

# Pylean



By python

Mohammad amin shahidi



Iran University of  
Science and Technology

## What it does? What is done?

- **Pylean** ( python + clean ) is a program which makes you able to check some Clean Code principles in your python code.
- Assumed principles may be unique and not necessarily from a particular source.
- pylean is python base and uses **Nltk** library for text processing and **Tkinter** for GUI development.
- Don't take wrong , it does not check your syntaxes



Did you know that the standard python IDE is made by Tkinter ?

# Get started

Enter your destination

Shows you the results

Shows you some information about opened file

Generates the results

Clears the pages

The screenshot shows the Pylean application window. At the top, there is a text input field containing the file path 'C:/Users/Microsoft/Desktop/change\_text\_to\_list.py'. To the right of this field are two buttons: 'Analyze' and 'Clear Results'. Below the input field is a 'Tip' box with the text: 'Enter your directory in the entry field , then the results will be shown in lower screen. the lower right field shows you general informations.'

The main area of the application is divided into two panels. The left panel displays the analysis results for the file, which include:

- Functions detected with more than 4 arguments:
- In: Row:3, Column:11, 5 given arguments.
- In: Row:6, Column:19, 6 given arguments.
- Variables and classes detected with meaningless names:
- The variable: given\_list, In row:1, In column:12
- The variable: l\_i\_h\_j\_l\_o\_k\_i, In row:1, In column:30
- The variable: e\_t\_y\_h, In row:3, In column:12

The right panel displays general information about the opened file:

- File opened correctly.
- Number of opened file lines: 19
- size:540 Bytes
- Number of functions having more than 4 arguments: 2
- Number of variables and classes with meaningless names: 19
- Number of functions with meaningless names: 2
- Number of wrong indexes: 7

## What it shows to user?

- **Functions** with more than 4 arguments.
- **Variables** and **classes** with meaningless names.
- **Functions** with non-verb names.
- **Indexes** which are characters and does not belong to a specific for block.

## How it works?

- It's a function base program.
- Each function does a specific thing.
- A result can be a combination of several functions return values.
- Functions must have a specific design to be used in a Tkinter based program.



## How it works?

Our work  
zone

```
1  from tkinter import *
2  from tkinter import messagebox
3  from tkinter import ttk
4
5  class manager:
6      def __init__(self, master):
7          #GUI Elements
8          #Functions
9
10 def main():
11     root = Tk()
12     root.geometry('600x700')
13     app = manager(root)
14     root.mainloop()
15
16 if __name__ == "__main__": main()
17
```

## GUI elements

```
class manager:

    def __init__(self, master):

        self.show = Text(master , width=69 , height=27)
        self.show.place(x=15 , y = 200)
        self.show.config(background='lightblue' , foreground='darkblue' ,borderwidth=4)
        self.show.config(font=('italic'))

self.b1=ttk.Button(self.frame, text = "    Analyze    ",command = self.action).place(x=600 , y=5)
self.b2=ttk.Button(self.frame, text = " Clear Results ",command = self.clear).place(x=690 , y=5)
```

# Functions

- Why I can't explain all of them?!
- There are 16 different functions that arguments of most of them are return value/s of other one/s.
- Some functions are created because of Tkinter platform rules.
- So if I decide to explain one function I must explain some others which takes time.
- Next , I explain most important ones.



# Functions

## Most important one

- The function which recognize words and give us information about them (tokenizer).
- Information: Row , Column , Word itself and Type .
- Doesn't result tokens as what we know better “ **class instance**” .
- Puts information in **lists** (4 ones)

# Functions

## Most important one result

Words	Apple	for	count	C1	...	X
Rows	1	1	2	5	...	Y
Columns	1	25	4	7	...	Z
Types	variable	keyword	function	class	...	W
	Token 1	Token 2	Token 3	Token 4		Token n

## Functions

### One that detects for block

- To realize if a index belongs to a for block , first , we need to know the **position** (first row and last row) of are for blocks ( actually we don't have any blocks in python and it's better to call it "**for zone**").

We detect a  
for

We detect  
the index

Check for every line by  
it's first word:  
Is it placed in a column  
which makes it belong  
to the for block?

We detect it's  
row and  
column  
Row = 1  
Column = 1

```
for i in range ( 0 , 100 ) :
```

```
count = count + 1
```

```
print ( "THE COUNT : " + count )
```

```
print ( "THE FOR INDEX : " + i )
```

```
def TEST_1 ( VALUE ) :
```

```
temp = VALUE * VALUE
```

```
return temp
```

It doesn't  
belong to the  
for block.  
So it's whole  
line doesn't

now we know  
which lines  
belongs to  
which for

[ i , 1 , 4 ]

we save this as a "for token"

## For indexes

- Now we know which lines every **for** covers
- We can check for indexes:

“Is the line that index is in it , covered by a for with same index?”

- Yes → Ok!
- No → Save it as a wrong Index