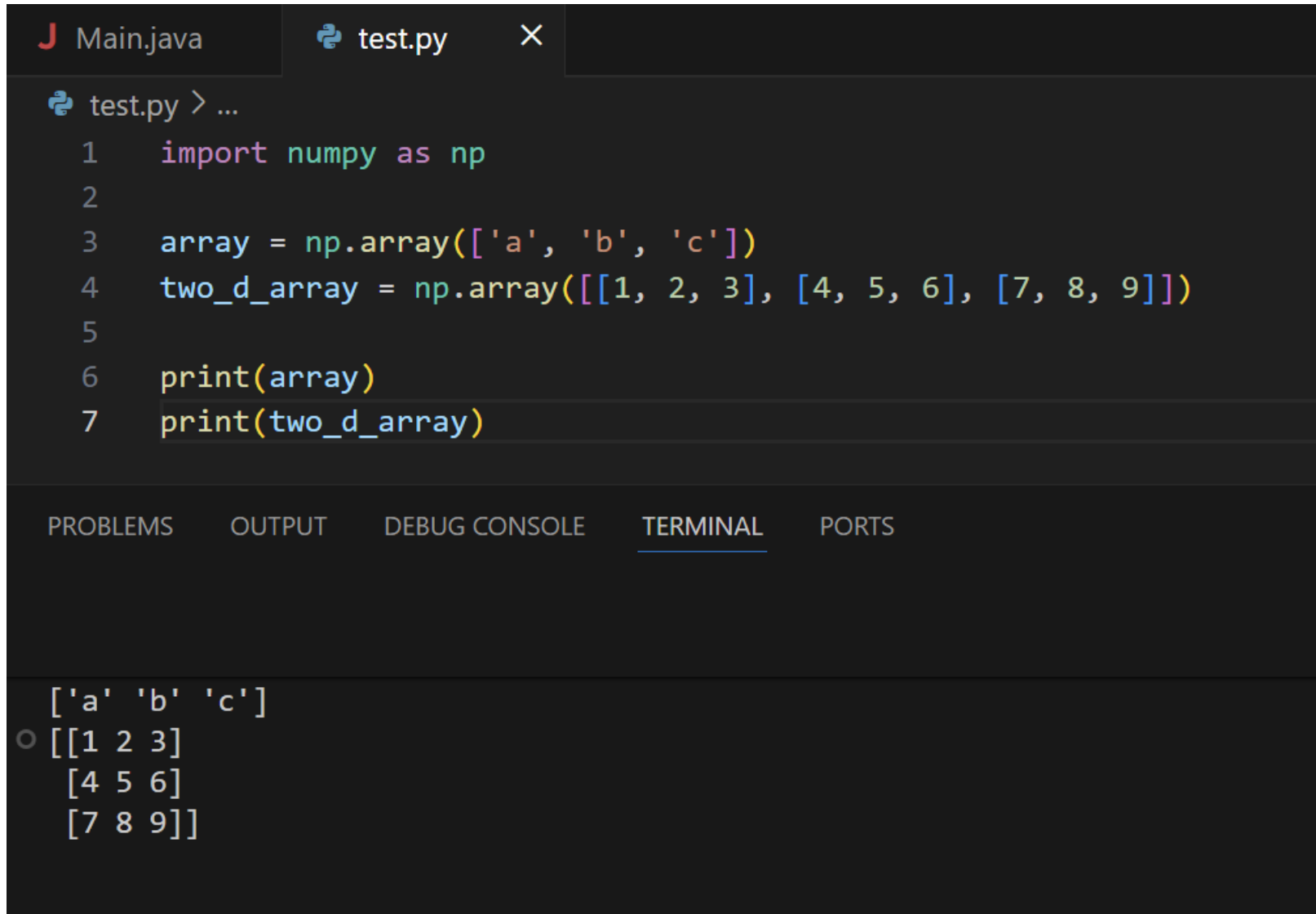
DEBUG CONSOLE

TERMINAL

PORTS

- [2, 4, 6, 8]
 ['^_^', '^_^', '^_^', '^_^']
 [4, 8]
 [1, 4, 3, 8]

Numpy



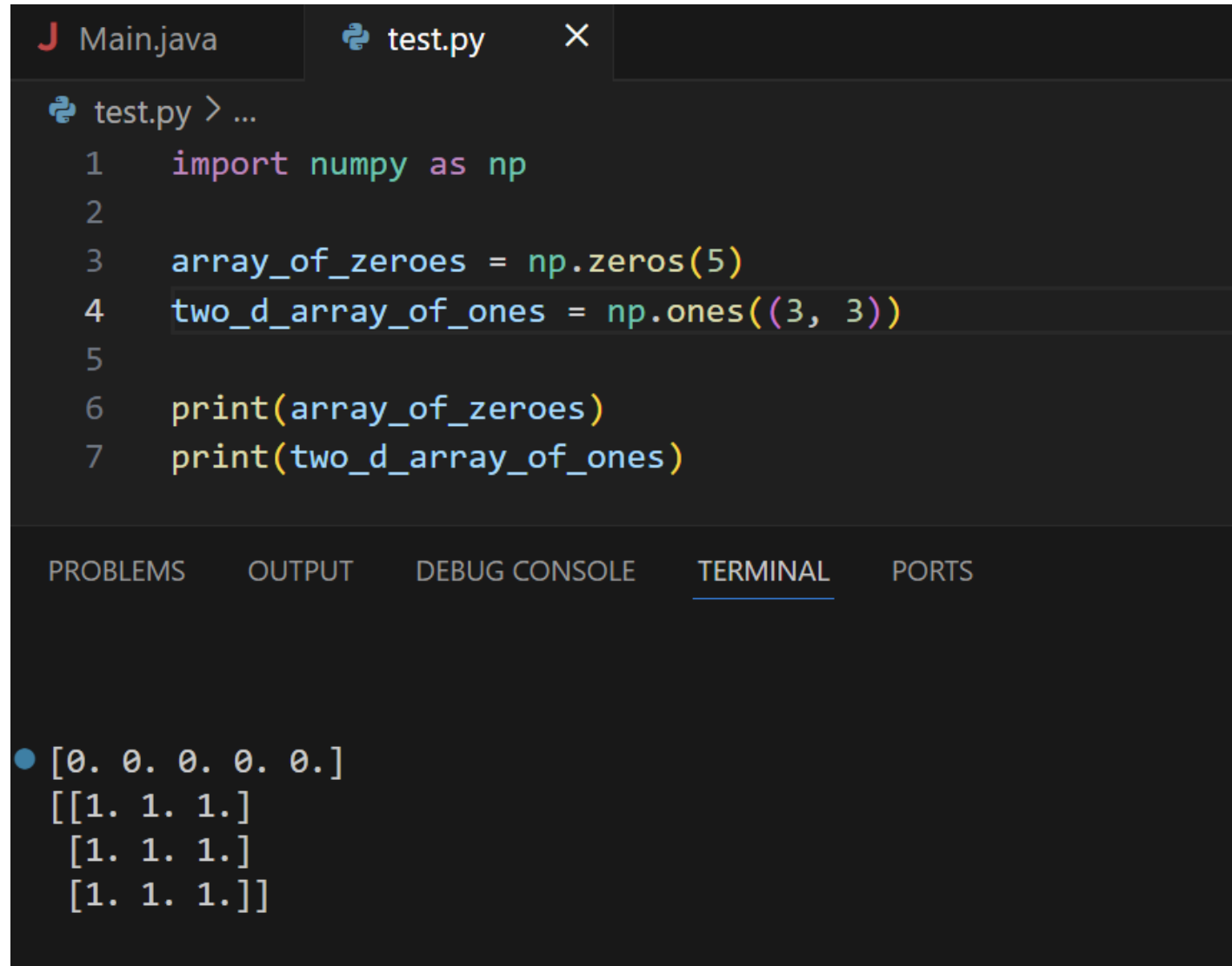
The image shows a code editor window with two tabs: 'Main.java' and 'test.py'. The 'test.py' tab is active, displaying a Python script that imports NumPy and creates two arrays. The script is as follows:

```
1 import numpy as np
2
3 array = np.array(['a', 'b', 'c'])
4 two_d_array = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
5
6 print(array)
7 print(two_d_array)
```

Below the code editor, there is a terminal window with tabs for 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL', and 'PORTS'. The 'TERMINAL' tab is selected, showing the output of the script:

```
['a' 'b' 'c']
[[1 2 3]
 [4 5 6]
 [7 8 9]]
```

More Numpy



```

J Main.java  test.py  X
test.py > ...
1  import numpy as np
2
3  array_of_zeroes = np.zeros(5)
4  two_d_array_of_ones = np.ones((3, 3))
5
6  print(array_of_zeroes)
7  print(two_d_array_of_ones)

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```

● [0. 0. 0. 0. 0.]
  [[1. 1. 1.]
   [1. 1. 1.]
   [1. 1. 1.]]

```


Even more Numpy?

- Go RTFM:
- https://numpy.org/doc/stable/user/absolute_beginners.html