1. What is the result of the code, and explain?

>>> X = 'iNeuron'
>>> def func():
print(X)

>>> func()

```
X = 'iNeuron'
def func(): print(X)
func()
```

```
'the  result of code is to printing ineuron, this is because X is global variable so it will be accessed in func method'
```

2. What is the result of the code, and explain?

>>> X = 'iNeuron'
>>> def func():
X = 'NI!'

>>> func()
>>> print(X)

```
X = 'iNeuron'
def func():
    X = 'NI!'
func()
print(X)
```

```
‘This prints iNeuron, new assigned value will not be accessed outside function’
```

3. What does this code print, and why?

```
>>> X = 'iNeuron'
>>> def func():
X = 'NI'
print(X)
```

```
>>> func()
>>> print(X)
```

```
This prints below one, This prints both local value of function and also global value. The value of globally assigned will not be accessed outside function.
```

```
NI
iNeuron
```

4. What output does this code produce? Why?

```
>>> X = 'iNeuron'
>>> def func():
global X
X = 'NI'
```

```
>>> func()
>>> print(X)
```

```
This print NI since the we declared X as global in method and assigned NI to it.
```

5. What about this code—what’s the output, and why?

```
>>> X = 'iNeuron'
>>> def func():
X = 'NI'
def nested():
print(X)
nested()


>>> func()
>>> X
```

```
It prints INueron, since global value cannot be changed inside method
```

6. How about this code: what is its output in Python 3, and explain?

```
>>> def func():
X = 'NI'
def nested():
nonlocal X
X = 'Spam'
nested()
print(X)


>>> func()
```

```
This gives error - SyntaxError: no binding for nonlocal 'X' found
```