

Nama : Andini Wulandari

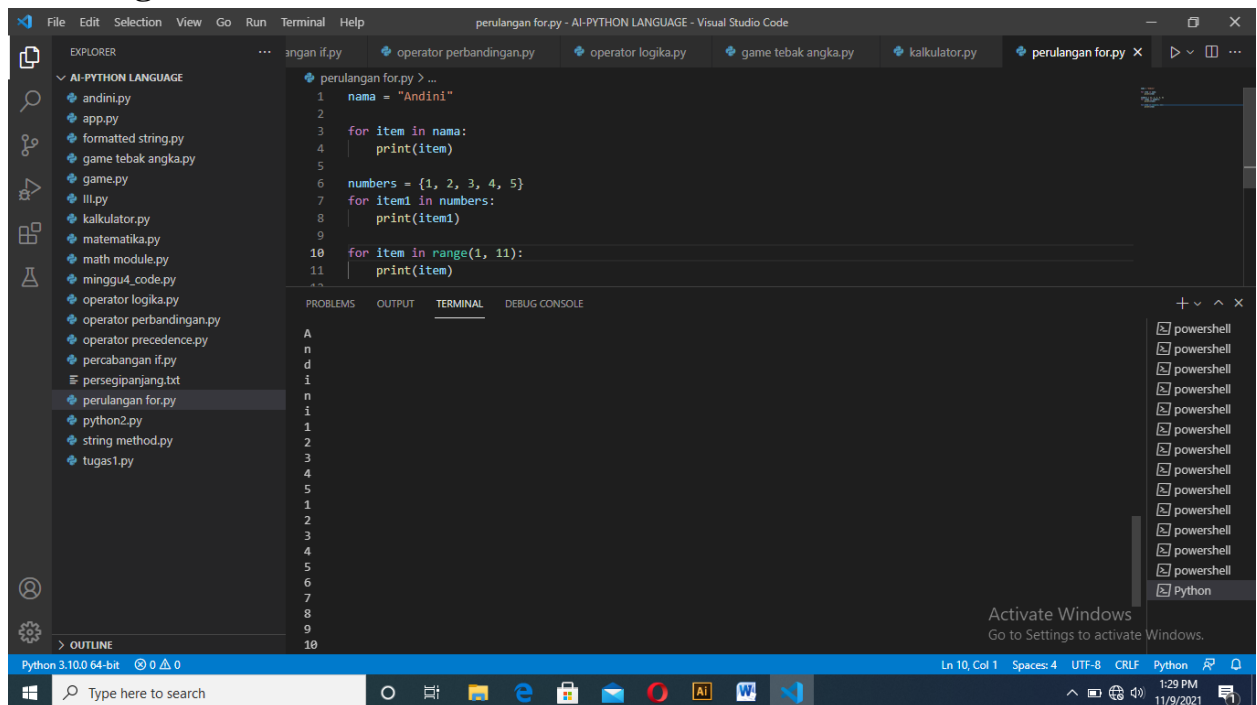
NIM : 20.01.013.020

Kelas : Teknik Informatika B

MK : Kecerdasan Buatan

TUGAS :

1. Perulangan for

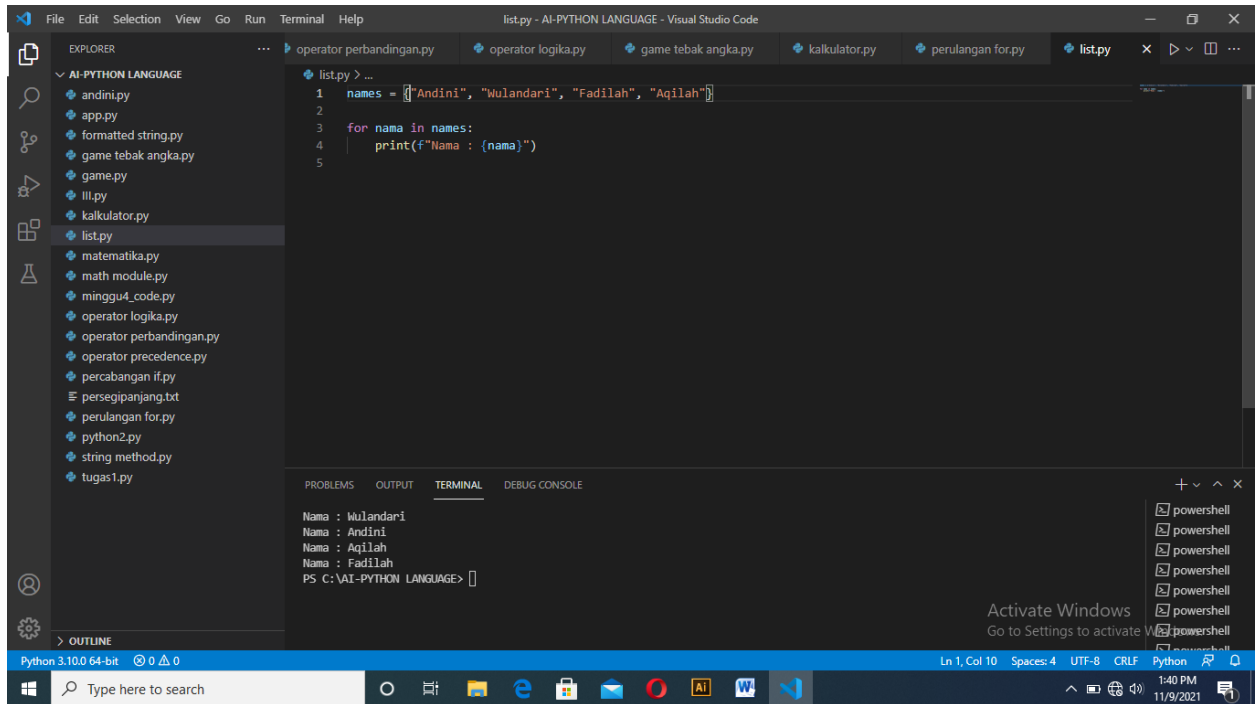


```
perulangan for.py > ...
1  nama = "Andini"
2
3  for item in nama:
4      print(item)
5
6  numbers = {1, 2, 3, 4, 5}
7  for item1 in numbers:
8      print(item1)
9
10 for item in range(1, 11):
11     print(item)
```

Terminal Output:

```
A
n
d
i
n
i
1
2
3
4
5
1
2
3
4
5
6
7
8
9
10
```

2. List



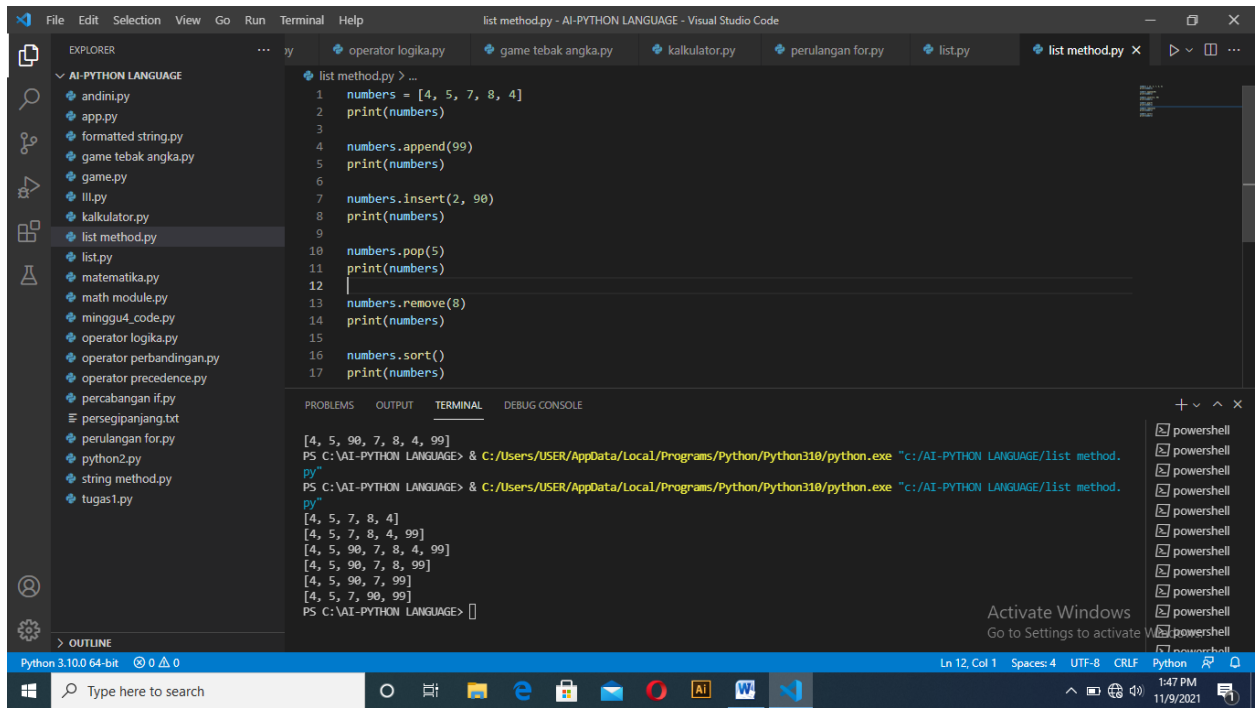
The screenshot shows the Visual Studio Code interface with a Python file named `list.py` open. The code defines a list `names` containing the strings `"Andini"`, `"Wulandari"`, `"Fadilah"`, and `"Aqilah"`. A `for` loop iterates over this list, printing each name with a label. The terminal output shows the execution results: `Nama : Wulandari`, `Nama : Andini`, `Nama : Aqilah`, and `Nama : Fadilah`.

```
list.py > ...
1 names = ["Andini", "Wulandari", "Fadilah", "Aqilah"]
2
3 for nama in names:
4     print(f>Nama : {nama}")
5
```

TERMINAL

```
Nama : Wulandari
Nama : Andini
Nama : Aqilah
Nama : Fadilah
PS C:\AI-PYTHON LANGUAGE>
```

3. List method



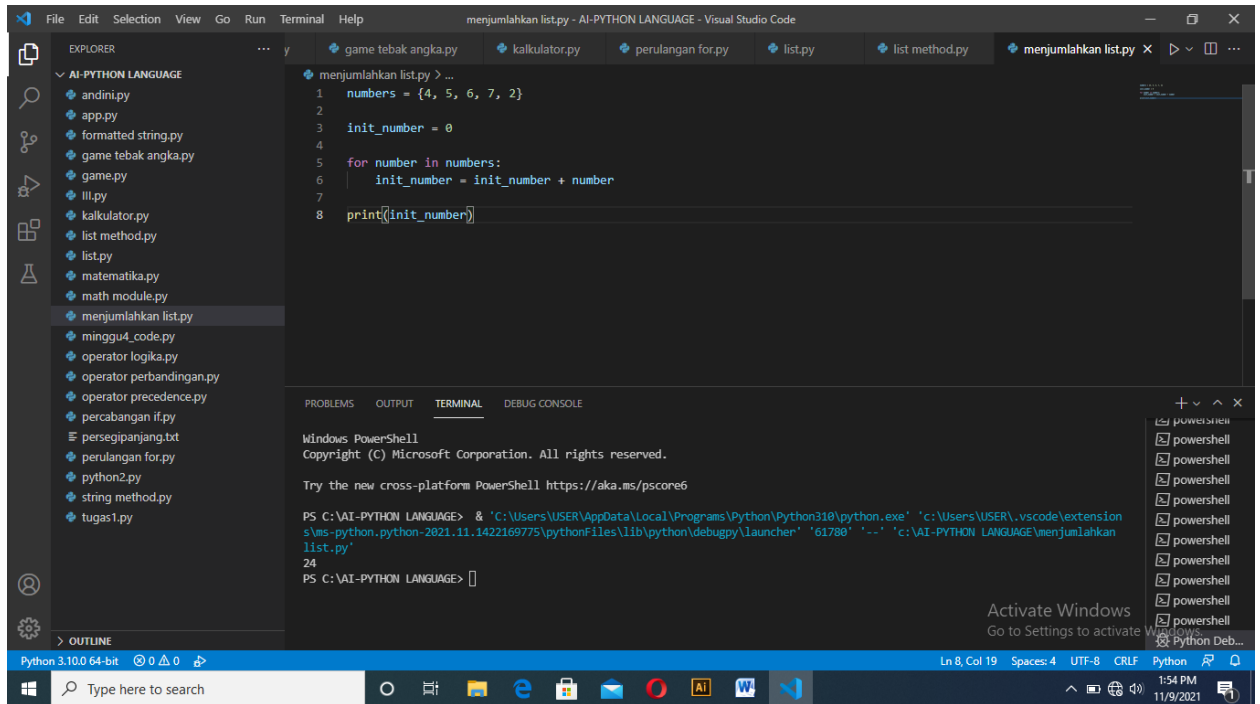
The screenshot shows the Visual Studio Code interface with a Python file named `list method.py` open. The code demonstrates various list methods on a list `numbers` initially containing `[4, 5, 7, 8, 4]`. The methods shown are `append`, `insert`, `pop`, `remove`, and `sort`. The terminal output shows the state of the list after each operation: `[4, 5, 90, 7, 8, 4, 99]` after `append`, `[4, 5, 7, 8, 4]` after `pop`, `[4, 5, 7, 8, 4, 99]` after `insert`, `[4, 5, 90, 7, 8, 4, 99]` after `remove`, and `[4, 5, 7, 90, 99]` after `sort`.

```
list method.py > ...
1 numbers = [4, 5, 7, 8, 4]
2 print(numbers)
3
4 numbers.append(99)
5 print(numbers)
6
7 numbers.insert(2, 90)
8 print(numbers)
9
10 numbers.pop(5)
11 print(numbers)
12
13 numbers.remove(8)
14 print(numbers)
15
16 numbers.sort()
17 print(numbers)
```

TERMINAL

```
[4, 5, 90, 7, 8, 4, 99]
PS C:\AI-PYTHON LANGUAGE> & C:/Users/USER/AppData/Local/Programs/Python/Python310/python.exe "c:/AI-PYTHON LANGUAGE/list method.py"
[4, 5, 7, 8, 4]
[4, 5, 7, 8, 4, 99]
[4, 5, 90, 7, 8, 4, 99]
[4, 5, 90, 7, 99]
PS C:\AI-PYTHON LANGUAGE>
```

4. Menjumlahkan list



The screenshot shows the Visual Studio Code interface with a Python file named `menjumlahkan list.py` open. The code defines a list `numbers = {4, 5, 6, 7, 2}`, initializes `init_number = 0`, and uses a `for` loop to iterate through the list, adding each element to `init_number`. The final result is printed using `print(init_number)`. The terminal window shows the command prompt output, indicating the script was executed successfully. The status bar at the bottom shows the Python version as 3.10.0 64-bit.

```
1 numbers = {4, 5, 6, 7, 2}
2
3 init_number = 0
4
5 for number in numbers:
6     init_number = init_number + number
7
8 print(init_number)
```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

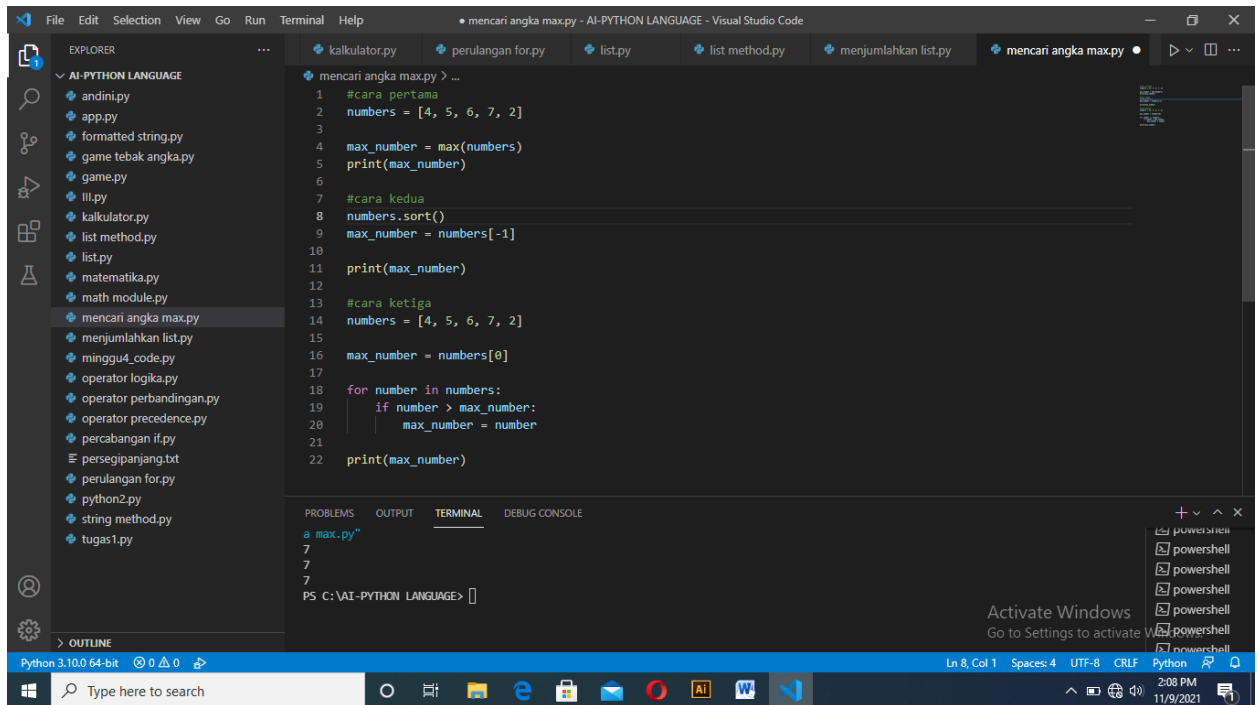
Try the new cross-platform PowerShell <https://aka.ms/pscore6>

PS C:\AI-PYTHON LANGUAGE> & 'C:\Users\USER\AppData\Local\Programs\Python\Python310\python.exe' 'c:\Users\USER\.vscode\extension\sms-python-python-2021.11.1422169775\pythonFiles\lib\python\debugpy\launcher' '61780' '-.' 'c:\AI-PYTHON LANGUAGE\menjumlahkan list.py'

24

PS C:\AI-PYTHON LANGUAGE> █

5. Mencari angka max



The screenshot shows the Visual Studio Code interface with a Python file named `mencari angka max.py` open. The code defines a list `numbers = [4, 5, 6, 7, 2]` and uses three different methods to find the maximum value: `max(numbers)`, `numbers.sort()` followed by `numbers[-1]`, and a `for` loop with an `if` statement to compare each element. The final result is printed using `print(max_number)`. The terminal window shows the command prompt output, indicating the script was executed successfully. The status bar at the bottom shows the Python version as 3.10.0 64-bit.

```
1 #cara pertama
2 numbers = [4, 5, 6, 7, 2]
3
4 max_number = max(numbers)
5 print(max_number)
6
7 #cara kedua
8 numbers.sort()
9 max_number = numbers[-1]
10
11 print(max_number)
12
13 #cara ketiga
14 numbers = [4, 5, 6, 7, 2]
15
16 max_number = numbers[0]
17
18 for number in numbers:
19     if number > max_number:
20         max_number = number
21
22 print(max_number)
```

a max.py"

7

7

7

PS C:\AI-PYTHON LANGUAGE> █

6. Tuple

The screenshot shows the Visual Studio Code interface with a file named `tuple.py` open. The code in the editor is as follows:

```
1 numbers = (4, 5, 6, 7, 2)
2 print(numbers[2])
3
4
5
```

The `PROBLEMS` panel at the bottom displays the following error:

```
File "c:\AI-PYTHON LANGUAGE\tuple.py", line 3, in <module>
  numbers[0] = 10
TypeError: 'tuple' object does not support item assignment
```

The terminal shows the command used to run the script:

```
PS C:\AI-PYTHON LANGUAGE> & C:/Users/USER/AppData/Local/Programs/Python/Python310/python.exe "c:/AI-PYTHON LANGUAGE/tuple.py"
```

The status bar at the bottom indicates the file is at line 5, column 1, with 4 spaces, using UTF-8 encoding and CRLF line endings. The Python version is 3.10.0 64-bit.

7. Unpack

The screenshot shows the Visual Studio Code interface with a file named `unpack.py` open. The code in the editor is as follows:

```
1 numbers = (1, 2, 3)
2
3 # x = numbers[0]
4 # y = numbers[1]
5 # z = numbers[2]
6
7 x, y, z = numbers
8 print(z)
9
10 x, *a = numbers
11 print("")
12 print(x)
13 print(a)
```

The terminal shows the command used to run the script:

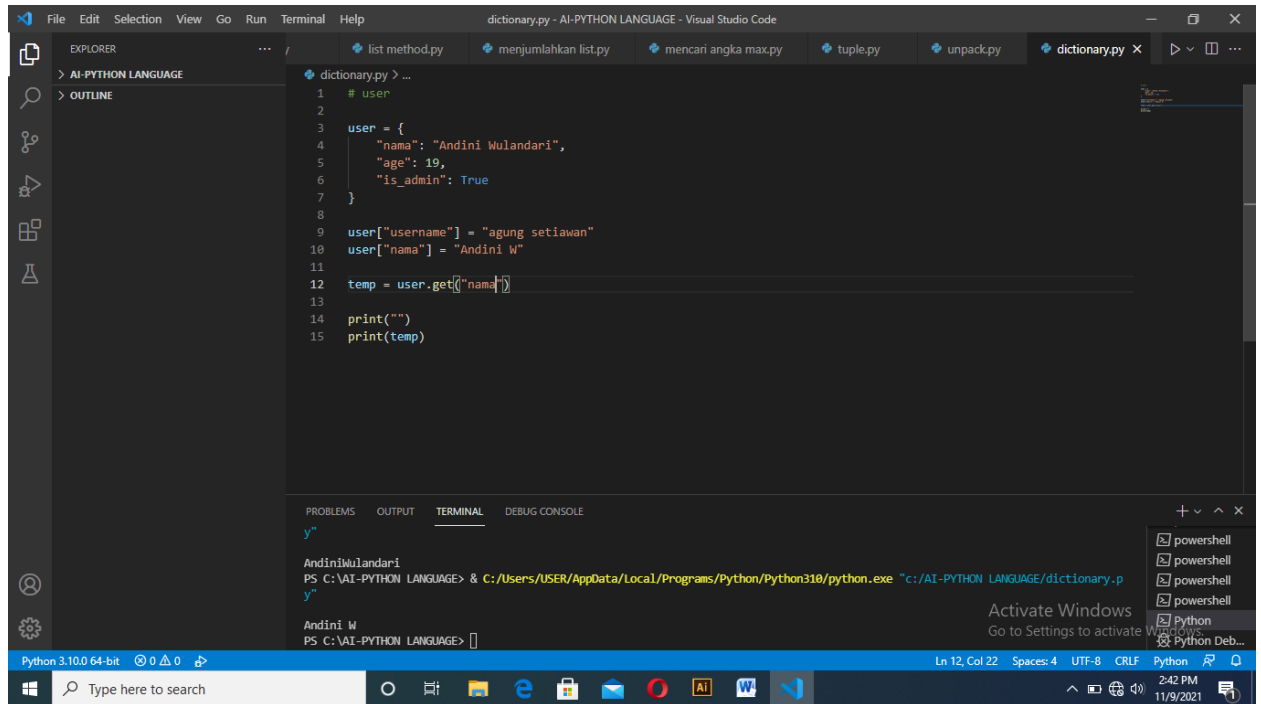
```
PS C:\AI-PYTHON LANGUAGE> & C:/Users/USER/AppData/Local/Programs/Python/Python310/python.exe "c:/AI-PYTHON LANGUAGE/unpack.py"
```

The output of the script is shown in the terminal:

```
3
[2, 3]
```

The status bar at the bottom indicates the file is at line 13, column 9, with 4 spaces, using UTF-8 encoding and CRLF line endings. The Python version is 3.10.0 64-bit.

8. Dictionary



The screenshot shows the Visual Studio Code editor with a file named `dictionary.py` open. The code defines a dictionary `user` with keys `nama`, `age`, and `is_admin`. It then updates the `username` key and uses `user.get()` to retrieve the value for `nama`. The terminal output shows the execution of the script, displaying the dictionary and the retrieved name.

```
1 # user
2
3 user = {
4     "nama": "Andini Wulandari",
5     "age": 19,
6     "is_admin": True
7 }
8
9 user["username"] = "agung setiawan"
10 user["nama"] = "Andini W"
11
12 temp = user.get("nama")
13
14 print("")
15 print(temp)
```

Terminal Output:

```
y"
AndiniWulandari
PS C:\VAI-PYTHON LANGUAGE> & C:/Users/USER/AppData/Local/Programs/Python/Python310/python.exe "c:/VAI-PYTHON LANGUAGE/dictionary.p
y"
Andini W
PS C:\VAI-PYTHON LANGUAGE>
```