? Quizze AD



#### Python OOP

- Python OOP
- Classes and Objects in Python
- Constructors in Python
- Python Destructors
- ⊕ Encapsulation in Python
- Polymorphism in Python
- Inheritance in Python
- Python Instance Variables
- Python Instance Methods
- Python Class Variables
- Python Class Method
- Python Static Method
- Python Class Method vs. Static Method vs. Instance Method
- Python OOP exercise

#### **All Python Topics**

Python Basics Python Exercises **Python Quizzes** 

**Python File Handling** 

Python OOP

Python »

**Python** 

Object-

Oriented

**Programming** 

(OOP) »

**Python Class** 

Variables

# **Pyth** Class **Varia** bles

Updated

on: February

7, 2022 | + 5

Comments

In Object-

oriented

<u>programmi</u>

ng, when

we design a

class, we

use

instance

variables



Learn Python

Exercises

? Quizze AD

In Class,

attributes

can be

defined into

two parts:

• Instan

ce

<u>variab</u>

<u>les</u>: If

the

value

of a

variabl

е

varies

from

object

to

object,

then

such

variabl

es are

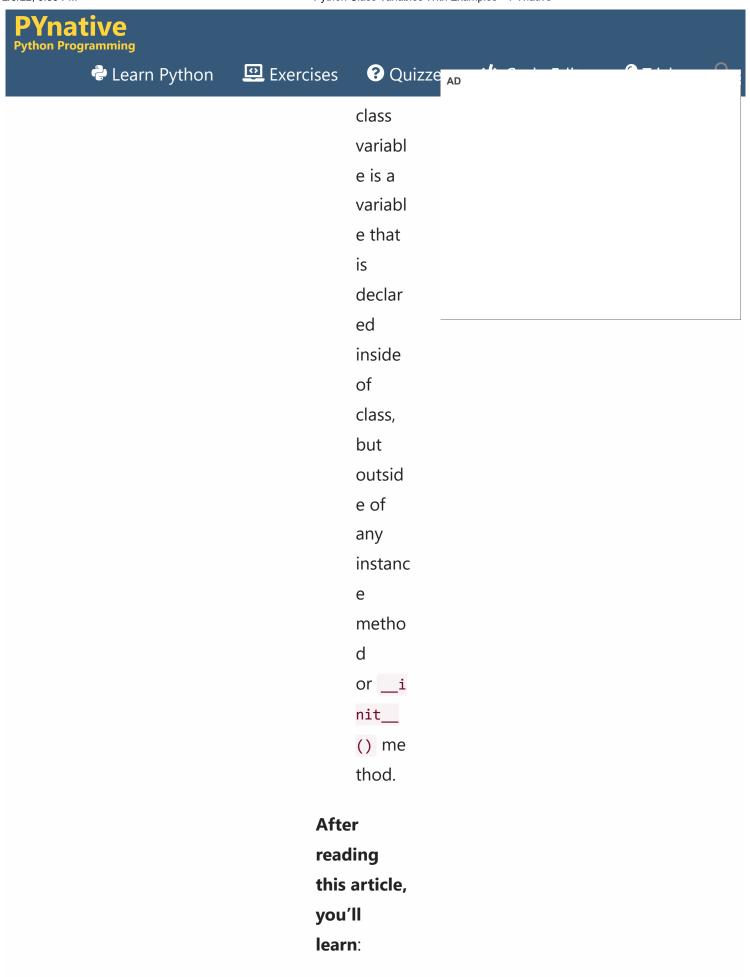
called

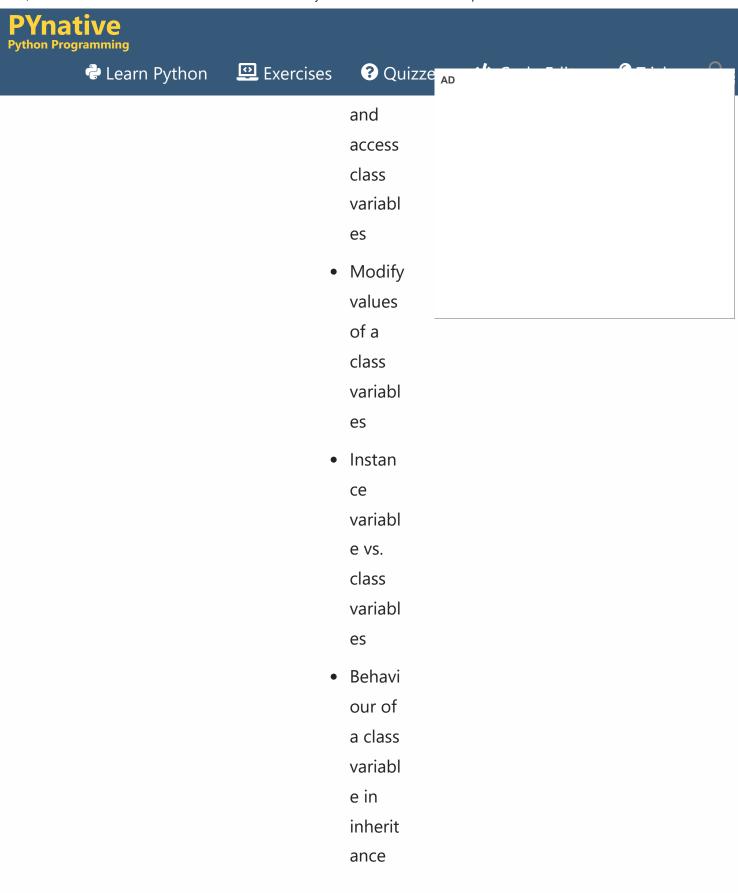
instanc

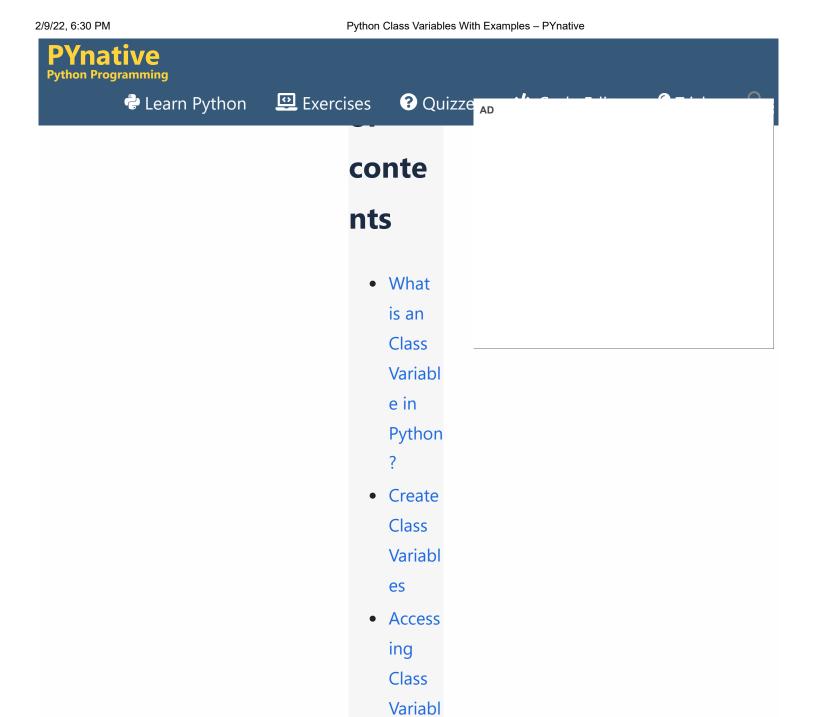
е

variabl

es.







Variabl

es

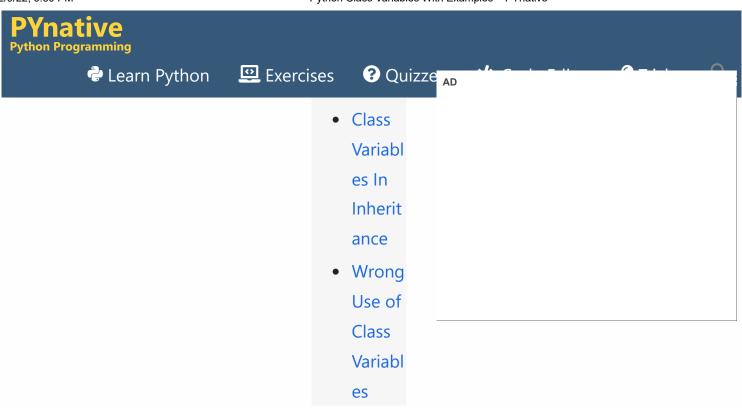
es

Class

Variabl

e vs

Instan



What is an Class Variab le in Pytho n?

of a
variable is
not varied

from





Exercises



of variables

are called

class

variables or

static

variables.

Class

variables

are **shared** 

by all

instances

of a class.

Unlike

instance

variable, the

value of a

class

variable is

not varied

from object

to object,

In Python,

Class

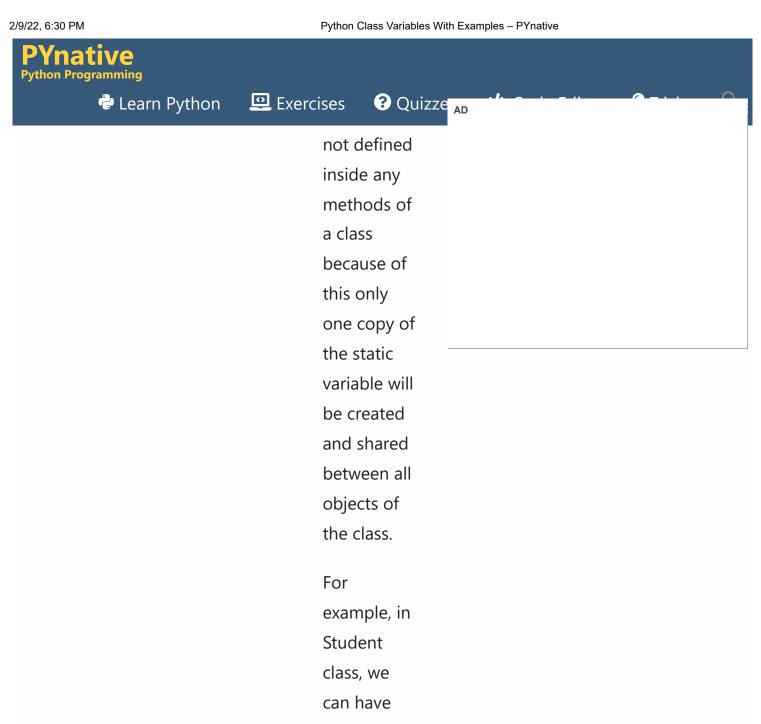
variables

are

declared

when a

class is



different

instance

variables

such as

name and

roll number

because

each

student's

name and



**Exercises** 



But, if we

want to

include the

school

name in the

student

class, we

must use

the class

variable

instead of

an instance

variable as

the school

name is the

same for all

students. So

instead of

maintaining

the

separate

copy in

each object,

we can

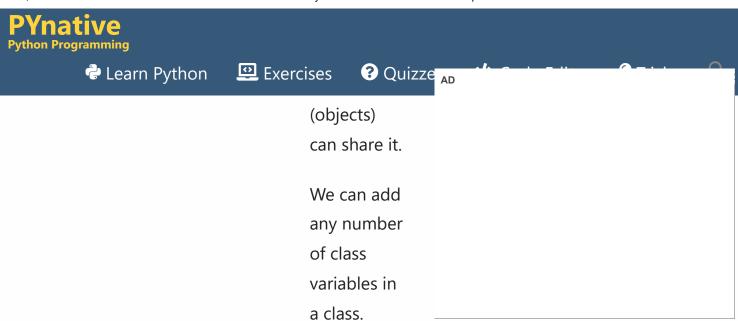
create a

class

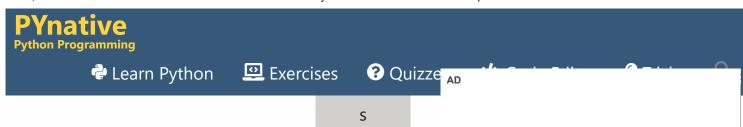
variable

that will

hold the



••• U n d е r S t а n d a S S ٧ a ri a



## **Create**

# **Class**

# **Variab**

# les

A class variable is declared inside of class, but outside of any instance method or \_\_init\_ \_() metho d. Ву convention, typically it is placed right

below the



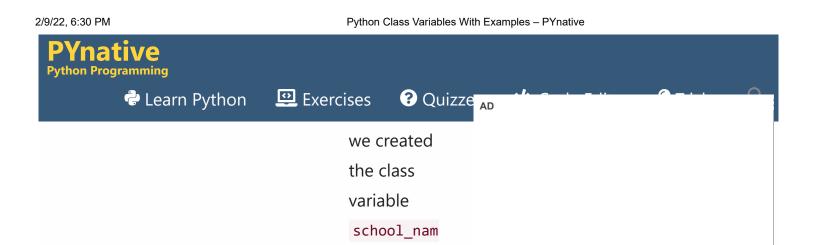
constructor method and other methods.

#### Example:

```
# creating state of the state o
```

#### Output

Emma 10 A Jessa 20



Note: Like

regular

e and

accessed it

using the

object and

class name.

variables,

class

variables

can store

data of any

type. We

can use

Python list,

**Python** 

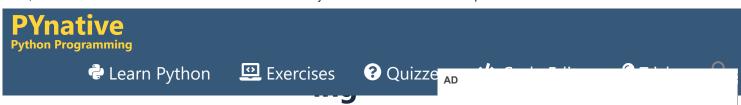
tuple, and

**Python** 

<u>dictionary</u>

as a class

variable.



# **Class**

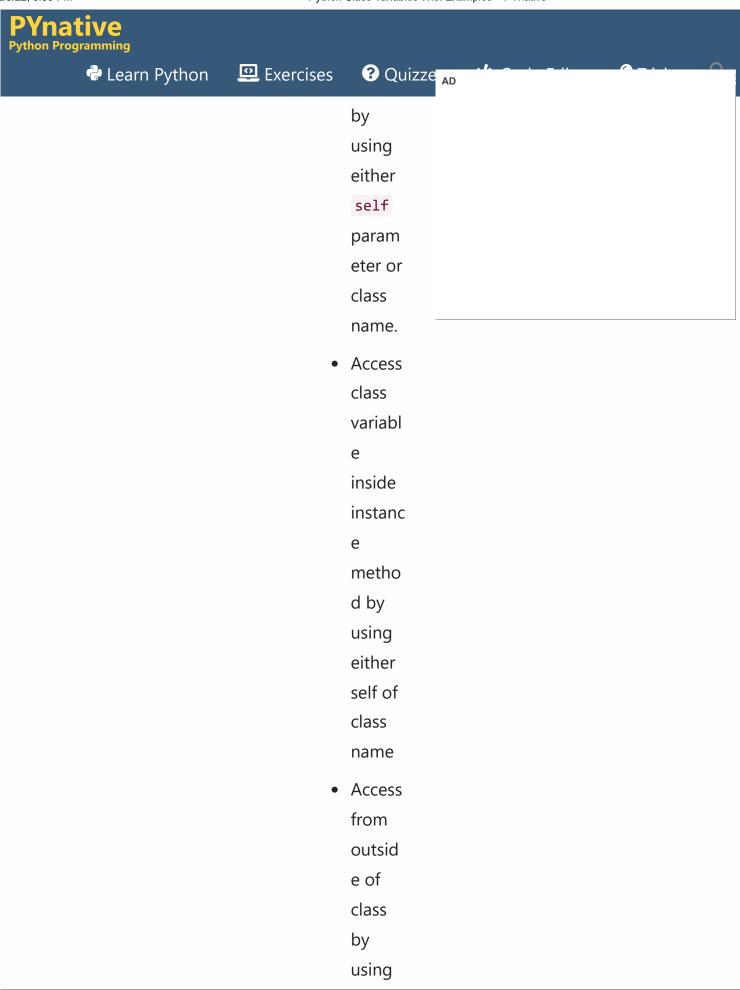
# **Variab**

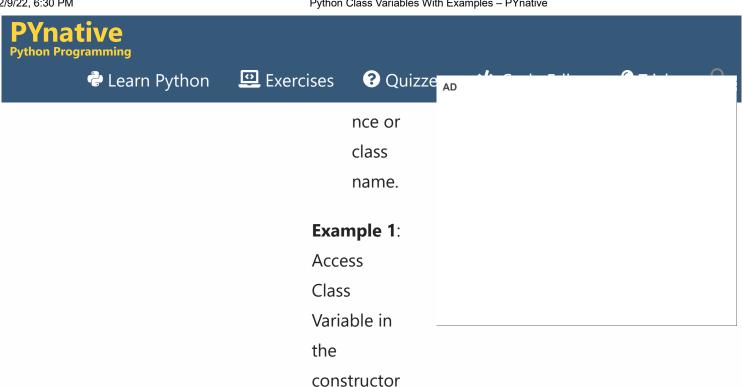
## les

We can access static variables either by class name or by object reference, but it is recommend ed to use the class name.

In Python, we can access the class variable in the following places

 Access inside



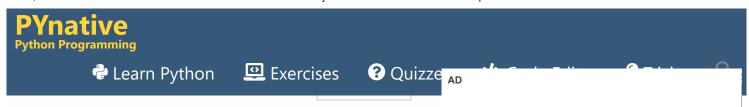


class S<sup>.</sup> sch def s1 = St **○** Run

#### Output

ABC Schoo ABC Schoo





#### **Output**

Inside in
Emma 10 A
ABC Schoo
Outside c
ABC Schoo
ABC Schoo

In this example,

we

accessed

the class

variable

school\_nam

e using

class name

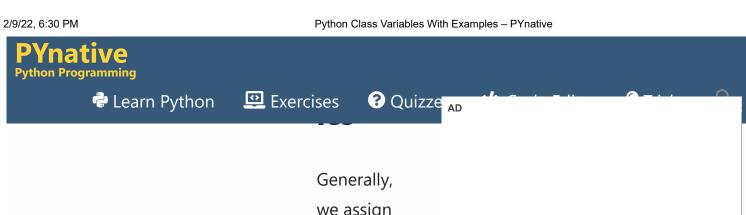
and a self

keyword

inside a

method.

# Modif y Class



we assign

value to a

class

variable

inside the

class

declaration.

However,

we can

change the

value of the

class

variable

either in the

class or

outside of

class.

Note: We

should

change the

class

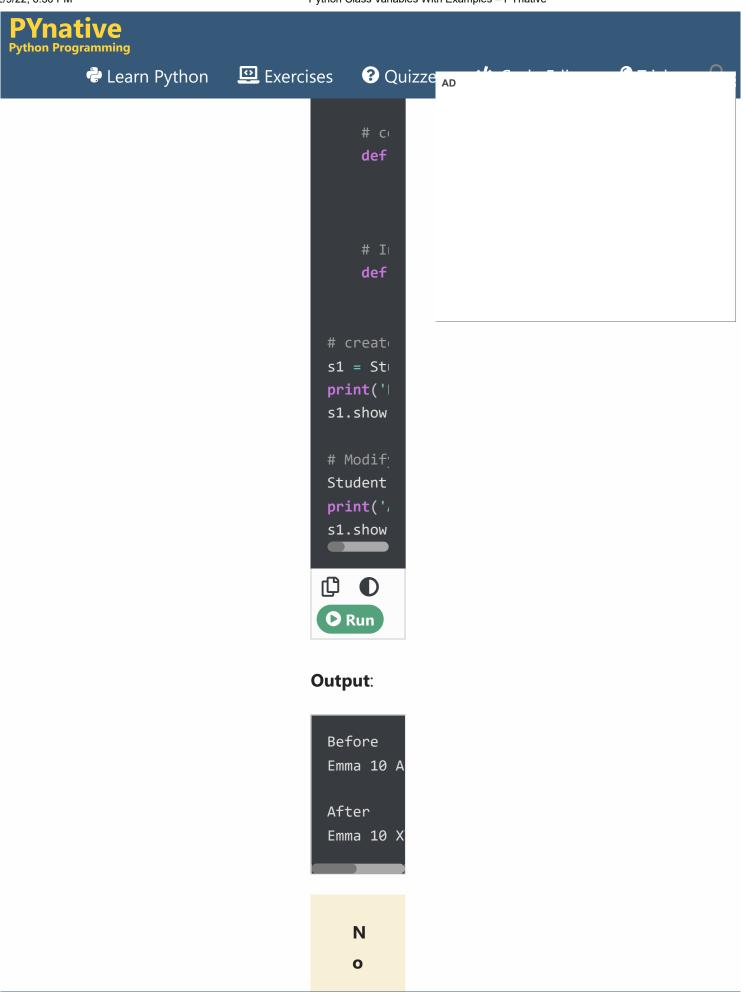
variable's

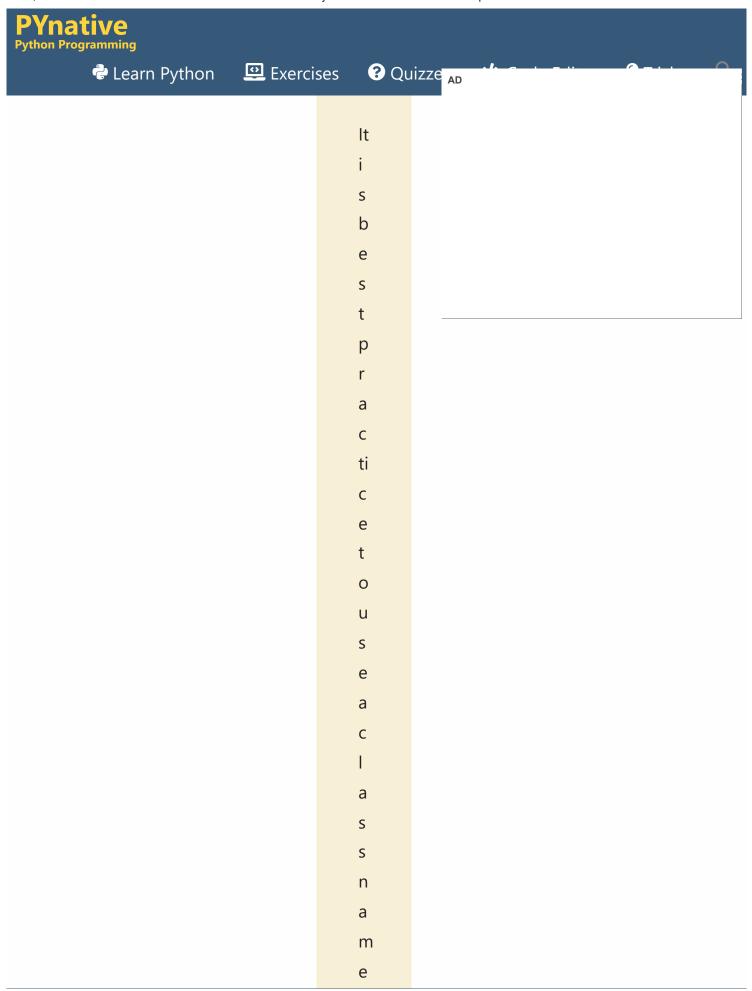
value using

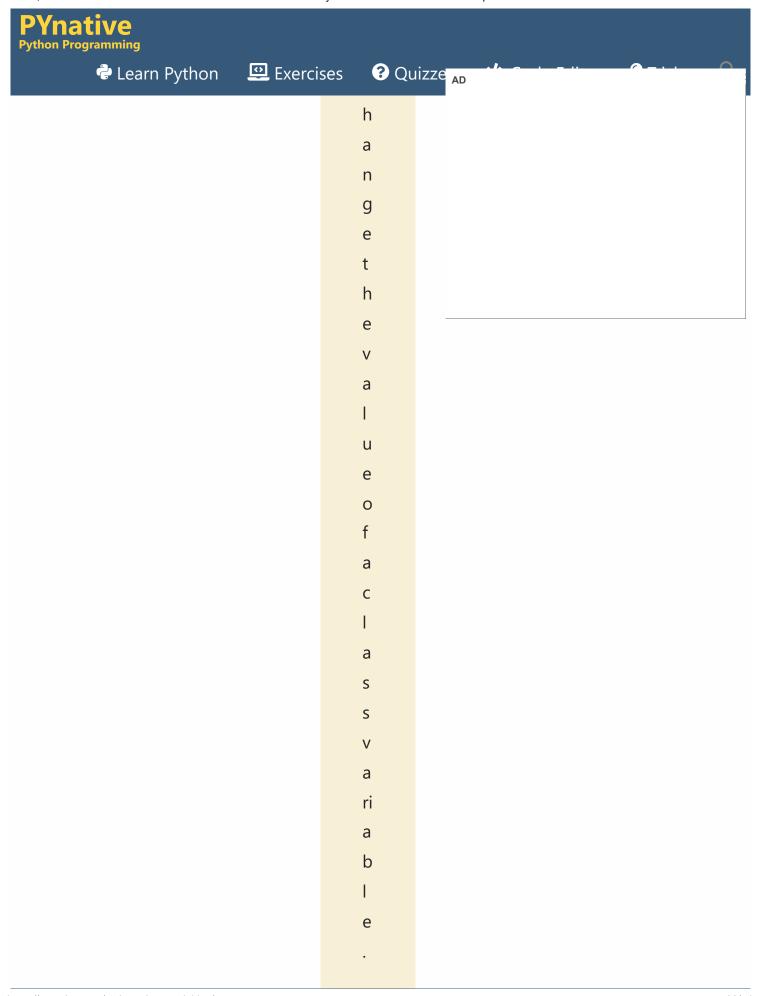
the class

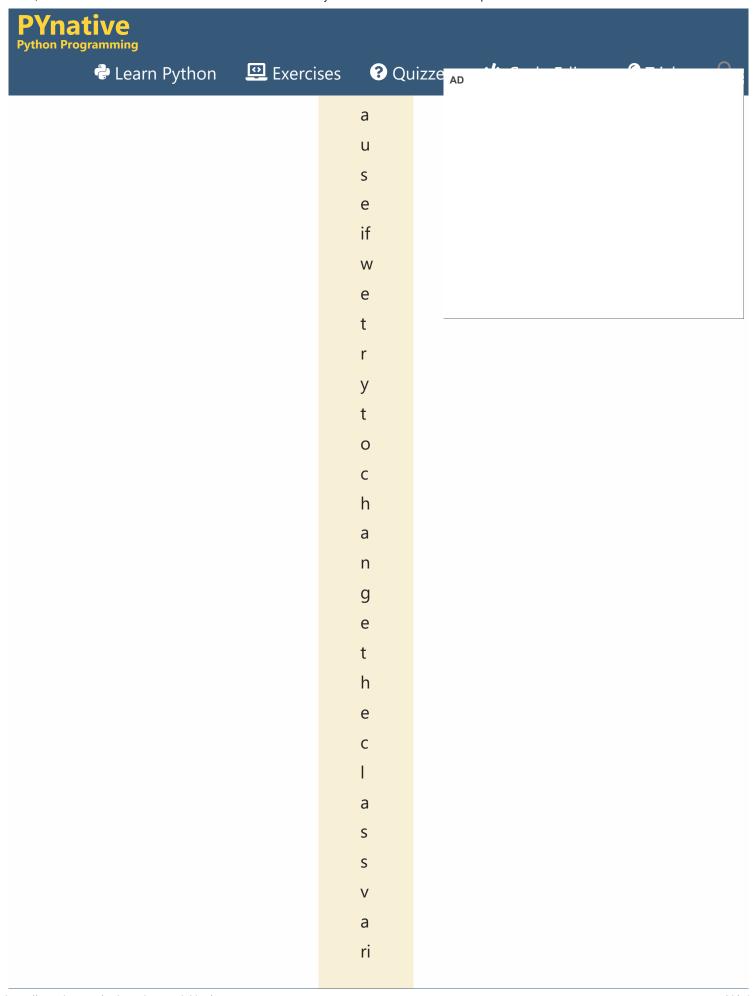
name only.

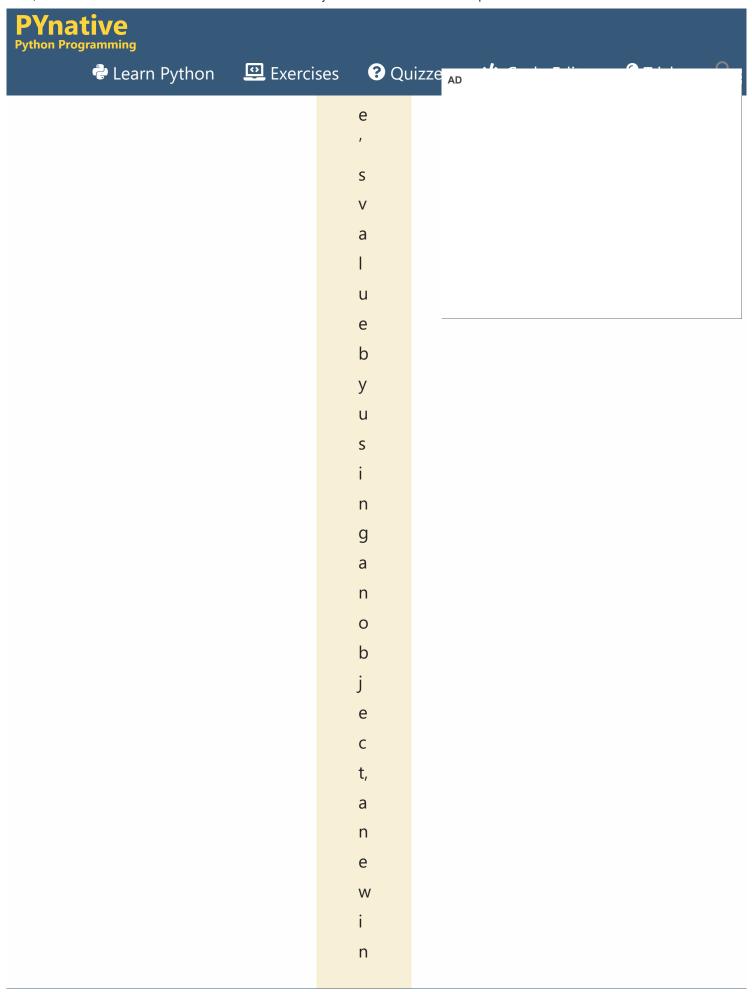
#### **Example**

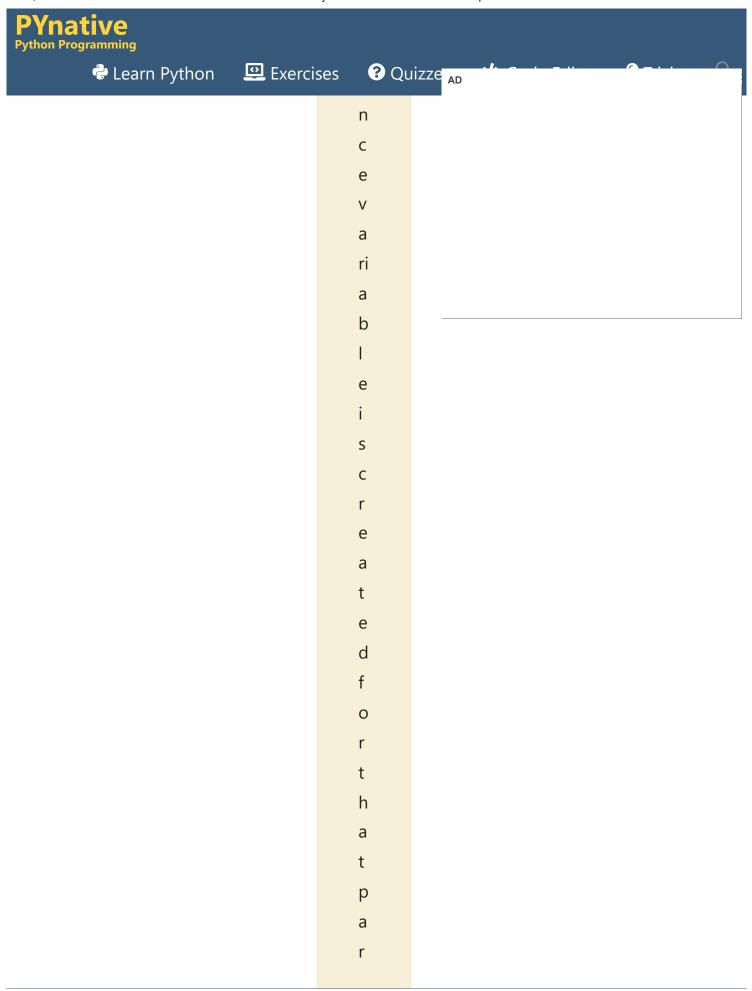


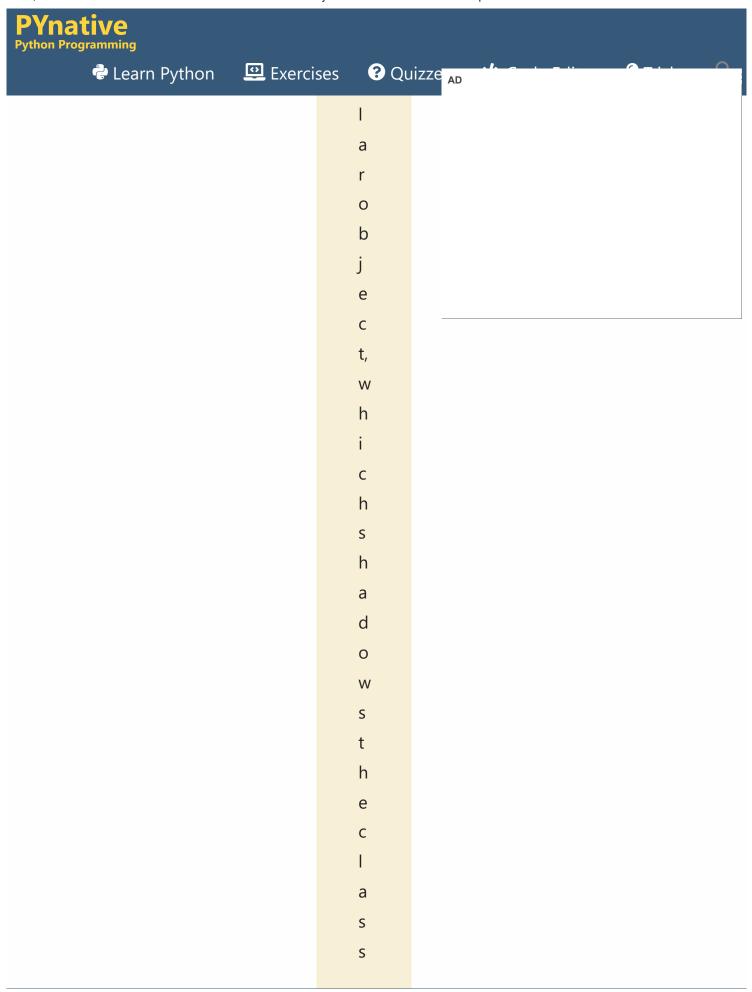


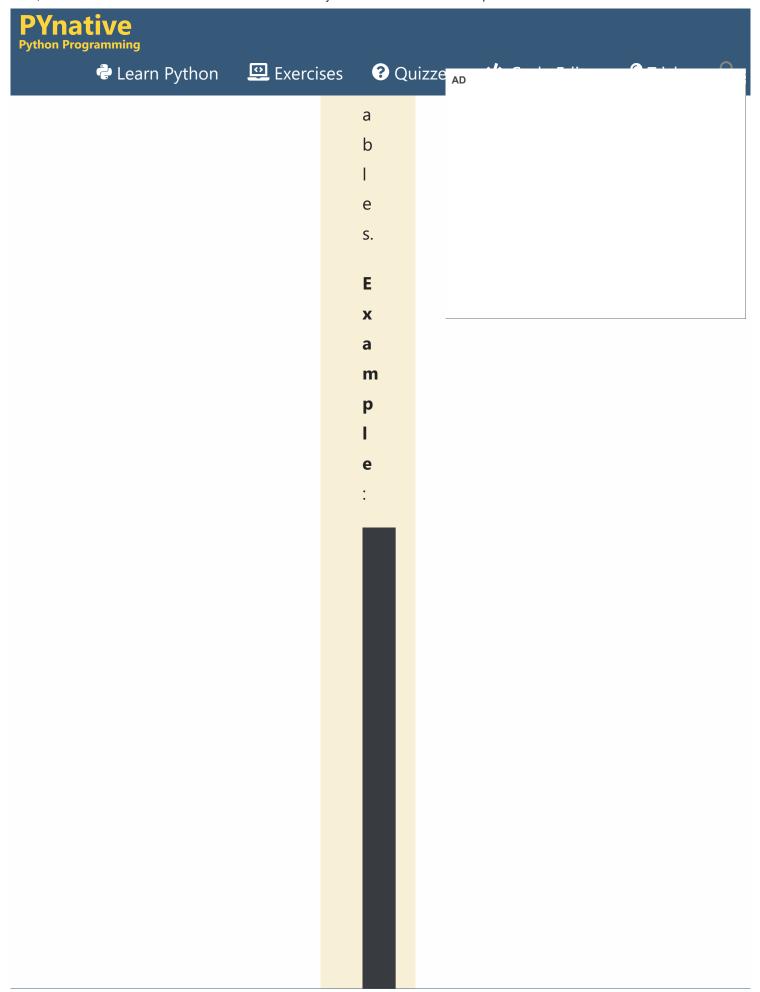


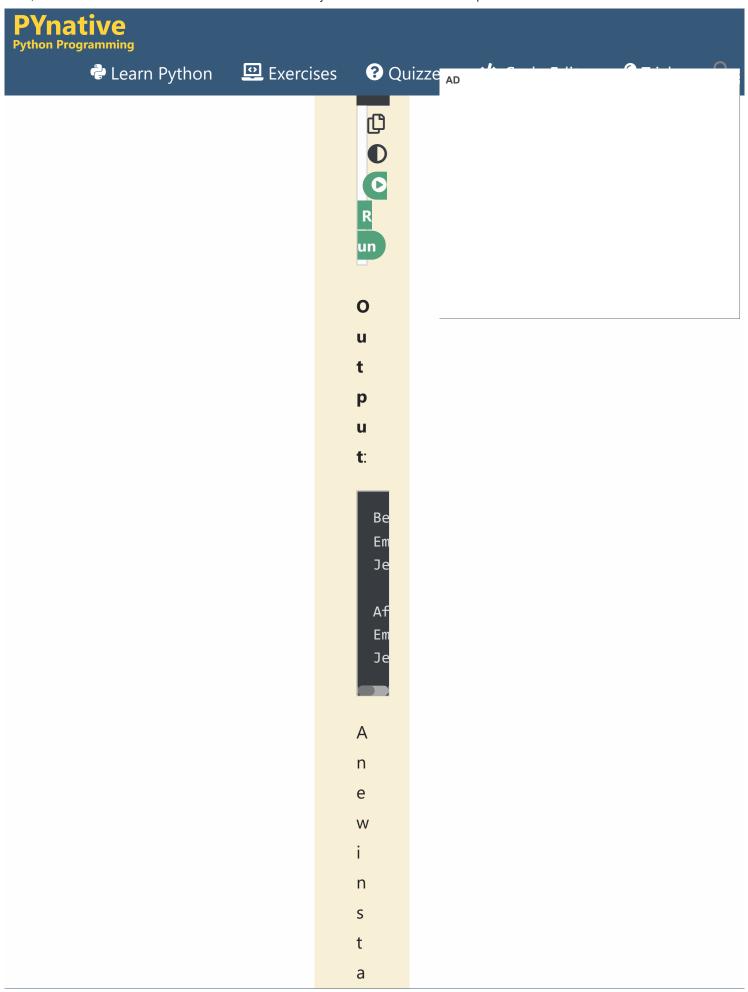


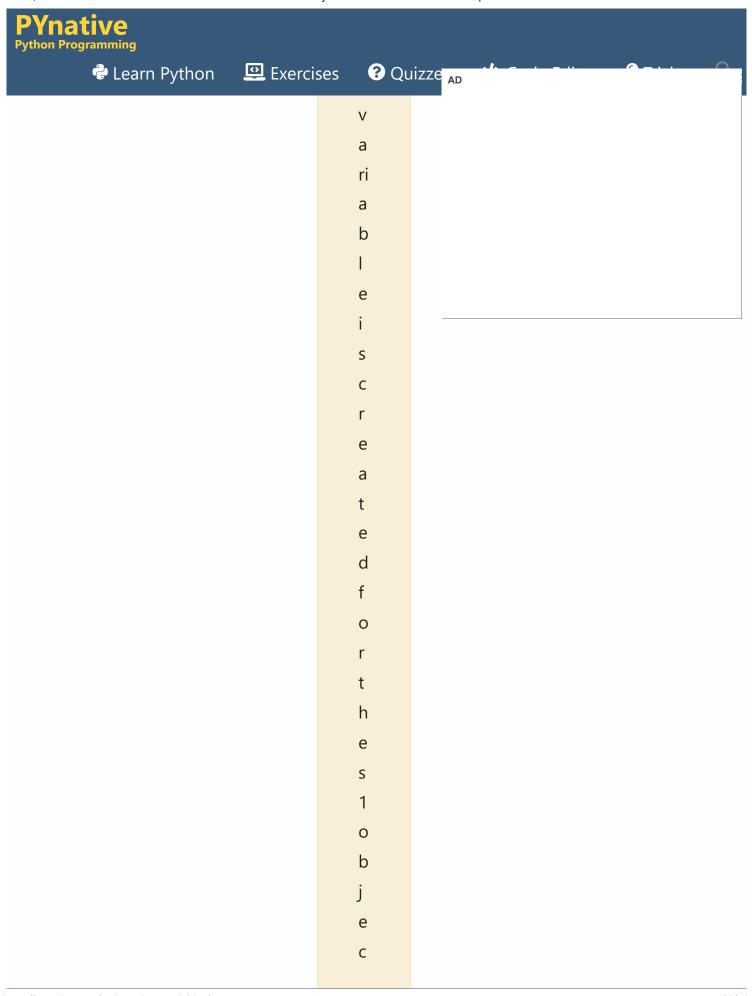


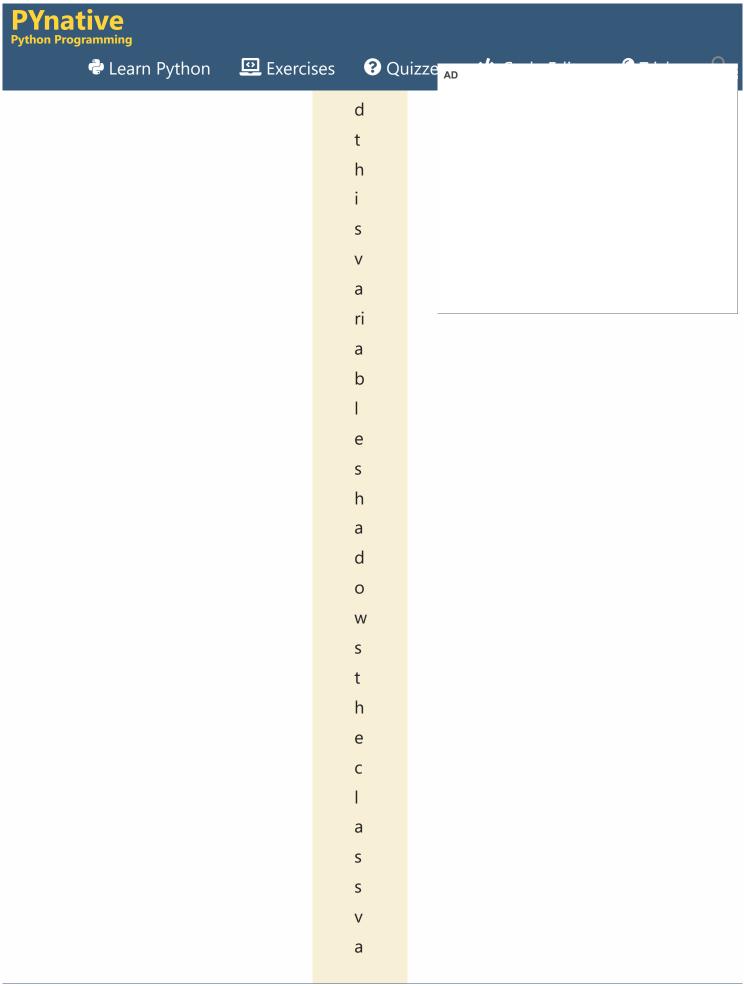


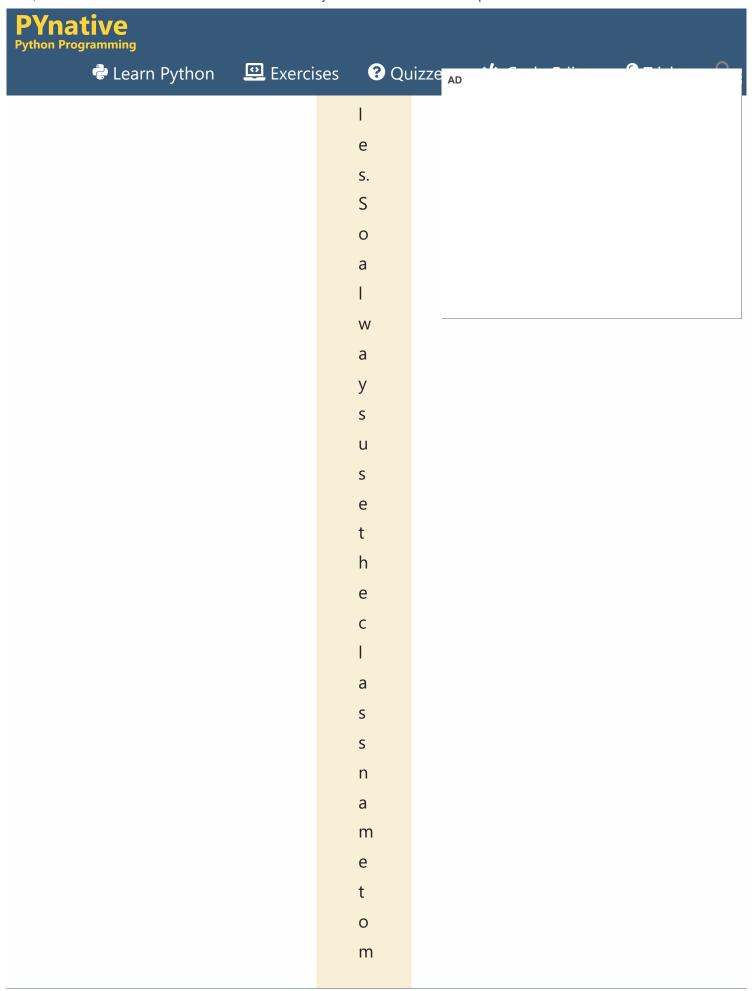


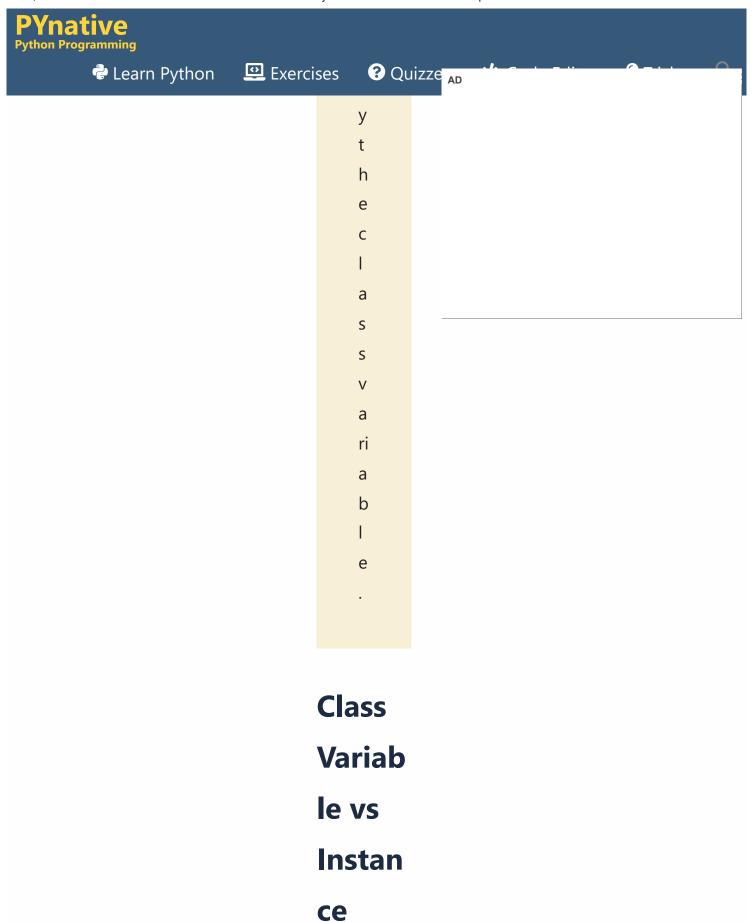


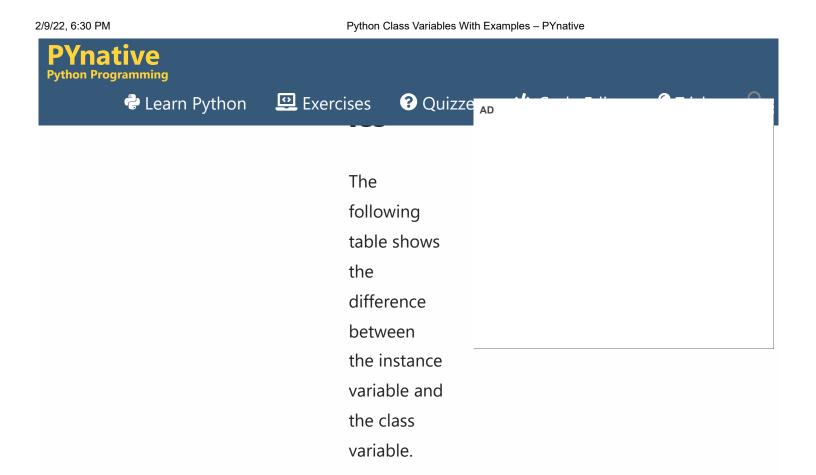












In Python, properties can be defined into

two parts:

Instan

ce

variab

les:

Instan

ce

variabl

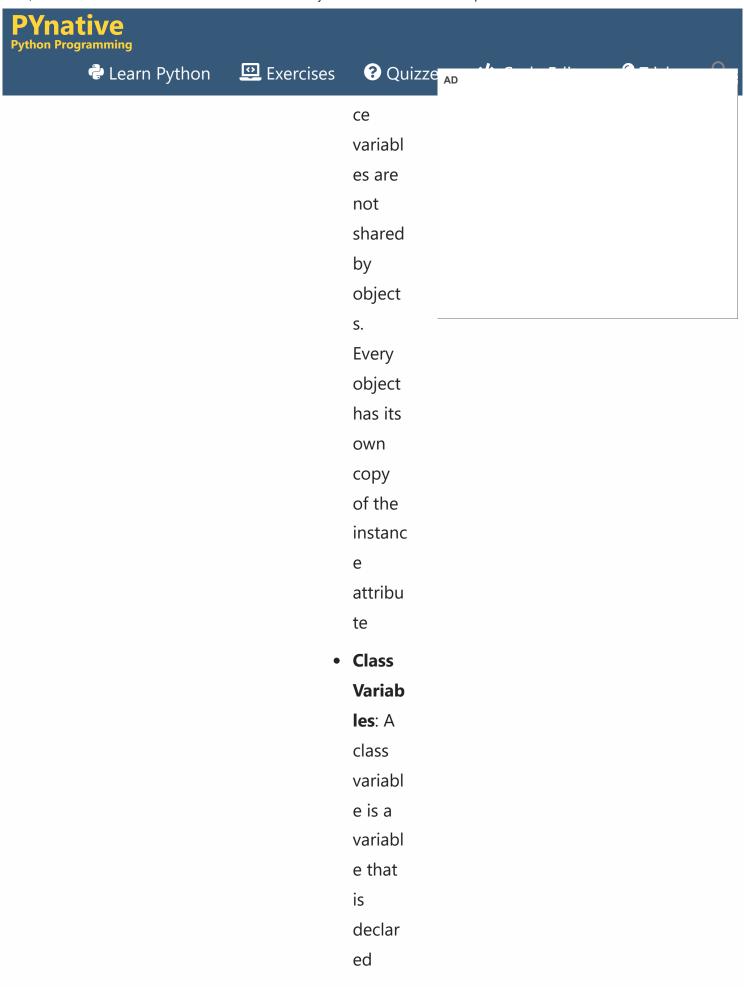
e's

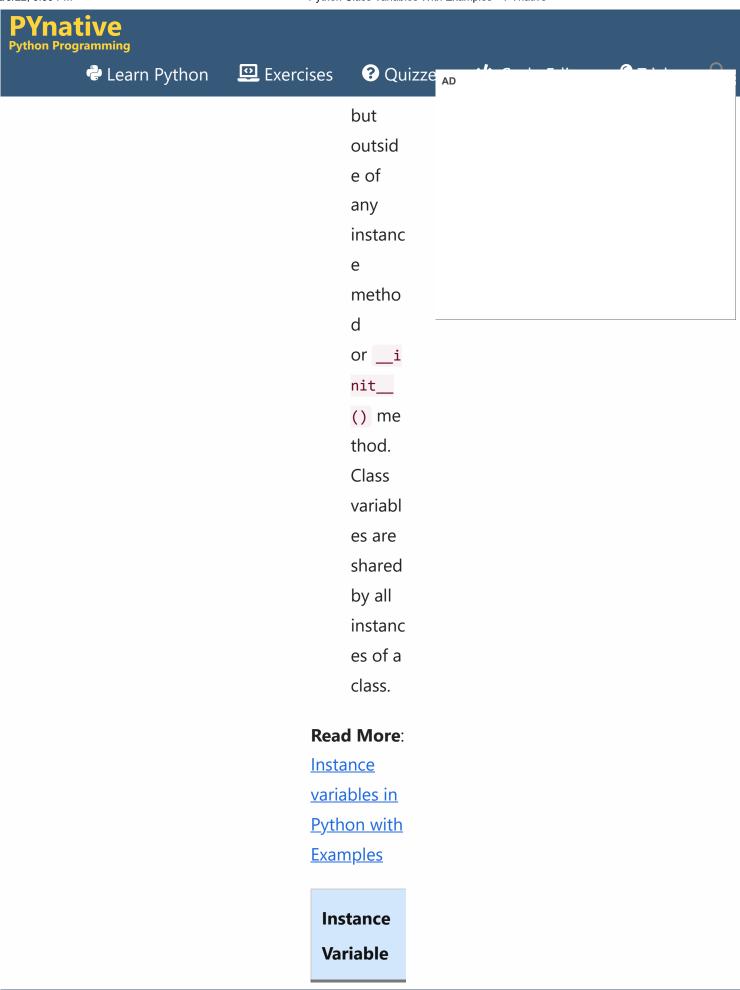
value

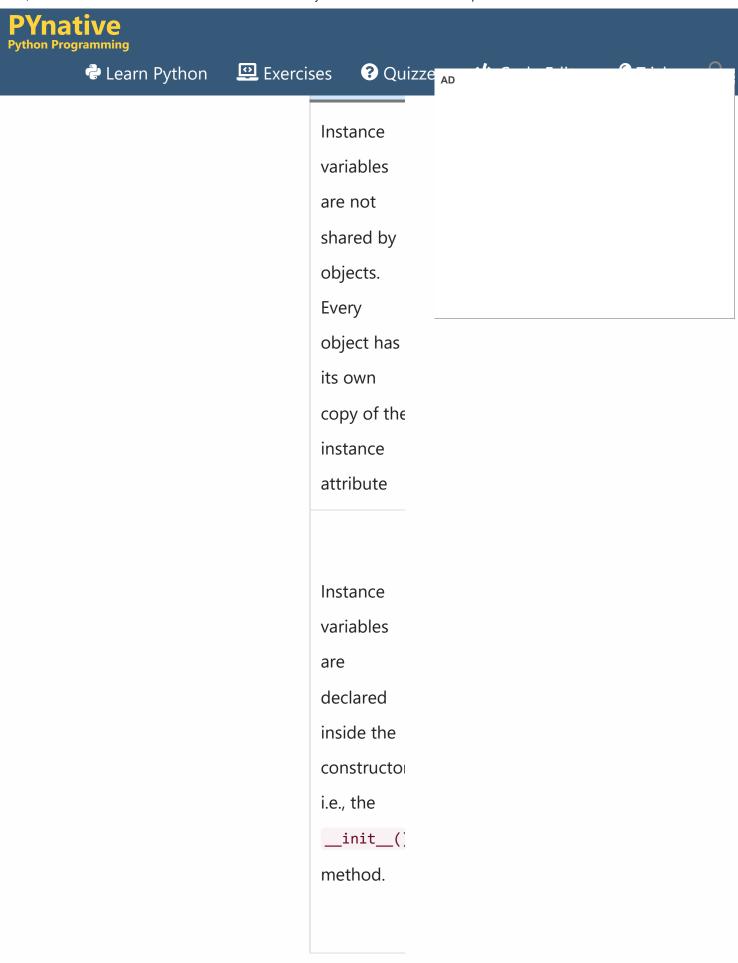
varies

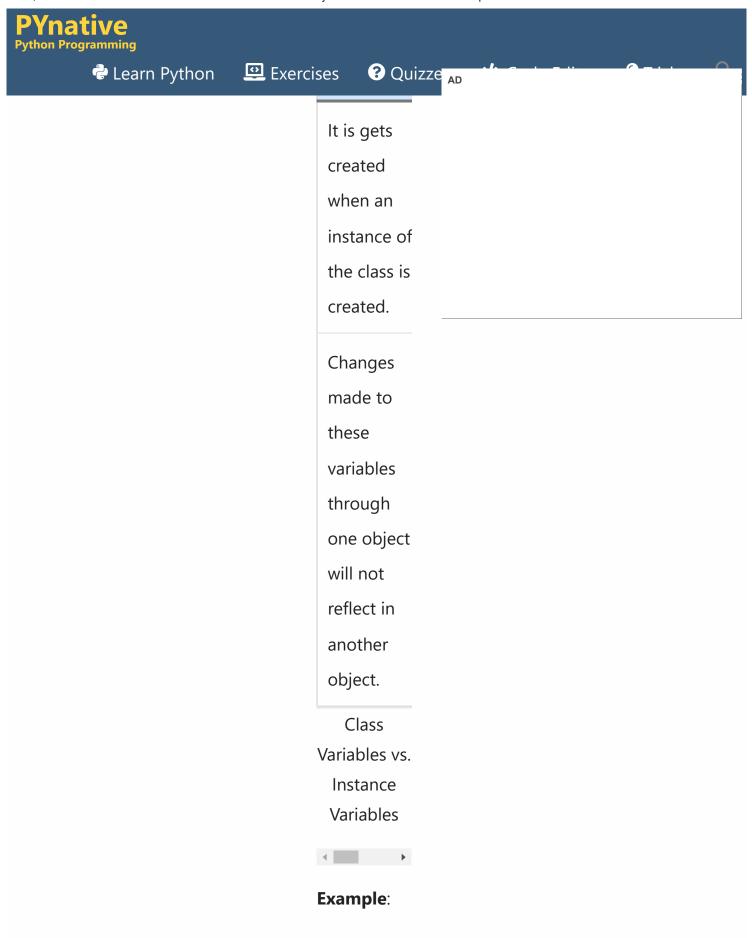
from

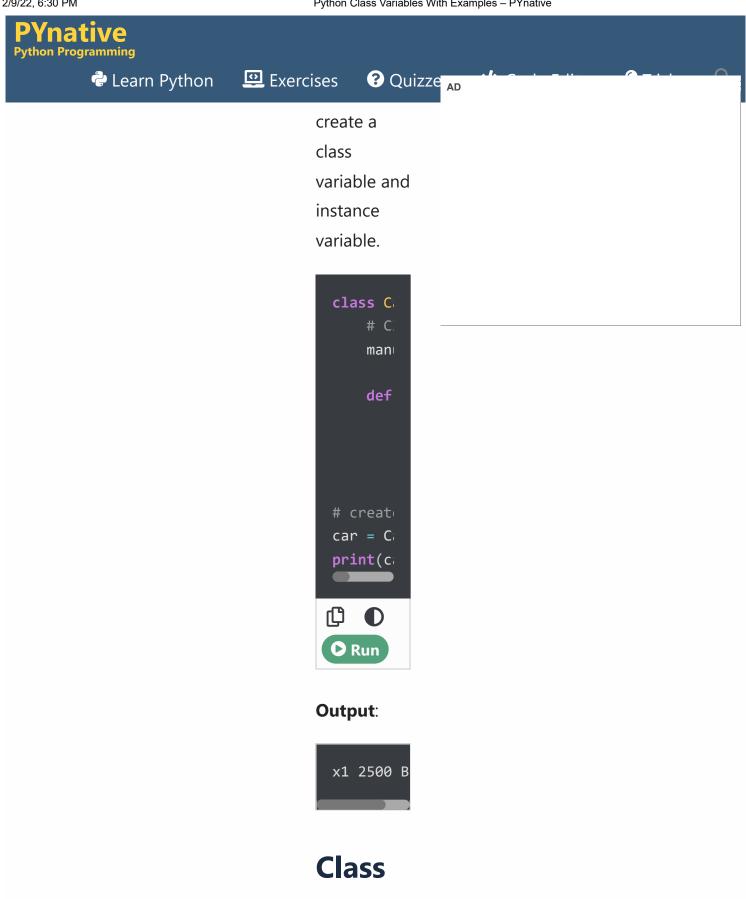
object











**Variab** 

les In



**?** Learn Python





As you

know, only

one copy of

the class

variable will

be created

and shared

between all

objects of

that class.

When we

use

inheritance,

all variables

and

methods of

the base

class are

available to

the child

class. In

such cases,

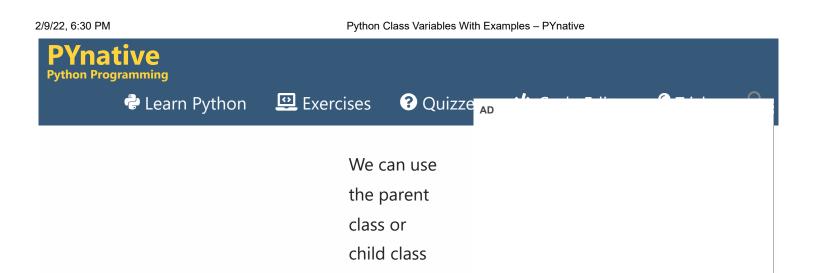
We can also

change the

value of the

parent

class's class

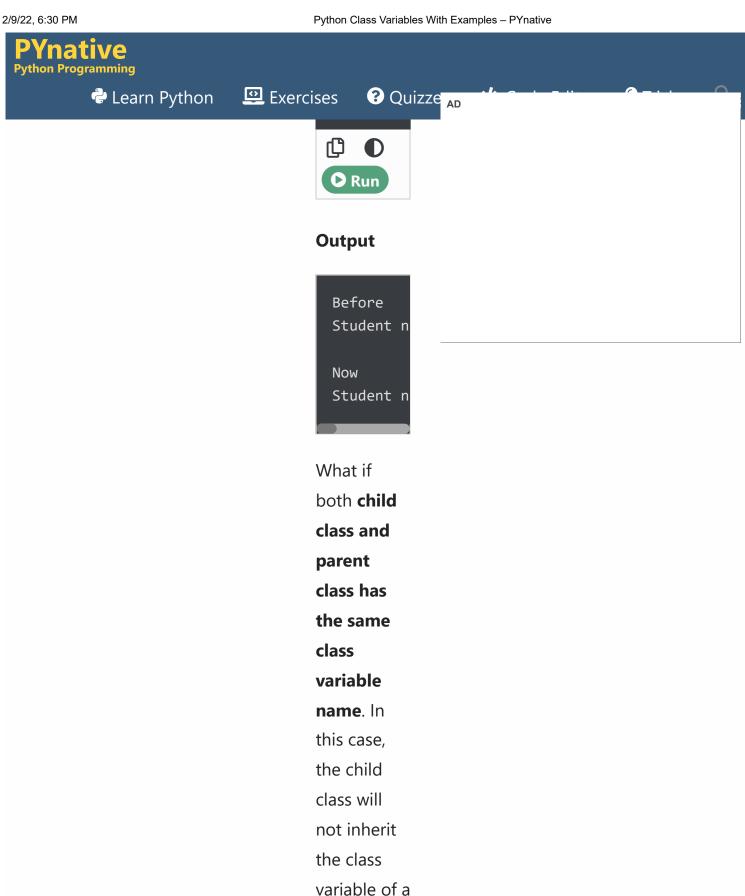


change the value of a parent class's class variable in the child class.

name to

#### **Example**

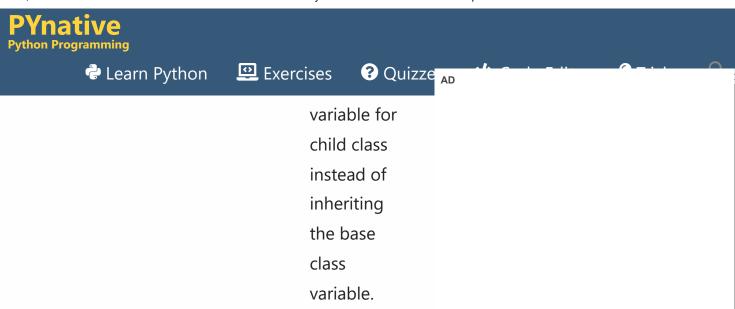
class Coursell coursell class State def



base class.

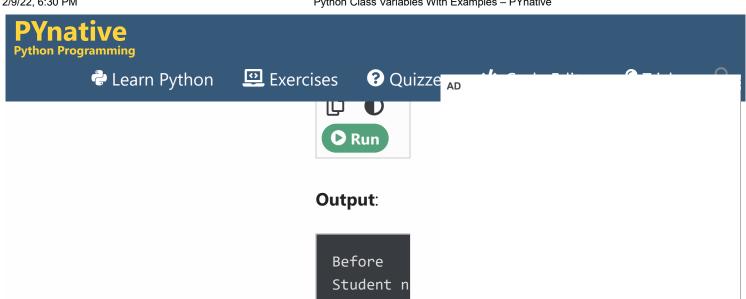
recommend

So it is



## Example:

```
class C
     cou
class S<sup>.</sup>
     cou
     def
     def
# creat:
stud = !
stud.sh
```



# Wron

Now

Student n

Parent Cl

g Use

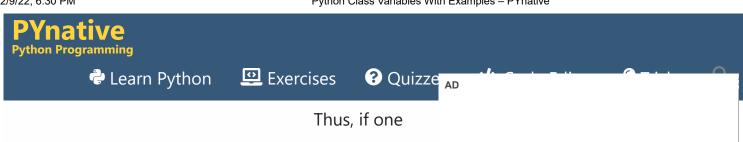
of

**Class** 

**Variab** 

## les

In Python, we should properly use the class variable because all



of the objects modifies the value of a class variable, then all

objects start referring to the fresh copy.

For example,

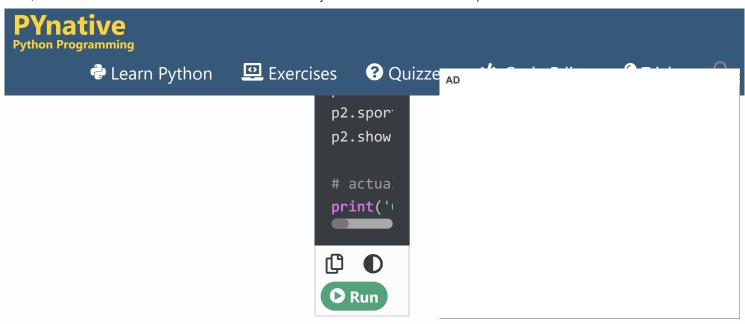
## **Example**

class P:
 # c.
 clul
 spo

def

def

# wrong



#### **Output**

Player : Player : Club: Che

In the above example, the instance variable

name is unique for each player.

The class

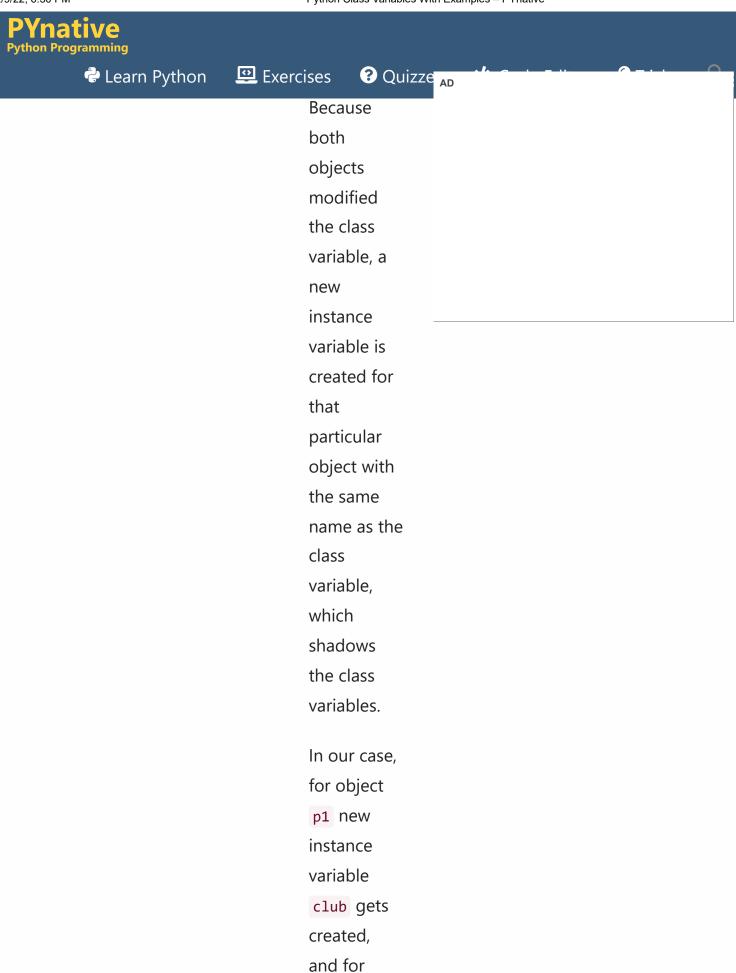
variable

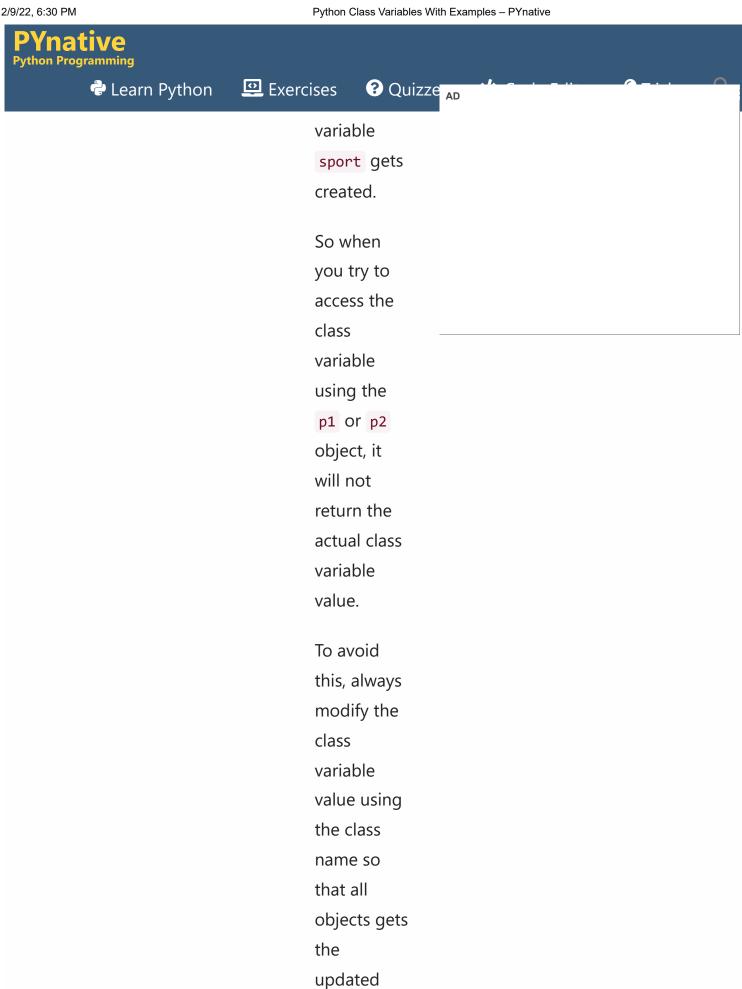
team and

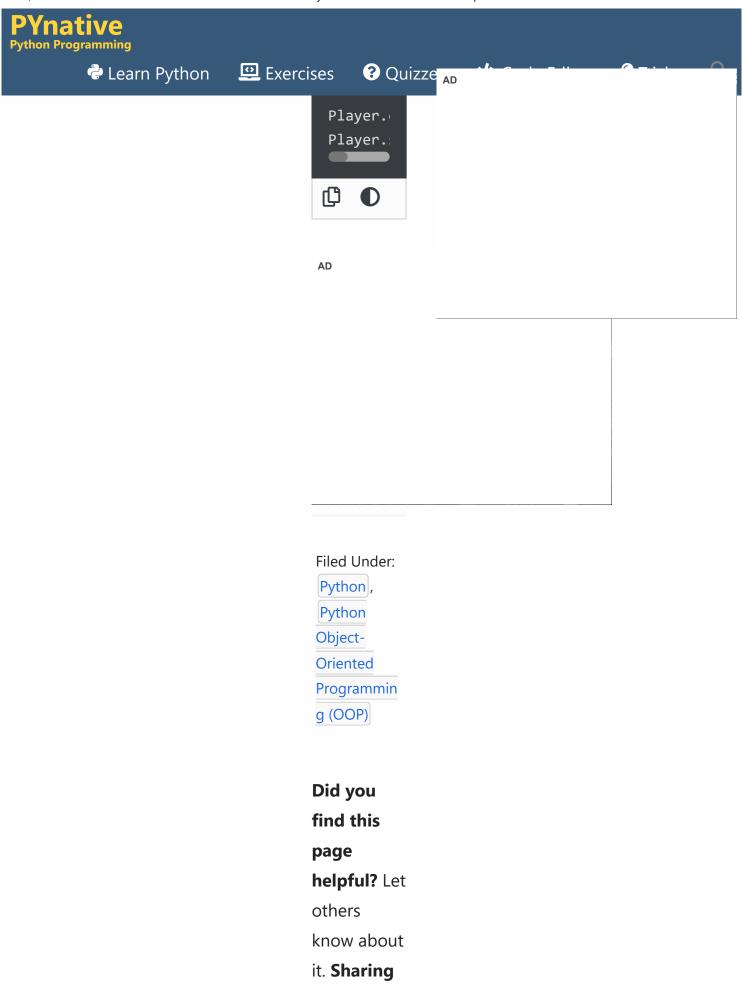
sport can

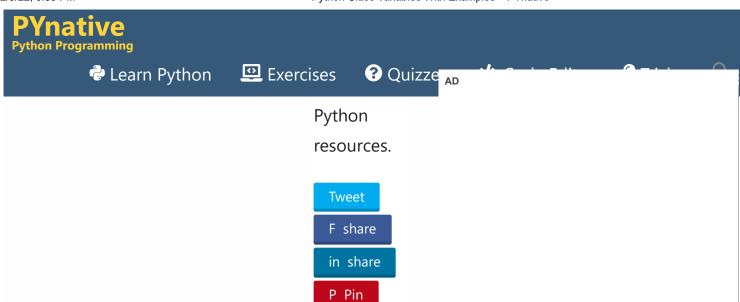
be accessed

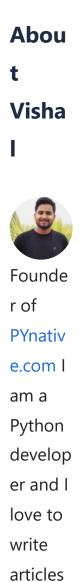
and



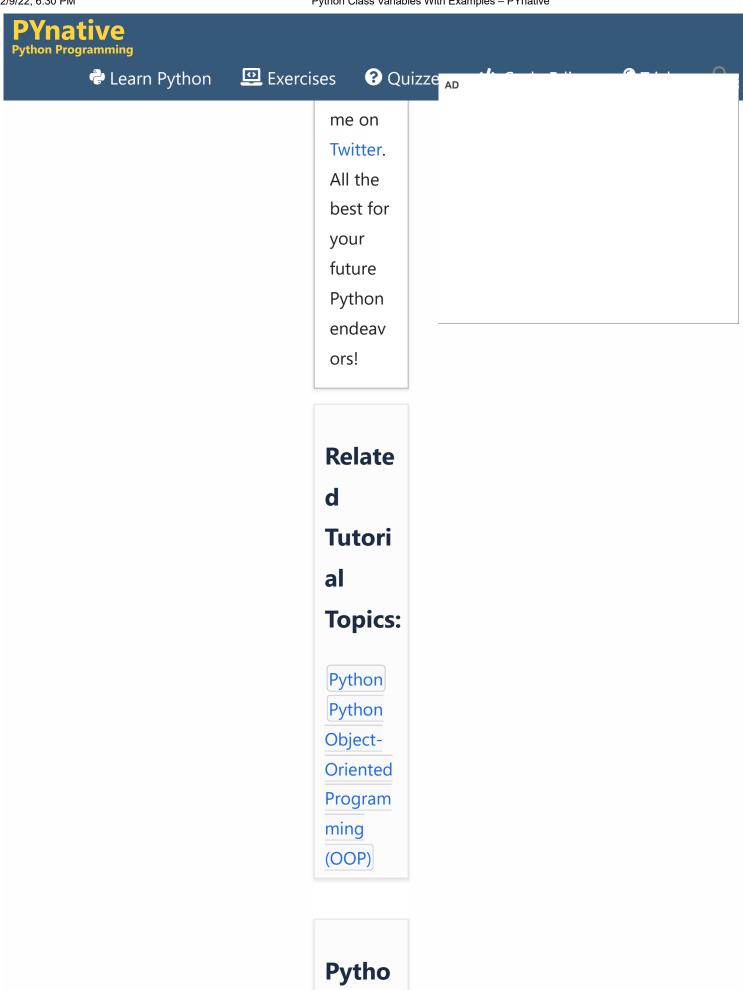


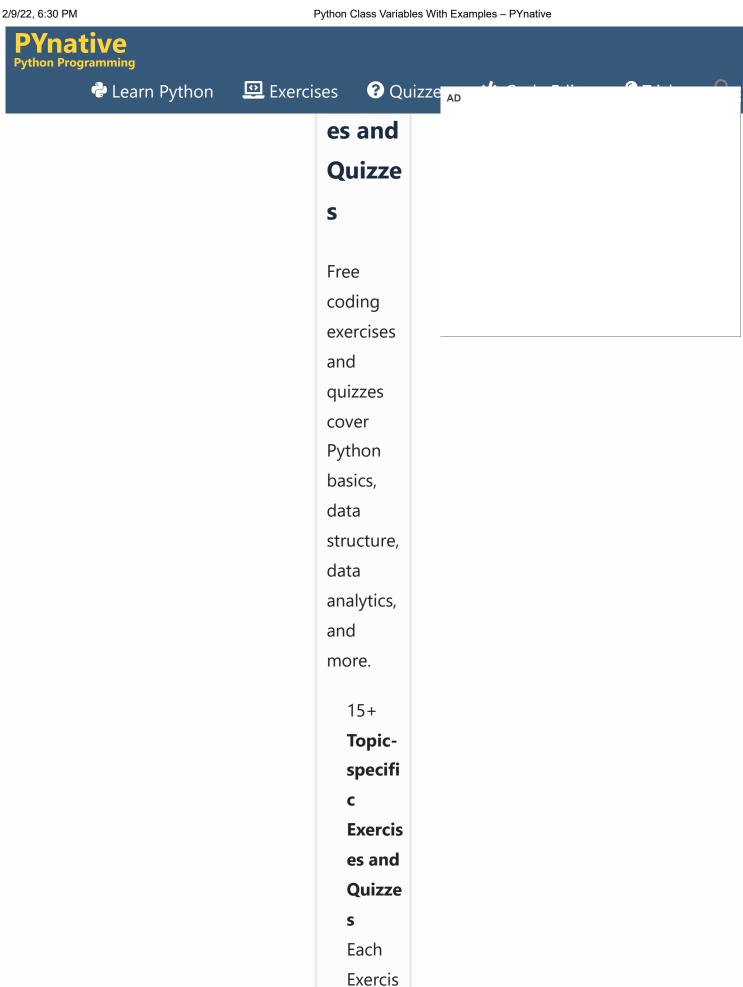


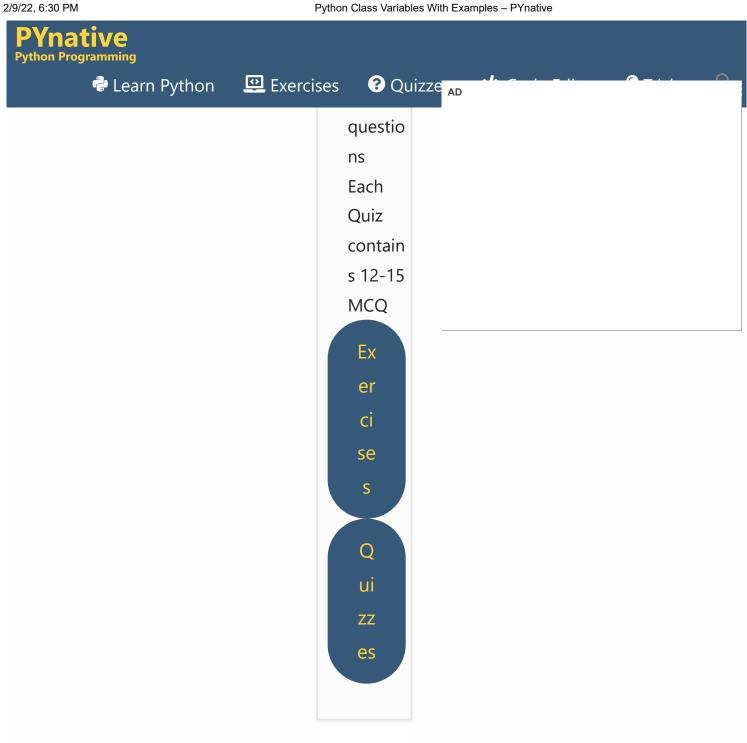


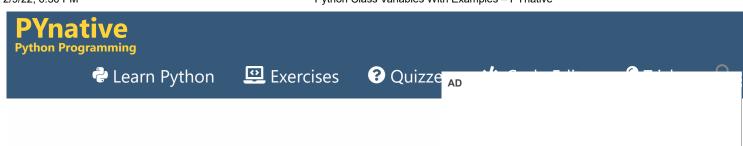


to help









#### **About PYnative**

PYnative.com is for Python lovers. Here, You can get Tutorials, Exercises, and Quizzes to practice and improve your Python skills.

## **Explore Python**

- Learn Python
- Python Basics
- PythonDatabases
- Python Exercises
- Python Quizzes
- Online Python
   Code Editor
- Python Tricks

### **Follow Us**

To get New Python Tutorials, Exercises, and Quizzes

- Twitter
- Facebook
- Sitemap

## **Legal Stuff**

- About Us
- Contact Us

We use cookies to improve your experience. While using PYnative, you agree to have read and accepted our Terms Of Use, Cookie Policy, and Privacy Policy.

