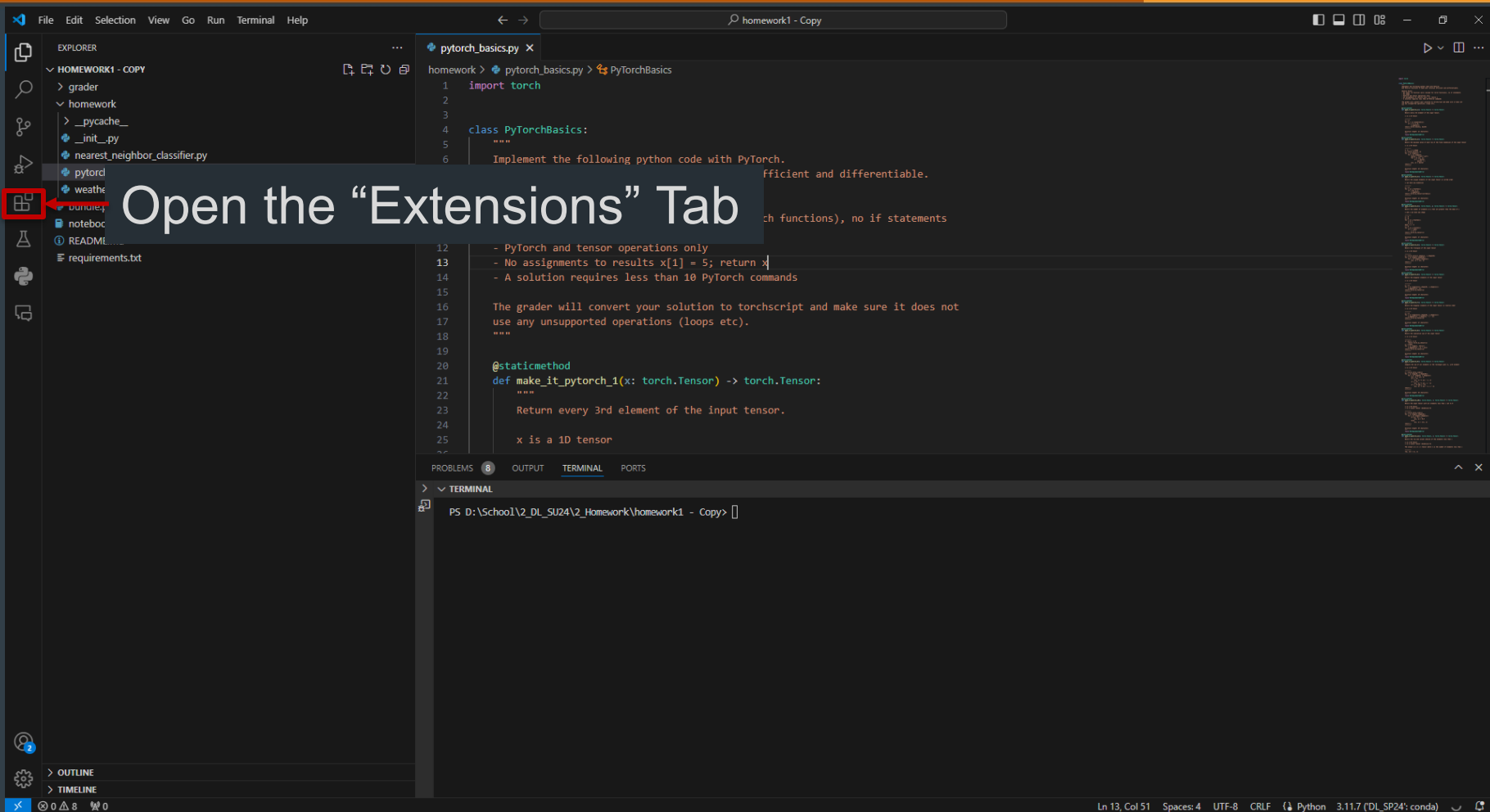


VSCODE

Debugger Setup



File Edit Selection View Go Run Terminal Help

pytorch_basics.py x

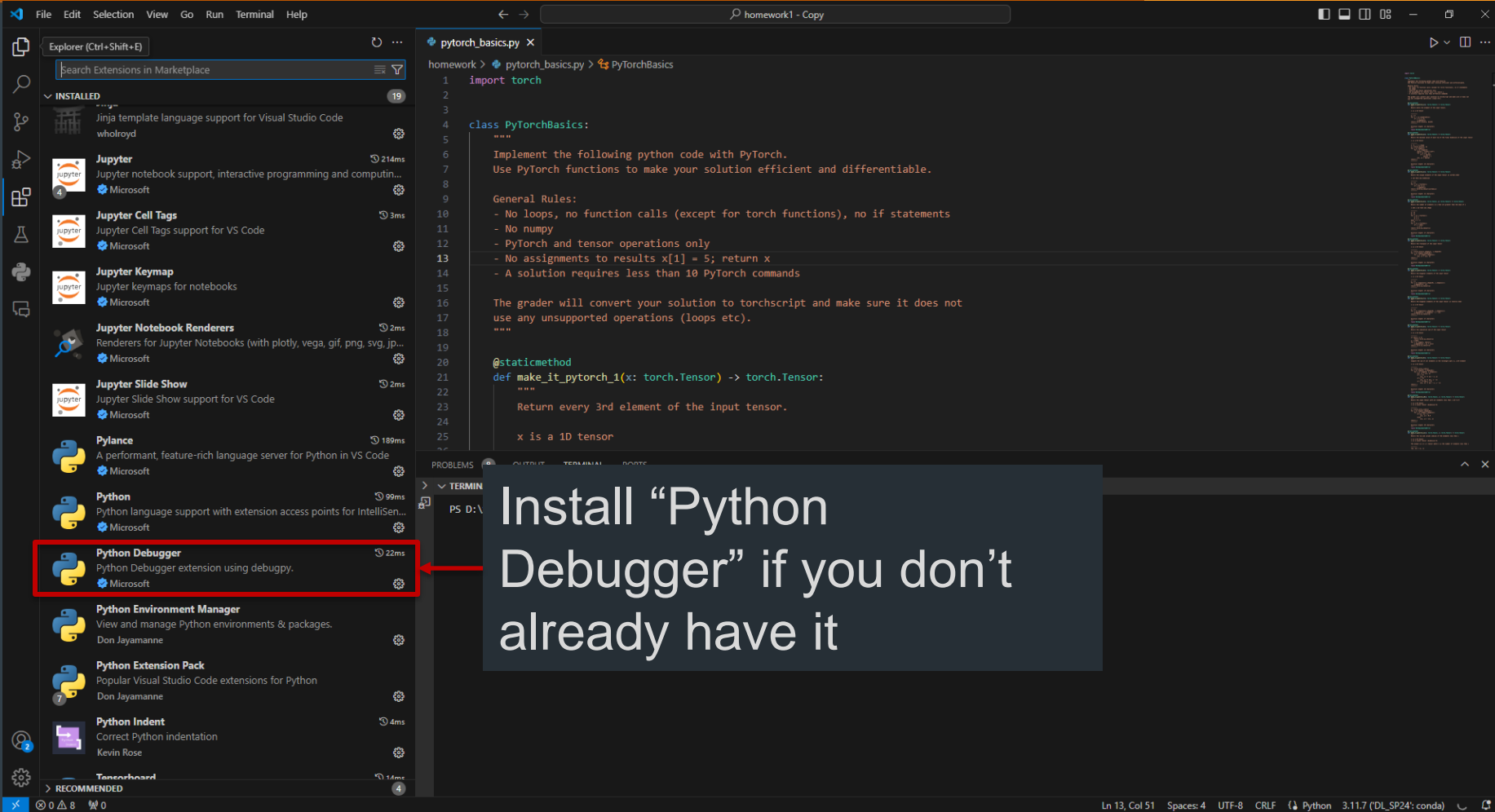
```
homework > pytorch_basics.py > PyTorchBasics
1 import torch
2
3
4 class PyTorchBasics:
5     """
6     Implement the following python code with PyTorch.
7     Efficient and differentiable.
8     (torch functions), no if statements
9
10
11
12 - PyTorch and tensor operations only
13 - No assignments to results x[1] = 5; return x
14 - A solution requires less than 10 PyTorch commands
15
16 The grader will convert your solution to torchscript and make sure it does not
17 use any unsupported operations (loops etc).
18 """
19
20 @staticmethod
21 def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22     """
23     Return every 3rd element of the input tensor.
24
25     x is a 1D tensor
26 """
```

PROBLEMS 8 OUTPUT TERMINAL PORTS

TERMINAL

```
PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>
```

Ln 13, Col 51 Spaces: 4 UTF-8 CRLF Python 3.11.7 (DL_SP24: conda)

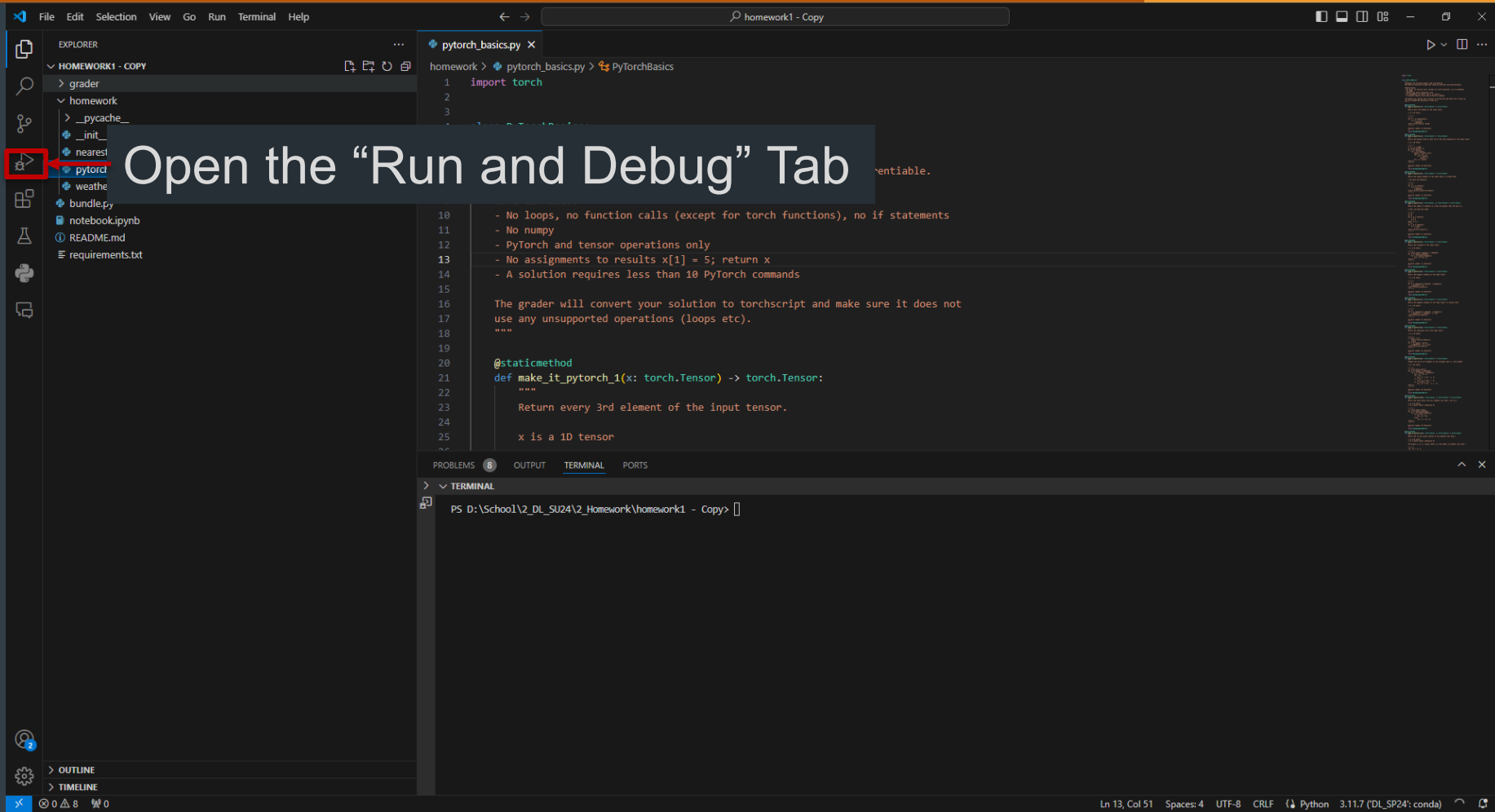


The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left. The Python Debugger extension is highlighted with a red box. The main editor area shows a Python file named `pytorch_basics.py` with the following code:

```
1 import torch
2
3
4 class PyTorchBasics:
5     """
6     Implement the following python code with PyTorch.
7     Use PyTorch functions to make your solution efficient and differentiable.
8
9     General Rules:
10    - No loops, no function calls (except for torch functions), no if statements
11    - No numpy
12    - PyTorch and tensor operations only
13    - No assignments to results x[1] = 5; return x
14    - A solution requires less than 10 PyTorch commands
15
16    The grader will convert your solution to torchscript and make sure it does not
17    use any unsupported operations (loops etc).
18    """
19
20    @staticmethod
21    def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22        """
23        Return every 3rd element of the input tensor.
24
25        x is a 1D tensor
```

A red arrow points from the text overlay to the Python Debugger extension in the sidebar.

Install “Python Debugger” if you don’t already have it



Open the “Run and Debug” Tab

The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar shows a project named 'HOMEWORK1 - COPY'. Under the 'homework' folder, the file 'pytorch' is highlighted. A red box is drawn around the 'Run and Debug' icon (a play button with a bug) in the left sidebar. A semi-transparent dark box with white text 'Open the “Run and Debug” Tab' is overlaid on the image, with an arrow pointing to the red box. The main editor area shows the 'pytorch_basics.py' file with the following code:

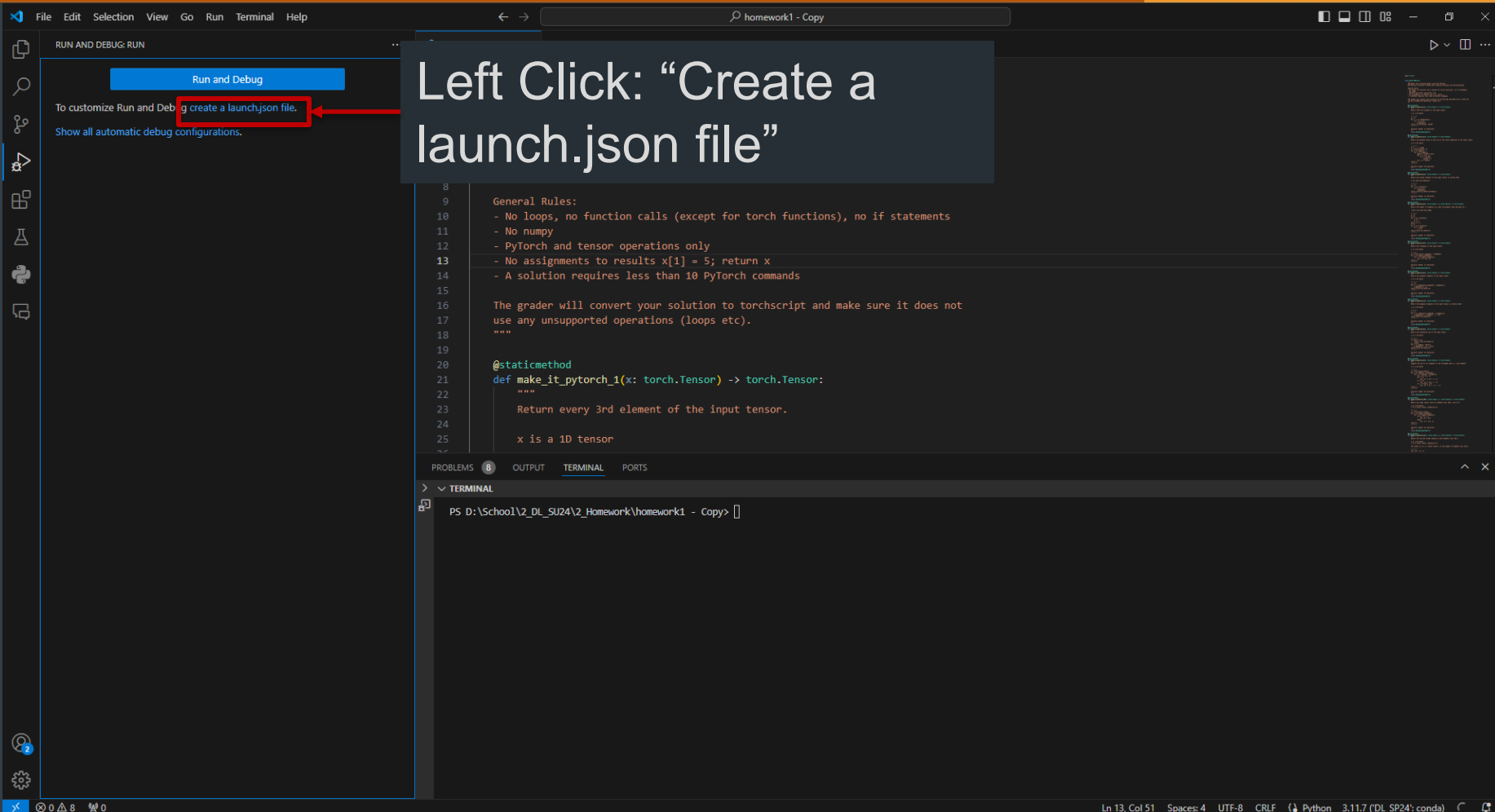
```
1 import torch
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
```

The code includes comments and a static method definition:

```
10 - No loops, no function calls (except for torch functions), no if statements
11 - No numpy
12 - PyTorch and tensor operations only
13 - No assignments to results x[1] = 5; return x
14 - A solution requires less than 10 PyTorch commands
15
16 The grader will convert your solution to torchscript and make sure it does not
17 use any unsupported operations (loops etc).
18
19
20 @staticmethod
21 def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22     """
23     Return every 3rd element of the input tensor.
24
25     x is a 1D tensor
```

The bottom of the interface shows the 'TERMINAL' tab with the command prompt:

```
PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>
```



Left Click: "Create a launch.json file"

Run and Debug

To customize Run and Debug, create a launch.json file.

Show all automatic debug configurations.

```
8
9
10 General Rules:
11 - No loops, no function calls (except for torch functions), no if statements
12 - No numpy
13 - PyTorch and tensor operations only
14 - No assignments to results x[1] = 5; return x
15 - A solution requires less than 10 PyTorch commands
16
17 The grader will convert your solution to torchscript and make sure it does not
18 use any unsupported operations (loops etc).
19
20 @staticmethod
21 def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22     """
23     Return every 3rd element of the input tensor.
24
25     x is a 1D tensor
```

PROBLEMS 8 OUTPUT TERMINAL PORTS

TERMINAL

PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>

Ln 13, Col 51 Spaces: 4 UTF-8 CRLF Python 3.11.7 (DL_SP24: conda)

File Edit Selection View Go Run Terminal Help

RUN AND DEBUG: RUN

Run and Debug

To customize Run and Debug [create a launch.json file](#).[Show all automatic debug configurations.](#)

Select debugger

Python Debugger

Suggested

pytorch_basics.py Install an extension for Python...

```
1 import torch
```

```
4 class PyTorchBasics:
```

Left Click on “Python Debugger”

```
16 The grader will convert your solution to torchscript and make sure it does not
```

```
17 use any unsupported operations (loops etc).
```

```
20 @staticmethod
```

```
21 def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
```

```
22     """
```

```
23     Return every 3rd element of the input tensor.
```

```
24     x is a 1D tensor
```

PROBLEMS 8

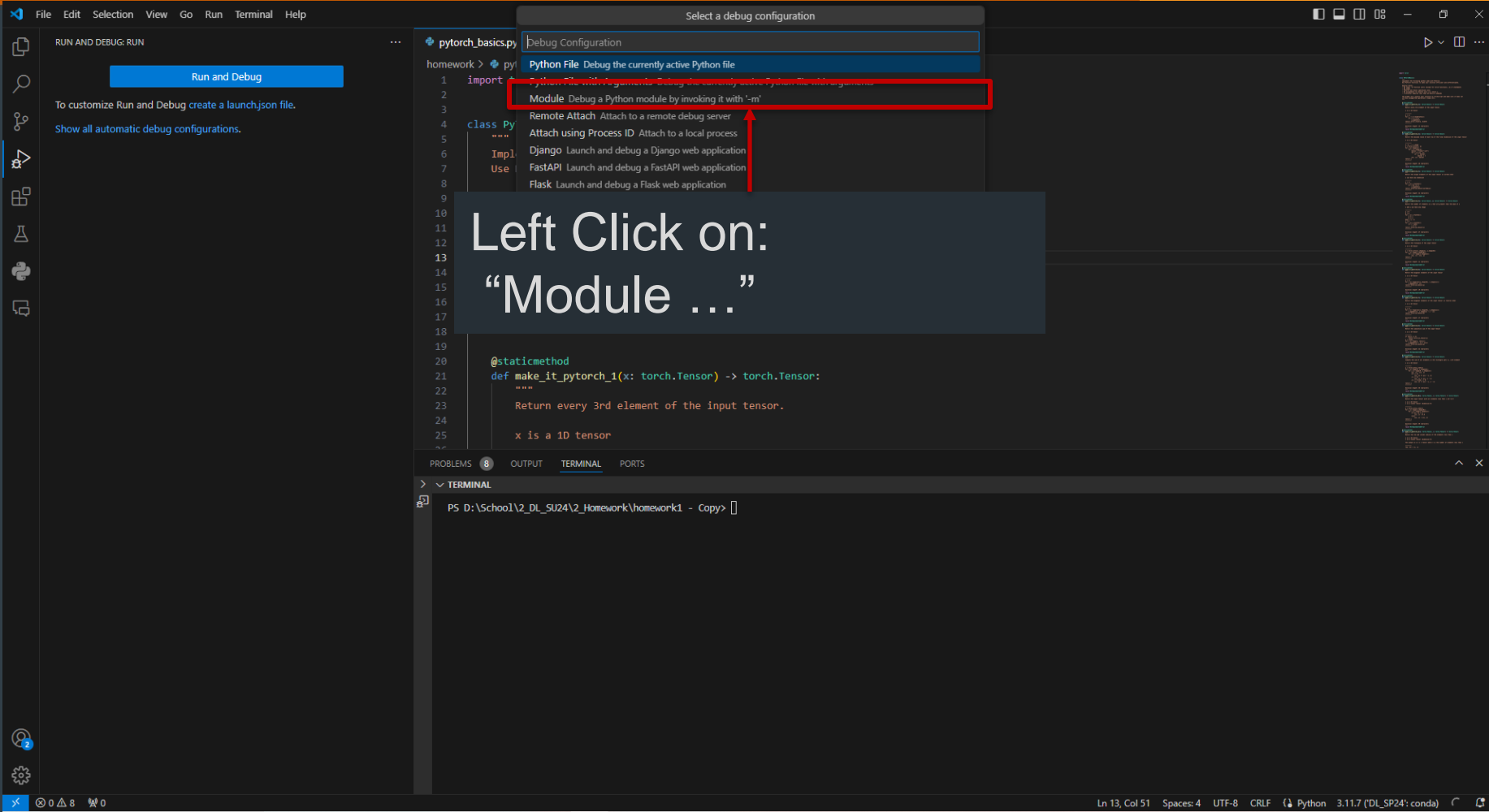
OUTPUT

TERMINAL

PORTS



PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>



The image shows the Visual Studio Code (VS Code) interface with the 'Select a debug configuration' dialog open. The dialog lists several debug configurations for Python. The 'Module' option, which is described as 'Debug a Python module by invoking it with "-m"', is highlighted with a red box. A red arrow points to this option from a text overlay that reads 'Left Click on: "Module ..."'. The background shows a Python file named 'pytorch_basics.py' with some code visible, including a class definition and a static method. The terminal at the bottom shows the command prompt 'PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>'.

File Edit Selection View Go Run Terminal Help

RUN AND DEBUG: RUN

Run and Debug

To customize Run and Debug create a launch.json file.

Show all automatic debug configurations.

Select a debug configuration

Debug Configuration

Python File Debug the currently active Python file

Python File with Arguments Debug the currently active Python file with arguments

Module Debug a Python module by invoking it with "-m"

Remote Attach Attach to a remote debug server

Attach using Process ID Attach to a local process

Impl Django Launch and debug a Django web application

Use FastAPI Launch and debug a FastAPI web application

Flask Launch and debug a Flask web application

Left Click on:
"Module ..."

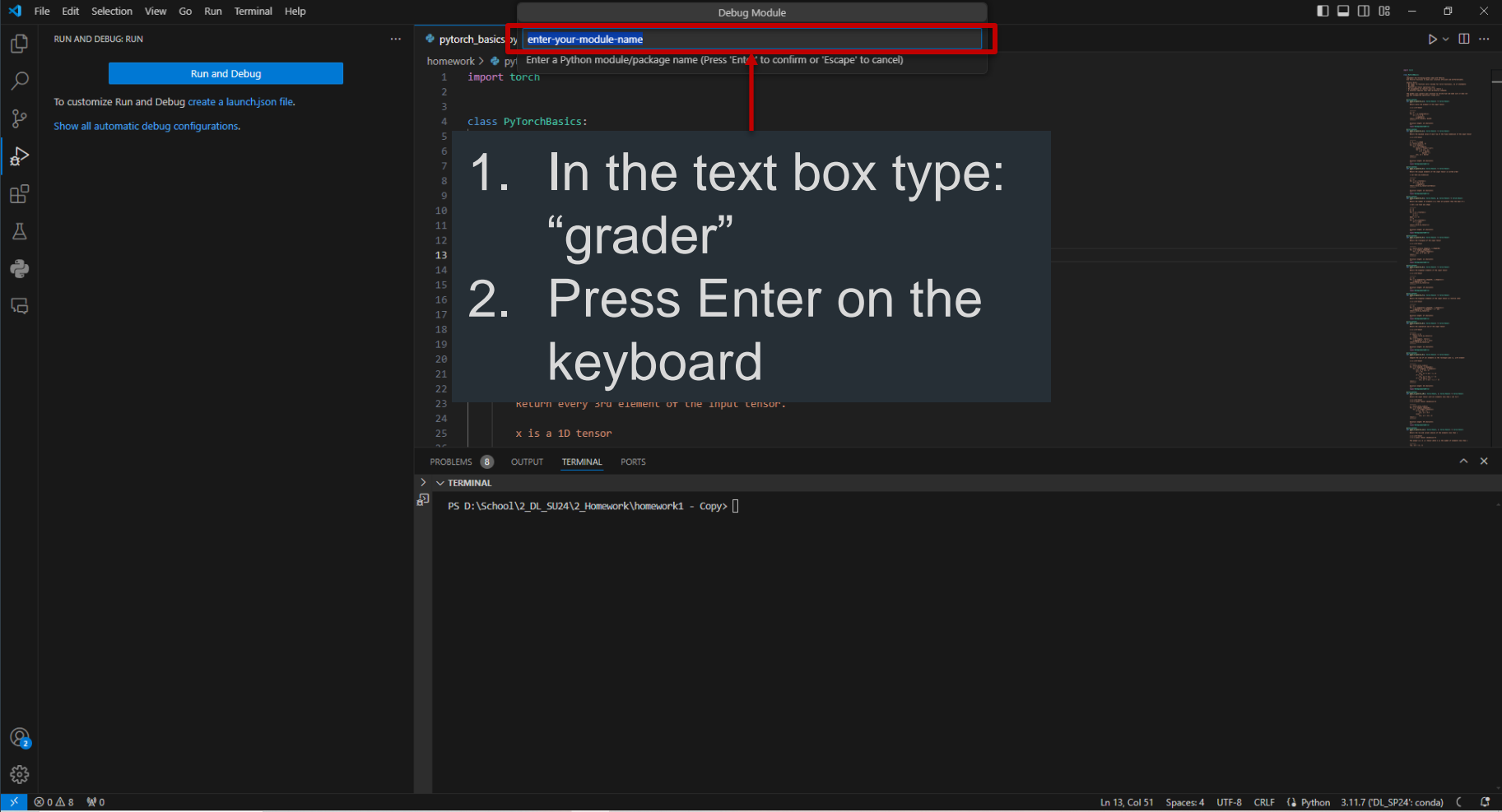
```
1 import torch
2
3
4 class PyTorchBasic:
5     """
6     Impl: Django Launch and debug a Django web application
7     Use: FastAPI Launch and debug a FastAPI web application
8     Flask Launch and debug a Flask web application
9
10
11
12
13
14
15
16
17
18
19
20 @staticmethod
21 def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22     """
23     Return every 3rd element of the input tensor.
24
25     x is a 1D tensor
26 """
```

PROBLEMS 0 OUTPUT TERMINAL PORTS

TERMINAL

PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>

Ln 13, Col 51 Spaces: 4 UTF-8 CRLF Python 3.11.7 (DL_SP24\conda)



File Edit Selection View Go Run Terminal Help

RUN AND DEBUG: RUN

Run and Debug

To customize Run and Debug create a launch.json file.

Show all automatic debug configurations.

Debug Module

pytorch_basics.py enter-your-module-name

homework > py Enter a Python module/package name (Press 'Enter' to confirm or 'Escape' to cancel)

```
1 import torch
2
3
4 class PyTorchBasics:
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23     return every 3rd element of the input tensor.
24
25     x is a 1D tensor
```

1. In the text box type:
"grader"

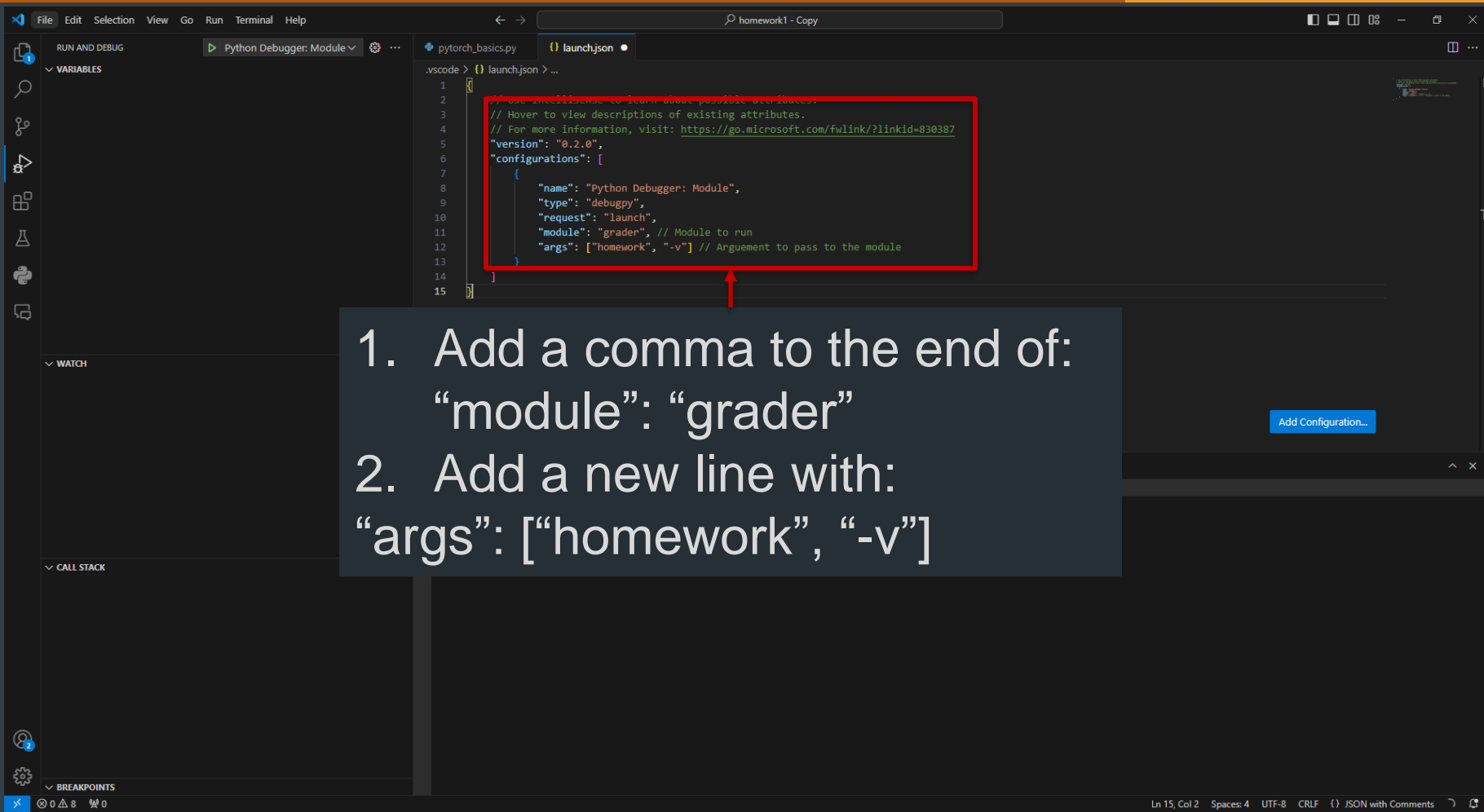
2. Press Enter on the
keyboard

PROBLEMS 8 OUTPUT TERMINAL PORTS

TERMINAL

PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy> []

Ln 13, Col 51 Spaces: 4 UTF-8 CRLF Python 3.11.7 (DL_SP24: conda)



The screenshot shows the VS Code editor with a `launch.json` file open. The configuration is for the Python Debugger. A red box highlights the following JSON structure:

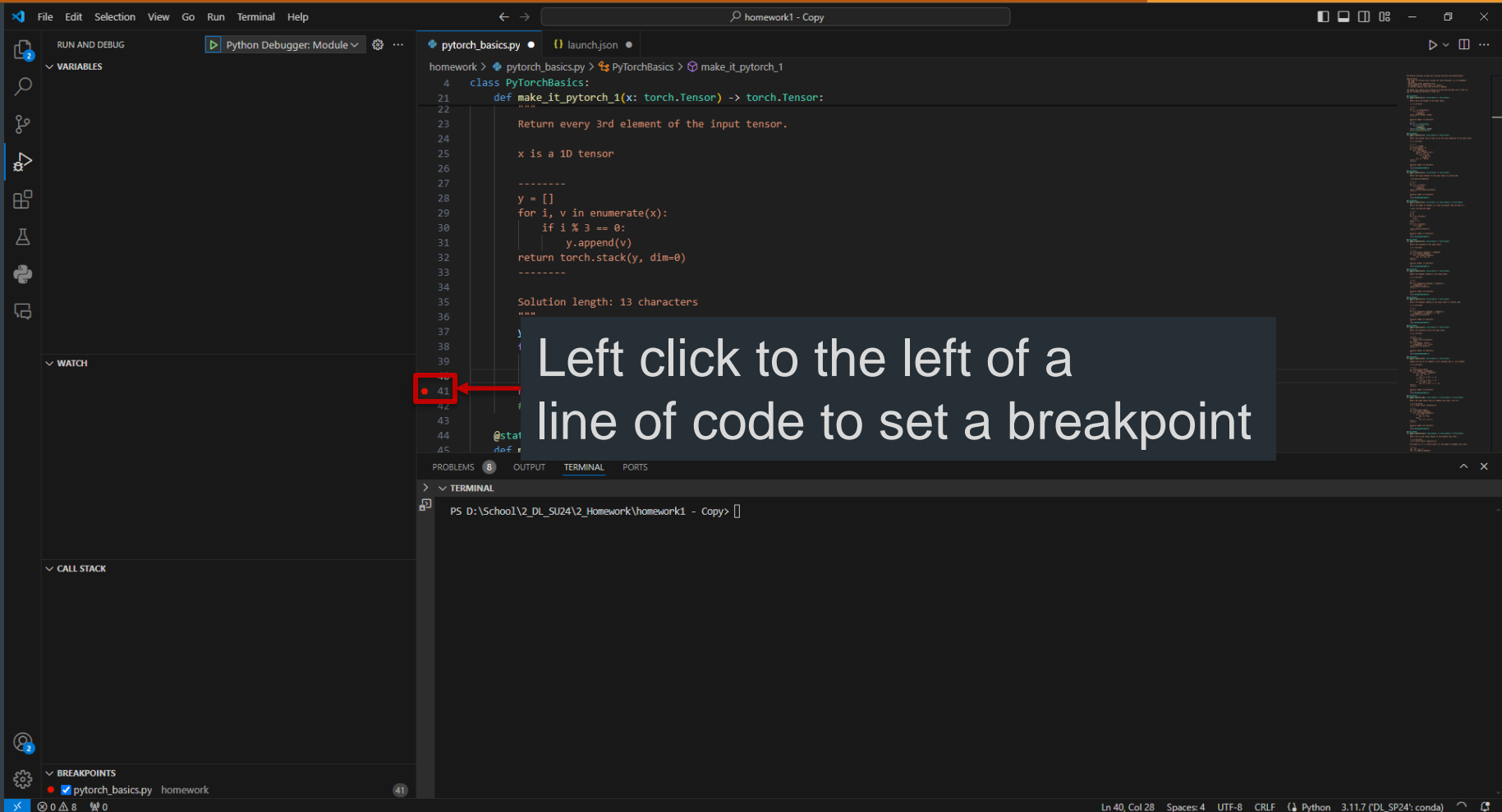
```
{
  // Use IntelliSense to learn about possible attributes.
  // Hover to view descriptions of existing attributes.
  // For more information, visit: https://go.microsoft.com/fwlink/?linkid=830387
  "version": "0.2.0",
  "configurations": [
    {
      "name": "Python Debugger: Module",
      "type": "debugpy",
      "request": "launch",
      "module": "grader", // Module to run
      "args": ["homework", "-v"] // Argument to pass to the module
    }
  ]
}
```

A red arrow points to the `"args": ["homework", "-v"]` line.

1. Add a comma to the end of:
"module": "grader"
2. Add a new line with:
"args": ["homework", "-v"]

Buttons: Add Configuration...

Status bar: Ln 15, Col 2 | Spaces: 4 | UTF-8 | CRLF | {} JSON with Comments



The screenshot shows the Visual Studio Code interface with a Python file named `pytorch_basics.py` open. The file contains a class `PyTorchBasics` with a method `make_it_pytorch_1`. A red dot on the left margin of line 41 indicates a breakpoint. A red arrow points from a text box to this breakpoint. The text box contains the instruction: "Left click to the left of a line of code to set a breakpoint". The bottom status bar shows the current line and column as "Ln 40, Col 28".

```
4 class PyTorchBasics:
21     def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22         """
23         Return every 3rd element of the input tensor.
24
25         x is a 1D tensor
26
27         -----
28         y = []
29         for i, v in enumerate(x):
30             if i % 3 == 0:
31                 y.append(v)
32         return torch.stack(y, dim=0)
33         -----
34
35         Solution length: 13 characters
36         """
37
38
39
40
41
42
43
44
45
```

Left click to the left of a
line of code to set a breakpoint

PROBLEMS 8 OUTPUT TERMINAL PORTS

TERMINAL

PS D:\School\2_DL_SJ24\2_Homework\homework1 - Copy>

0 8 0

Ln 40, Col 28 Spaces: 4 UTF-8 CRLF Python 3.11.7 (DL_SP24: conda)

Left click the green arrow or press F5 to start the debugger

File Edit Selection View Go Run Terminal Help

RUN AND DEBUG  Python Debugger

VARIABLES

```
23     Return every 3rd element of the input tensor.
24
25     x is a 1D tensor
26
27     -----
28     y = []
29     for i, v in enumerate(x):
30         if i % 3 == 0:
31             y.append(v)
32     return torch.stack(y, dim=0)
33     -----
34
35     Solution length: 13 characters
36     """
37     y = []
38     for i, v in enumerate(x):
39         if i % 3 == 0:
40             y.append(v)
41     return torch.stack(y, dim=0)
42     #raise NotImplementedError
43
44     @staticmethod
45     def make_it_ntorch_2(x: torch.Tensor) -> torch.Tensor:
```

PROBLEMS 8 OUTPUT TERMINAL PORTS

> TERMINAL

PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy> []

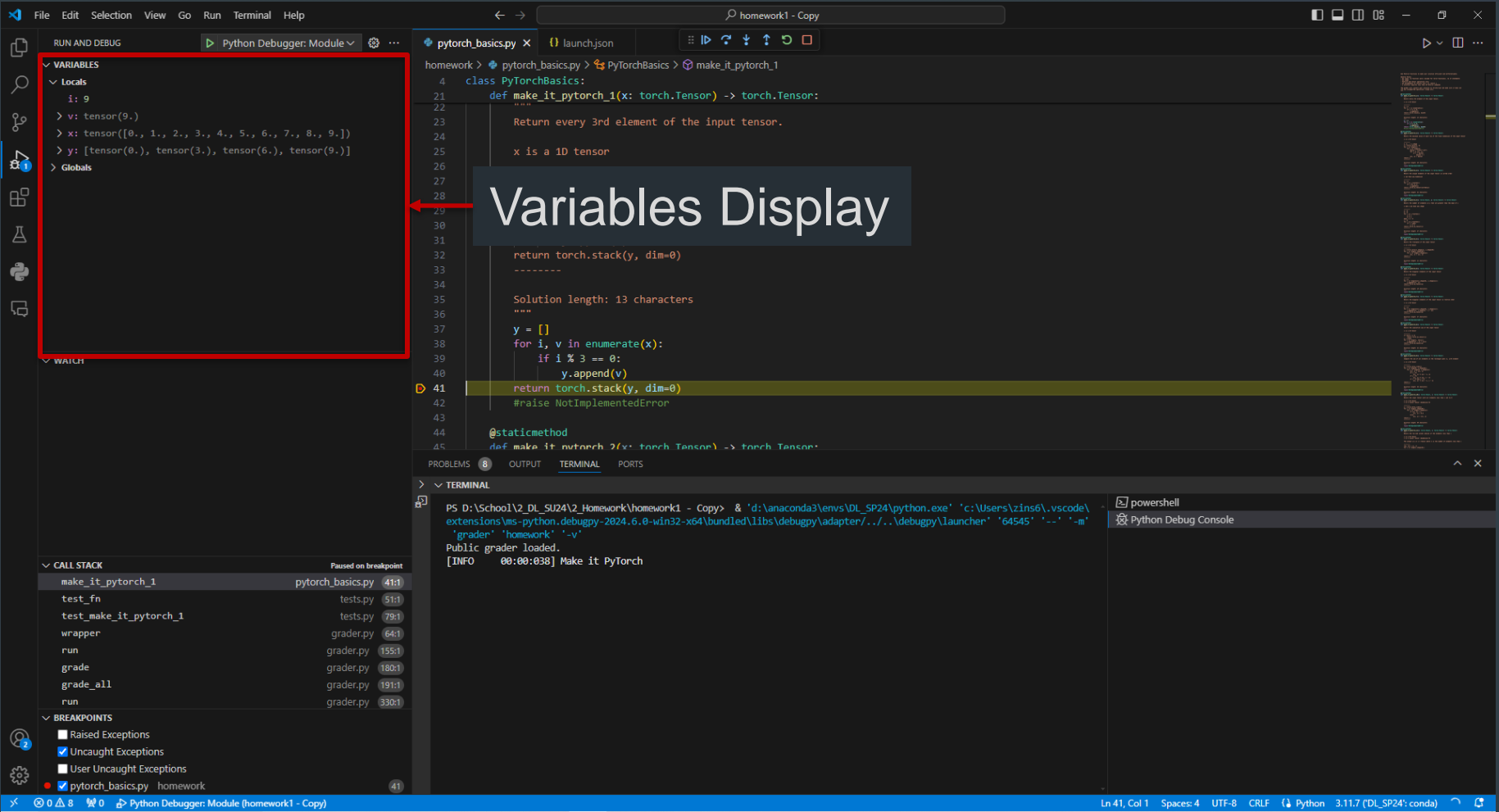
CALL STACK

BREAKPOINTS
pytorch_basics.py homework

41

x 0 8 0

Ln 40, Col 28 Spaces: 4 UTF-8 CRLF Python 3.11.7 (DL_SP24: conda)



The screenshot shows the VS Code Python Debugger interface. The **VARIABLES** panel on the left is highlighted with a red box and labeled "Variables Display" with a red arrow. It shows the current state of the program's variables:

- Variables:**
 - `i`: 9
 - `v`: `tensor(9.)`
 - `x`: `tensor([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])`
 - `y`: `[tensor(0.), tensor(3.), tensor(6.), tensor(9.)]`
- Globals:** (empty)

The main editor shows the `pytorch_basics.py` file. The code defines a `PyTorchBasics` class with a `make_it_pytorch_1` method. The method takes a `torch.Tensor` as input and returns a `torch.Tensor`. The code is currently paused at line 41, which is `return torch.stack(y, dim=0)`.

The **CALL STACK** panel at the bottom left shows the current call stack:

Function	File	Line
<code>make_it_pytorch_1</code>	<code>pytorch_basics.py</code>	41:1
<code>test_fn</code>	<code>tests.py</code>	51:1
<code>test_make_it_pytorch_1</code>	<code>tests.py</code>	79:1
<code>wrapper</code>	<code>grader.py</code>	64:1
<code>run</code>	<code>grader.py</code>	155:1
<code>grade</code>	<code>grader.py</code>	180:1
<code>grade_all</code>	<code>grader.py</code>	191:1
<code>run</code>	<code>grader.py</code>	330:1

The **TERMINAL** panel at the bottom right shows the command prompt output:

```
PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy> & 'd:\anaconda3\envs\DL_SP24\python.exe' 'c:\Users\zins6\.vscode\extensions\ms-python.debugpy-2024.6.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '64545' '--' '-m' 'grader' 'homework1' '-v'  
Public grader loaded.  
[INFO 00:00:038] Make it PyTorch
```

The image shows the VS Code Python Debugger interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The left sidebar contains icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The Run and Debug sidebar is open, showing the Python Debugger: Module view. It includes sections for VARIABLES (Locals, Globals), WATCH, CALL STACK, and BREAKPOINTS. The main editor displays the file 'pytorch_basics.py' with a class 'PyTorchBasics' and a method 'make_it_pytorch_1'. A red box highlights the debugger controls in the top right of the editor, including buttons for Run, Step Over, Step Into, Step Out, and Stop. A semi-transparent box with the text 'Debugger Controls' is overlaid on the code editor. The bottom panel shows the PROBLEMS, OUTPUT, TERMINAL, and PORTS tabs. The TERMINAL tab is active, showing the command prompt and the output of the program.

Debugger Controls

```

4 class PyTorchBasics:
21     def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22         """
23         Return even elements of x as a 1D tensor.
24
25         x is a 1D
26
27         -----
28         y = []
29         for i, v in enumerate(x):
30             if i % 2 == 0:
31                 y.append(v)
32         return torch.stack(y, dim=0)
33         -----
34
35         Solution length: 13 characters
36         """
37         y = []
38         for i, v in enumerate(x):
39             if i % 2 == 0:
40                 y.append(v)
41         return torch.stack(y, dim=0)
42         #raise NotImplementedError
43
44     @staticmethod
45     def make_it_pytorch_2(x: torch.Tensor) -> torch.Tensor:

```

CALL STACK

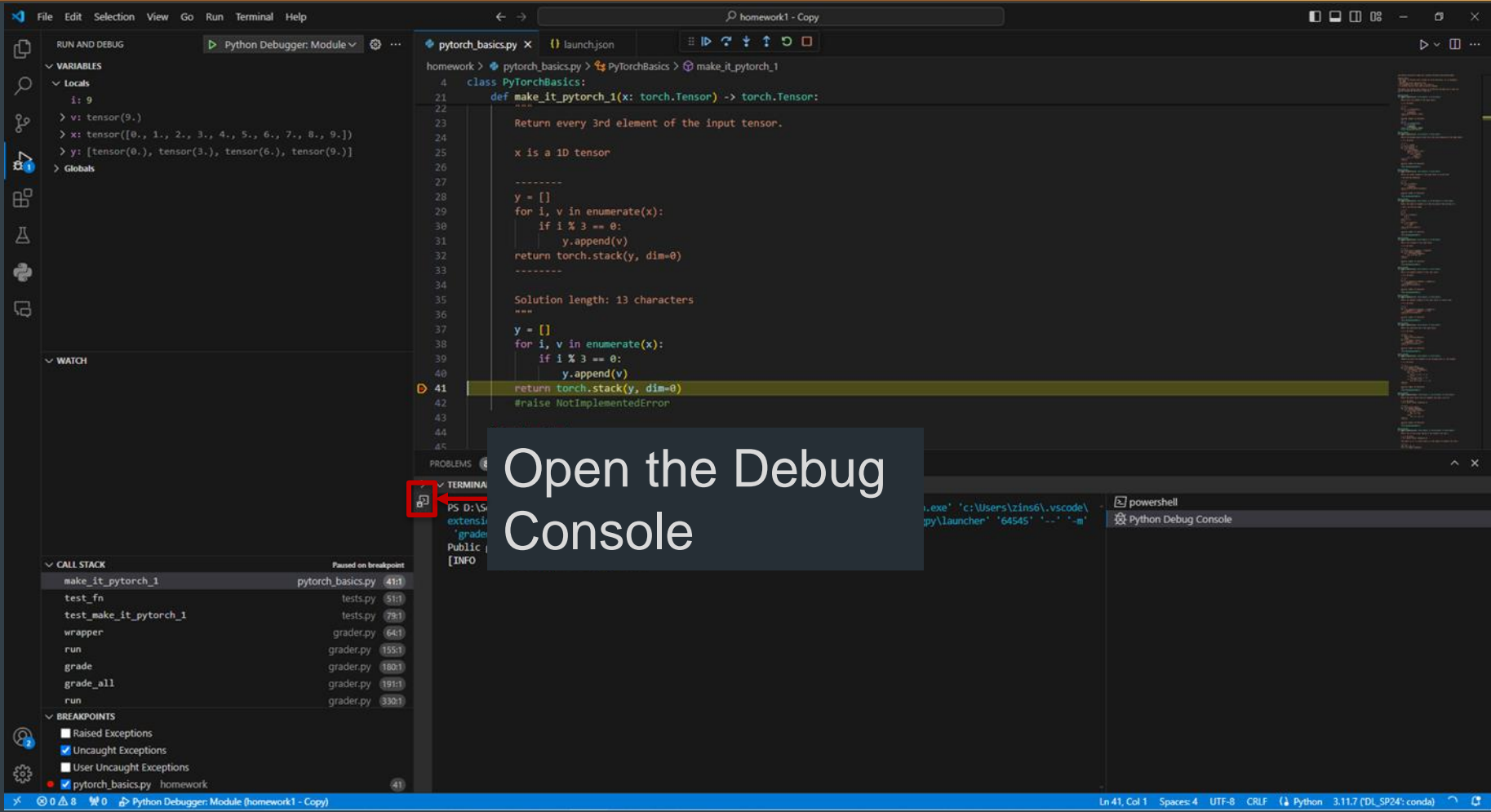
Function	File	Line
make_it_pytorch_1	pytorch_basics.py	41:1
test_fn	tests.py	51:1
test_make_it_pytorch_1	tests.py	79:1
wrapper	grader.py	64:1
run	grader.py	155:1
grade	grader.py	180:1
grade_all	grader.py	191:1
run	grader.py	330:1

TERMINAL

```

PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy> & 'd:\anaconda3\envs\DL_SP24\python.exe' 'c:\Users\zins6\.vscode\extensions\ms-python.debugpy-2024.6.0-win32-x64\bundled\libs\debugpy\adapter\..\..\debugpy\launcher' '64545' '--' '-m' 'grader' 'homework' '-v'
Public grader loaded.
[INFO 00:00:038] Make it PyTorch

```



The image shows a Visual Studio Code editor with a Python debugger. The main editor displays a file named `pytorch_basics.py` with the following code:

```

4 class PyTorchBasics:
21     def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22         """
23         Return every 3rd element of the input tensor.
24
25         x is a 1D tensor
26
27         -----
28         y = []
29         for i, v in enumerate(x):
30             if i % 3 == 0:
31                 y.append(v)
32         return torch.stack(y, dim=0)
33         """
34
35         Solution length: 13 characters
36         """
37         y = []
38         for i, v in enumerate(x):
39             if i % 3 == 0:
40                 y.append(v)
41         return torch.stack(y, dim=0)
42         #raise NotImplementedError
43
44
45

```

The left sidebar shows the **VARIABLES** panel with the following state:

```

> VARIABLES
  > Locals
    i: 9
    > v: tensor(9.)
    > x: tensor([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])
    > y: [tensor(0.), tensor(3.), tensor(6.), tensor(9.)]
  > Globals

```

The bottom panel shows the **CALL STACK** with the following entries:

Function	File	Line
make_it_pytorch_1	pytorch_basics.py	41:1
test_fn	tests.py	51:1
test_make_it_pytorch_1	tests.py	78:1
wrapper	grader.py	64:1
run	grader.py	155:1
grade	grader.py	180:1
grade_all	grader.py	191:1
run	grader.py	330:1

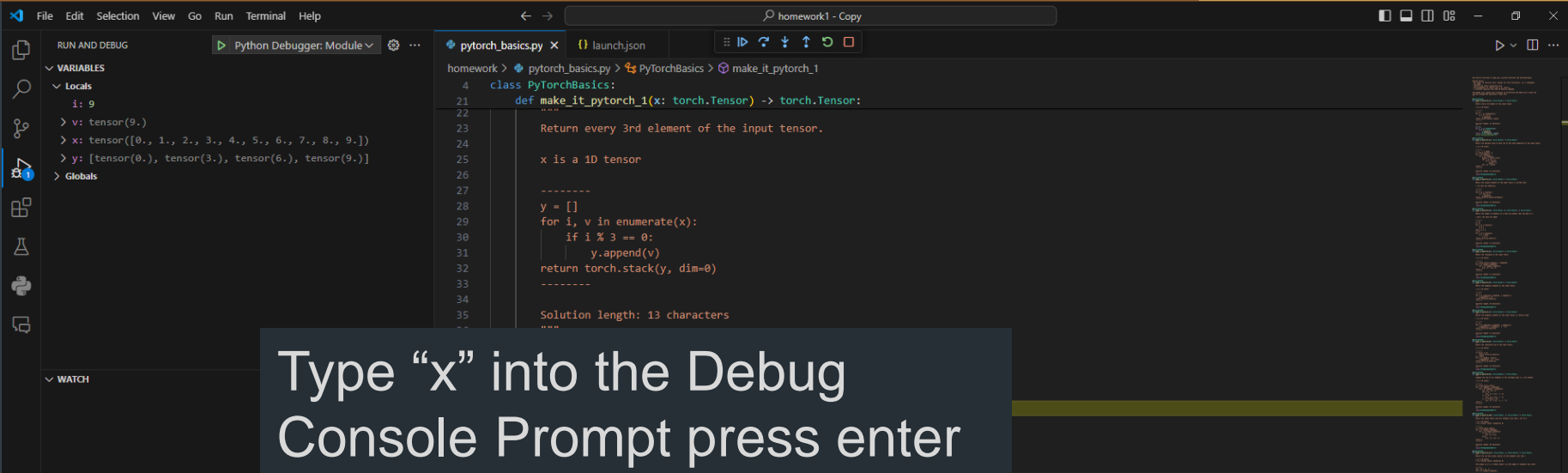
A red box highlights the **Open the Debug Console** icon in the **TERMINAL** panel. The terminal shows the following output:

```

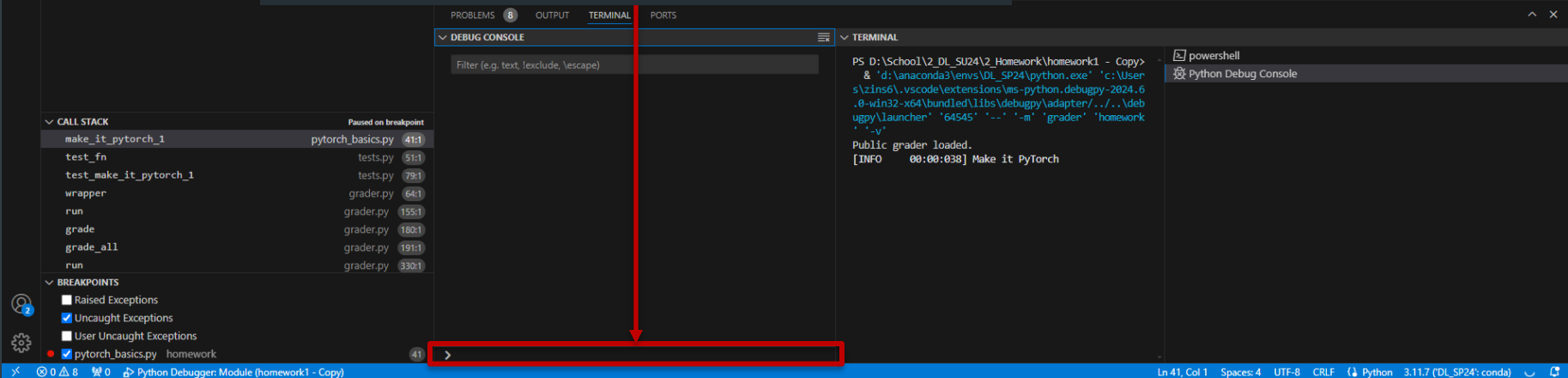
PS D:\S\extensiv\grader\Public [INFO]

```

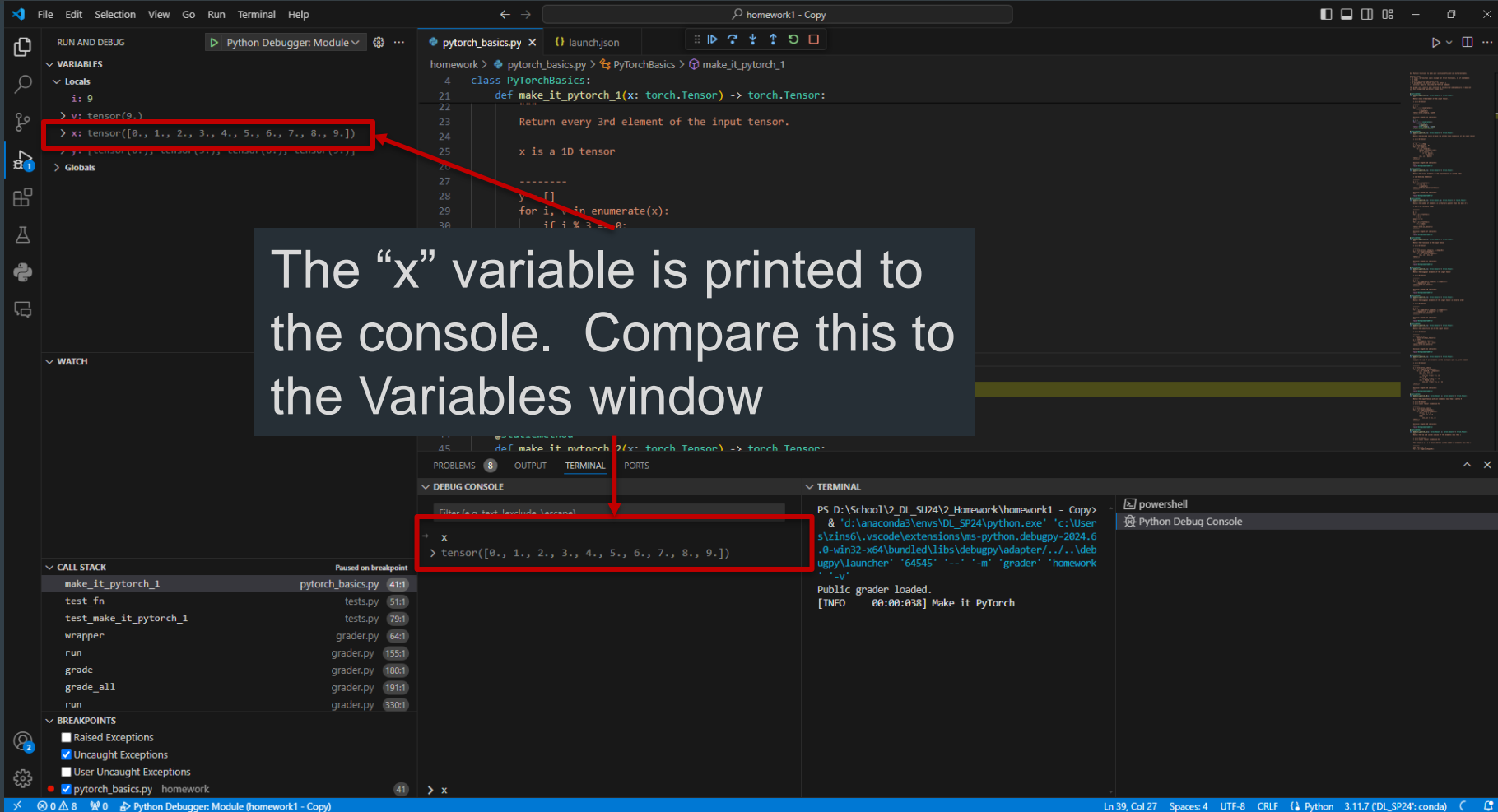
The status bar at the bottom indicates the current file is `pytorch_basics.py` in the `homework1 - Copy` workspace, with the following settings: `Ln 41, Col 1`, `Spaces: 4`, `UTF-8`, `CRLF`, `Python`, `3.11.7 (DL_SP24: conda)`.



VS Code interface showing a Python file named `pytorch_basics.py` with a class `PyTorchBasics` and a method `make_it_pytorch_1`. The method takes a `torch.Tensor` `x` and returns a `torch.Tensor`. The code is currently paused at a breakpoint. A large text overlay in the center of the screen reads: "Type "x" into the Debug Console Prompt press enter".



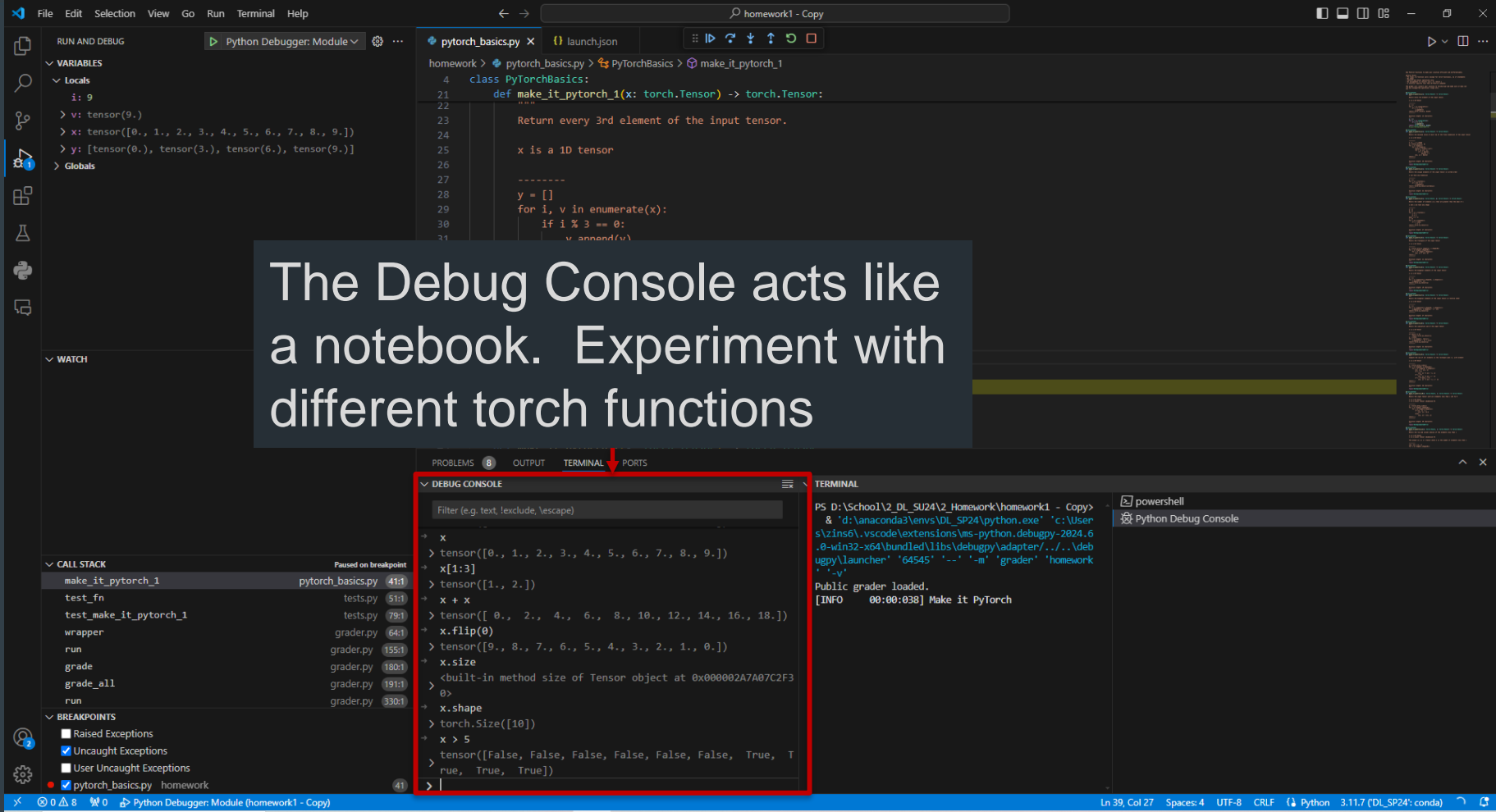
VS Code interface showing the Debug Console and Terminal. The Debug Console is active and shows a prompt `>`. A red arrow points from the text overlay to the prompt. The Terminal shows the command `python -m 'grader' 'homework1' 'x'` being executed.



The screenshot displays the VS Code Python Debugger interface. The **VARIABLES** window on the left shows the local variable `x` with the value `tensor([0., 1., 2., 3., 4., 5., 6., 7., 8., 9.])`, which is highlighted with a red box. A red arrow points from this box to the **DEBUG CONSOLE** at the bottom, where the same tensor value is printed to the console, also highlighted with a red box. The **CALL STACK** on the left shows the execution flow from `make_it_pytorch_1` in `pytorch_basics.py` at line 41, through `test_fn`, `test_make_it_pytorch_1`, `wrapper`, `run`, `grade`, `grade_all`, and finally `run` in `grader.py`. The **TERMINAL** on the right shows the command prompt output, including the command to run the debugger and the message `[INFO 00:00:038] Make it PyTorch`.

The “x” variable is printed to the console. Compare this to the Variables window

The Debug Console acts like a notebook. Experiment with different torch functions



The screenshot displays the Visual Studio Code Python Debugger interface. The main editor shows a Python file named `pytorch_basics.py` with a `make_it_pytorch_1` function. The left sidebar contains the 'VARIABLES' pane with 'Locals' and 'Globals' sections, and the 'WATCH' pane. The bottom panel is divided into 'PROBLEMS', 'OUTPUT', 'TERMINAL', and 'PORTS'. The 'DEBUG CONSOLE' is active, showing a list of breakpoints and a detailed view of the current breakpoint at line 41 of `pytorch_basics.py`. The 'TERMINAL' pane shows the command prompt output, including the command to run the script and the output of the `make_it_pytorch_1` function. The status bar at the bottom indicates the current file is `pytorch_basics.py` in the `homework1` workspace, with the Python interpreter set to `Python 3.11.7 (DL_SP24: conda)`.

```
homework1 > pytorch_basics.py > PyTorchBasics > make_it_pytorch_1
4 class PyTorchBasics:
21     def make_it_pytorch_1(x: torch.Tensor) -> torch.Tensor:
22         """
23         Return every 3rd element of the input tensor.
24
25         x is a 1D tensor
26
27         -----
28         y = []
29         for i, v in enumerate(x):
30             if i % 3 == 0:
31                 v.append(v)
```

DEBUG CONSOLE

Filter (e.g. text, lexclude, 'escape')

Paused on breakpoint

Function	File	Line
make_it_pytorch_1	pytorch_basics.py	41:1
test_fn	tests.py	51:1
test_make_it_pytorch_1	tests.py	79:1
wrapper	grader.py	64:1
run	grader.py	155:1
grade	grader.py	180:1
grade_all	grader.py	191:1
run	grader.py	330:1

BREAKPOINTS

- ☐ Raised Exceptions
- ☒ Uncaught Exceptions
- ☐ User Uncaught Exceptions
- ☒ pytorch_basics.py homework1 (41)

TERMINAL

```
PS D:\School\2_DL_SU24\2_Homework\homework1 - Copy>
& 'd:\anaconda3\envs\DL_SP24\python.exe' 'c:\User
s\zins6\.vscode\extensions\ms-python.debugpy-2024.6
.0-win32-x64\bundled\libs\debugpy\adapter\..\deb
ugpy\launcher' '64545' '--' '-m' 'grader' 'homework
1' '-v'
Public grader loaded.
[INFO 00:00:038] Make it PyTorch
```

References:

- <https://code.visualstudio.com/docs/editor/debugging>