



**MALAD KANDIVALI EDUCATION SOCIETY'S
NAGINDAS KHANDWALA COLLEGE OF COMMERCE,
ARTS & MANAGEMENT STUDIES & SHANTABEN NAGINDAS
KHANDWALA COLLEGE OF SCIENCE
MALAD [W], MUMBAI – 64
(AUTONOMOUS)**

**(Reaccredited 'A' Grade by NAAC)
(AFFILIATED TO UNIVERSITY OF MUMBAI)
(ISO 9001:2015)**

CERTIFICATE

Name: Mr./Ms. Pooja Pramanik

Roll No: 67 Programme: BSc IT Semester: II

This is certified to be a bonafide record of practical works done by the above student in the college laboratory for the course **IT platforms, Tools and Practices** (Course Code: **2026UISTP**) for the partial fulfillment of Second Semester of BSc IT during the academic year 2020-2021.

The journal work is the original study work that has been duly approved in the year 2020-2021 by the undersigned.

**External Examiner
(Ms.Sweety Garg)**

Subject-In-Charge

Date of Examination: (College Stamp)

Name: Pooja Pramanik

Roll No: 67

Sr. No.	DATE	TITLE	SIGN
1.	02/02/2021	INTRODUCTION and CONTRIBUTING TO WIKIPEDIA a) What is Wikipedia? b) Steps to Create Account on Wikipedia c) Creating Page on Wikipedia d) Edit your page	
2.	09/02/2021	Creating account, repository on GitHub and Cloning repository in GitHub Page	
3.	16/02/2021	BASIC UNDERSTANDING ON FREE AND OPEN-SOURCE SOFTWARE a) Describe Open-Source Software with Example. b) Describe Free Software with Example c) Difference between Free and Open-Source Software.	
4.	23/02/2021	WRITING EMAIL	
5.	25/02/2021	Using practical examples, describe green computing. List and explain the steps that you take to contribute to green computing	
6.	02/03/2021	WRITING BLOGS	
7.	09/03/2021	Implementing coding practices in Python using PEP8.	
8.	20/03/2021	PRESENTATION: PEP8	

Practical1:Introduction and Contribution to Wikipedia

a) Description about Wikipedia and its features

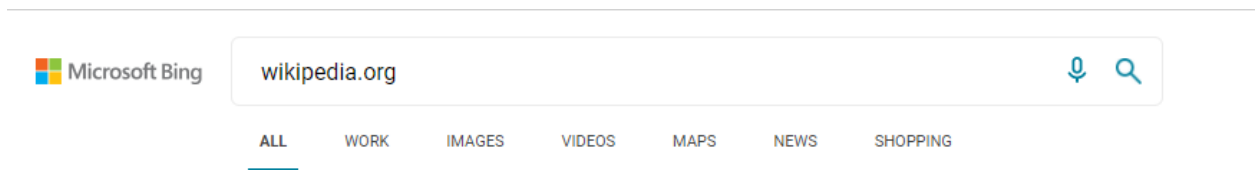
A Wikipedia consist of many wiki pages. A single page in a wiki website is referred to as a "wiki page".Wikipedia's purpose is to bnefit readers by acting as a free widely accessible encyclopedia.

Features of Wikipedia:-

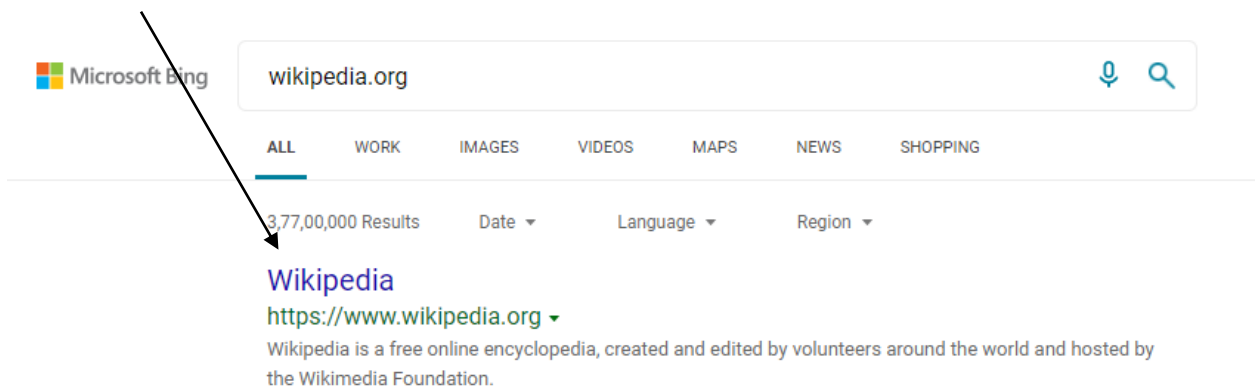
Wikipedia is a great place to start research, giving background information on the topic and possible keywords to help conduct more in-depth research . Sources used in the articles are cited, allowing further investigation into any topic.

b) Creating Account on Wikipedia

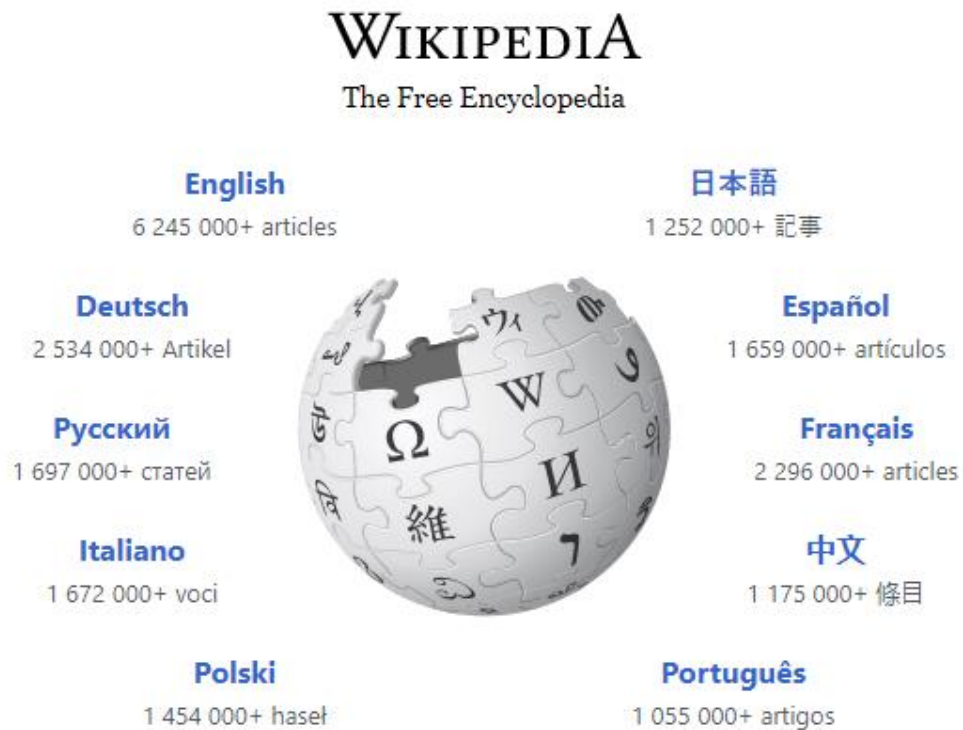
Step1. On the search bar type Wikipedia.org



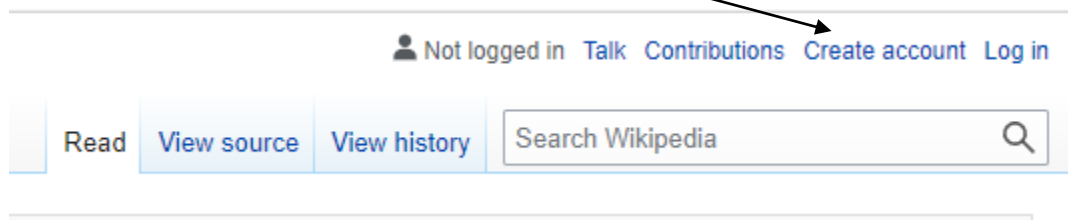
Step2. Click on Wikipedia Below mention link



Step3. Select your Language



Step 4. Click on the create account on the top right



Step 5. After clicking on create account it will look like the below image

Special page

Create account

Please consider using an **anonymous username**, and not your real name.

Once an account has been created, it is essentially impossible to delete it.

Username (help me choose)

Password

It is recommended to use a unique password that you are not using on any other website.

Confirm password

Email address (optional)

 Rectangular Snip

To protect the wiki against automated account creation, please complete the CAPTCHA below.

Wikipedia

Help

Learn to edit

Community portal

Recent changes

Upload file

Tools

Upload file

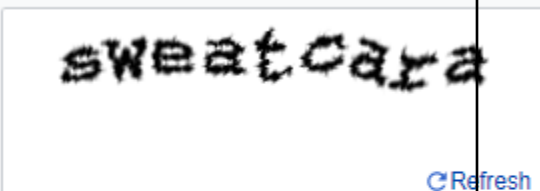
Special pages

Printable version

Languages

Step 6. Enter all your details and then click on create your account done.

CAPTCHA Security check

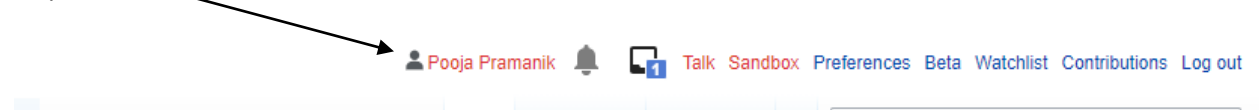


[Refresh](#)

Can't see the image? [Request an account](#)

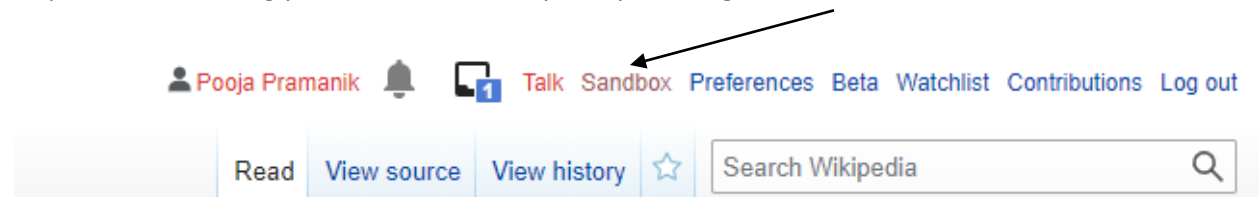
Create your account

Step 7. Done



c) Creating your page on Wikipedia

Step 1. After creating your account on Wikipedia you will get a button name sandbox

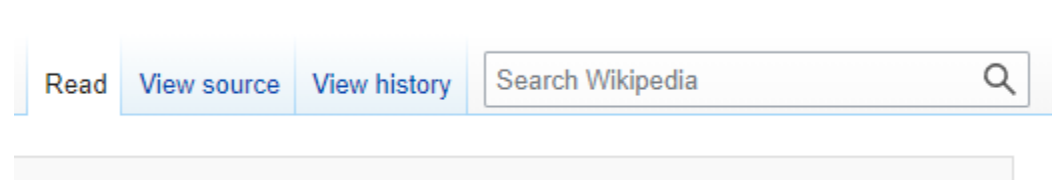


Step 2. After clicking on the mention button

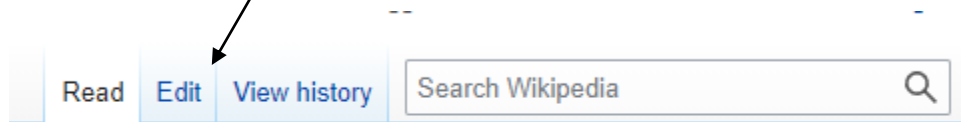


d) Editing your page on Wikipedia

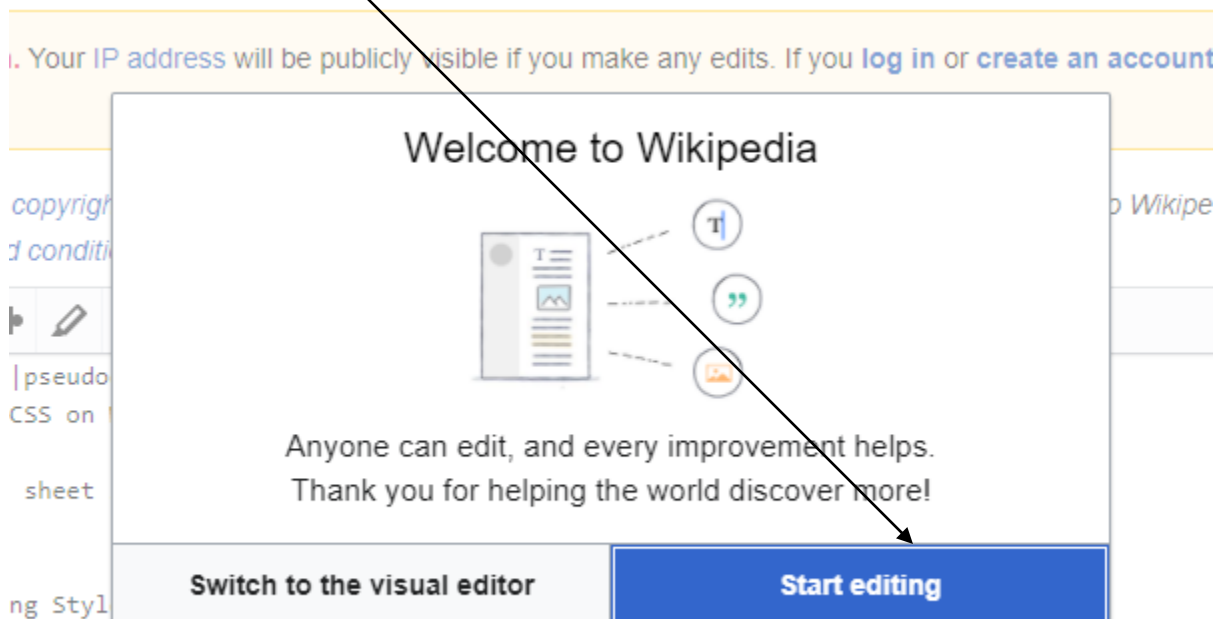
Step1. Search for the page which you want to edit on the search page



Step 2. Click on the edit option



Step2. Click on Start Editing than you can edit

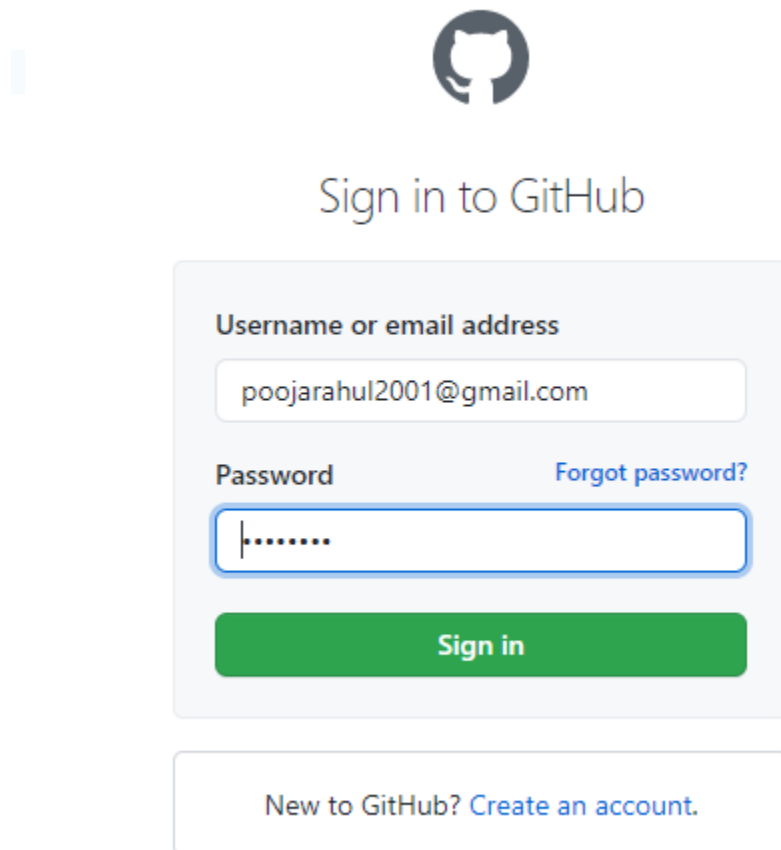


Practical2:Creating account, repository on Github and Cloning repository in Github

Mention Steps with Screenshot

a) Creating Account

Step1 . Fill your details to sign in



The screenshot shows the GitHub sign-in interface. At the top is the GitHub logo. Below it is the text "Sign in to GitHub". The main form is a light gray box containing two input fields: "Username or email address" with the value "poojarahul2001@gmail.com" and "Password" with masked characters ".....". A blue link "Forgot password?" is next to the password field. Below the inputs is a green "Sign in" button. At the bottom of the form is a link "New to GitHub? Create an account."

Step2. Fill the details to create your account and verify

[Join GitHub](#)

Create your account

Username *

PoojaPramanik



Email address *

poojarahul2001@gmail.com

Password *

.....

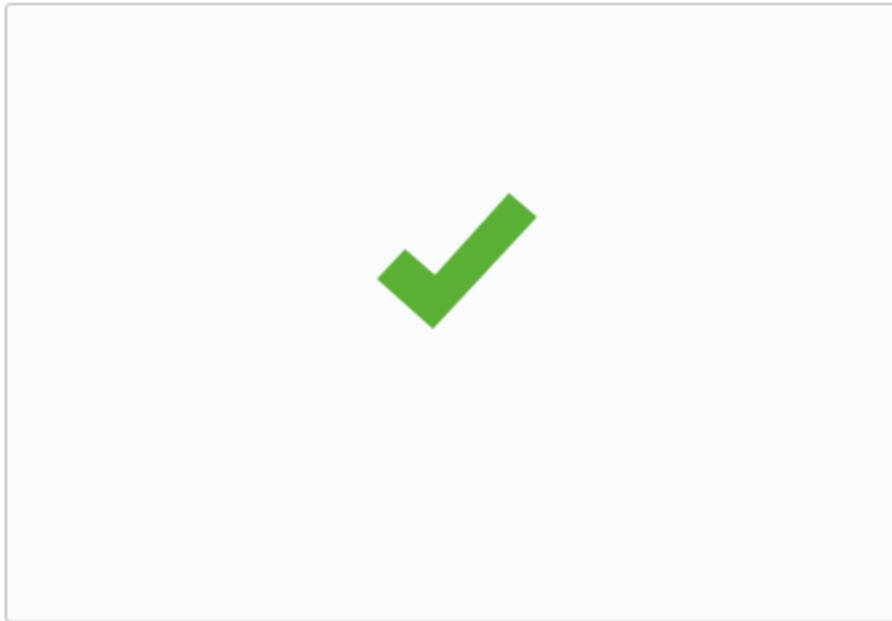
Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.
[Learn more.](#)

Email preferences

☒ Send me occasional product updates, announcements, and offers.

Verify your account

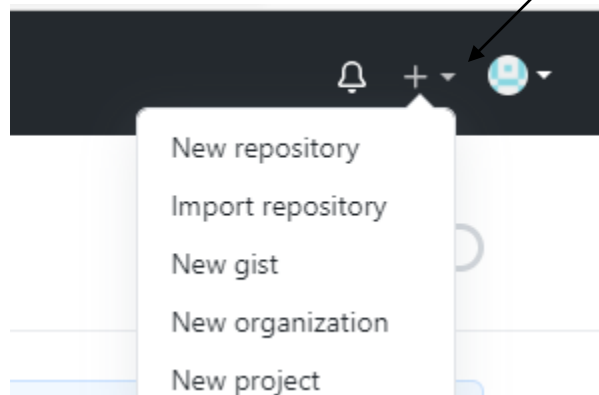
Verify your account



Create account


b) Creating Repository

Step1 . Click on this icon to create new repository




Step2. Fill the details to create a repository


Owner * Repository name *

 PoojaPramanik ▾ / IT_Tools ✓

Great repository names are short and memorable. Need inspiration? How about [super-octo-memory](#)?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.


☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:
Skip this step if you're importing an existing repository.

☒ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

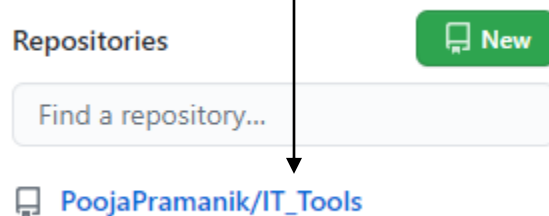
☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

This will set  **main** as the default branch. Change the default name in your [settings](#).

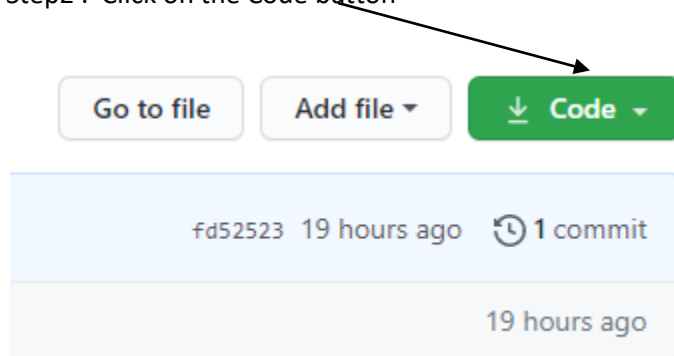
Create repository

c) Cloning repository

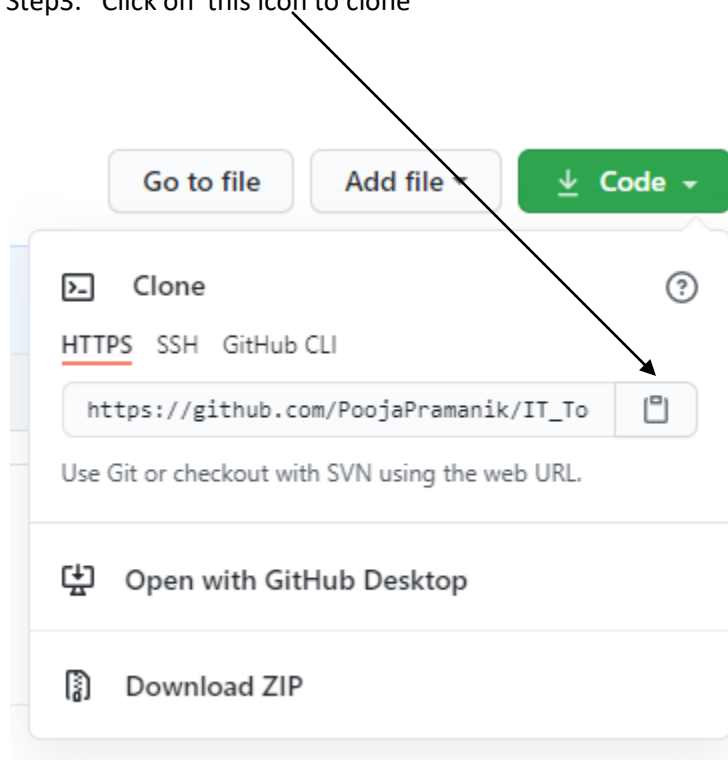
Step1. Click on your repository



Step2 . Click on the Code button



Step3. Click on this icon to clone



PRACTICAL 3: BASIC UNDERSTANDING ON FREE AND OPEN-SOURCE SOFTWARE

PRACTICAL 3: BASIC UNDERSTANDING ON FREE AND OPEN-SOURCE SOFTWARE

a) Describe Open Source Software with Example.

Open-source software is a type of computer software in which source code is released under a license in which the copyright holder grants users the rights to use, study, change, and distribute the software to anyone and for any purpose. Open-source software may be developed in a collaborative public manner. Open-source software is a prominent example of open collaboration.

The term open source refers to something people can modify and share because its design is publicly accessible.

The term originated in the context of software development to designate a specific approach to creating computer programs. Today, however, "open source" designates a broader set of values—what we call "the open source way."

Open source software is software with source code that anyone can inspect, modify, and enhance.



Linux



GNOME



Paint

Example : _____

b) Describe Free Software with Example

Free software is software that can be freely used, modified, and redistributed with only one restriction: any redistributed version of the software must be distributed with the original terms of free use, modification, and distribution..

Free software is a program used and distributed at no charge to the user. However, most free software licenses include terms prohibiting the sale, resale or commercial use.

Free software is often released for promotional purposes. For example, certain free software programs may require registration with an email address. The software owner collects these email addresses, which may be used to promote or sell other product

The concept of free software originally implied that when released the source code would be provided with permission to reuse the software within the public domain.



Example :

C) Difference between Free and Open Source Software.

Free Source software	Open Source Software
1 Open source software refers to the computer software which source is open means the general public can access and use.	1 Closed source software refers to the computer software which source code is closes means public is not given access to the source code.
2 In short it is referred as OSS.	2 In short it is referred as CSS.
3 The source code of open source software is public.	3 In closed source software the source code is protected.
4 The price of open source software is very less and there is no so much restrictions on users based on usability and modification of software.	4 The price of closed source software is high and users need to have valid and authenticated license to use the software.
5 Some examples of open source	5 Some examples of closed source

software are Firefox, OpenOffice, Gimp, Alfresco, Android, Zimbra, Thunderbird, MySQL, Mailman, Moodle, TeX, Samba, Perl, PHP, KDE etc.	software are Skype, Google earth, Java, Adobe Flash, Virtual Box, Adobe Reader, Microsoft office, Microsoft Windows, WinRAR, mac OS, Adobe Flash Player etc.
--	---

PRACTICAL 4: WRITING EMAIL

Seeking for your permission Ma'am

sweety@nkc.ac.in

Seeking for your permission Ma'am

Respected Ma'am

Greetings of the day

I, Pooja Pramanik, Student of FYIT Roll no 67 is writing this email to you in concern that on 16th of March I won't be able to attend the practical lecture of IT Tools Because on the same day . I have to attend a webminar of AI, IOT .

I request you kindly grant me a leave for 1 practical lecture.

Looking forward for your positive response Ma'am

Encl: My Reports

regards

Pooja Pramanik

FYIT - 67



Using practical examples, describe green computing. List and explain the steps that you take to contribute to green computing

Green computing is environmentally responsible and eco-friendly use of computers. It is also defined by being the using and disposing of computing devices in a way that reduces their environmental contact.

- 1) **Power down when not in use** Seems simple but many of us leave computers powered up for a long time when not in use. A large sum of power is being wasted, so if you're not using the computer, press the power button to shut it off until needed. This can be done even if the computer is working on something. Screensavers do not save power. Same goes for computers, you don't have to shut it down completely if you don't want to reboot, just use sleep or hibernation mode. This will help save energy and keep the system to its current state when you need it again.
- 2) **Use the power saving features** All computers include power saving options. Using these features you can command the computer to do various energy-saving tasks automatically, including shutting off unused hard disks, powering off a monitor after a given time or even placing the computer into sleep mode when not in use. This is very useful on laptops to help preserve battery life.
- 3) **Purchase energy saving hardware** If you don't need super-fast computing power then look out for energy efficient components when buying a new computer, such as green hard drives and low-energy processors. While performance is slower they can use remarkably less power. Purchasing an energy saving power supply unit for a desktop PC can help the environment and save money, they're often quieter too.
- 4) **Use a laptop instead of desktop** Laptops are much better for the environment than desktop computers as they have components which require less power. If you don't need a desktop computer consider buying a laptop instead, or if you have both use the laptop as much as possible before considering the desktop.
- 5) **Recycle responsibly** Computer hardware is filled with different material which can be hazardous to the environment so make sure you dispose of old components effectively. Don't just throw broken technology in the bin, take the time to trace local recycling organizations. There should be companies which can remove the metals which may fix or furnish items. You should check with your local authorities to find out what facilities they offer for safe disposal of old computing parts.
- 6) **Don't throw your old computer away**
Globally over 35 million PC's are thrown away every year - yet there are many companies now recycling or reconditioning components or whole computers. Don't throw it away. Your old computer might be worth something either to a dealer, a local school or a charity.

Blog Writting

Link For Blog :

[Tour in Delhi and Agra \(pramanikpooja.blogspot.com\)](http://pramanikpooja.blogspot.com)

← Experience Can't Be Greater Than Your Age



Tour in Delhi and Agra



- March 23, 2021

Red Fort

The Red Fort is a historic Fort in the city of Delhi this fort covers a quite good amount of space it consists of museums. In the museum the collections of thing that had been used in the ancient time .I found the Museum very informative when I had a visit in Red Fort during Diwali vacation time beside the Red Fort there was a fair with huge number of crowd I really had a fun time and in that fair .



Qutub Minar

The world's tallest brick
Tower the magnificent
Qutub Minar is situated
in Delhi its height is of
73 meters its premises cover
a huge space apart from
the Kutub Minar Tower
it is looks like a fort.



Chandni Chowk



Chandni Chowk is a busy shopping area with Market full of wide variety of things I was really fascinated to others market because it had a huge collection of everything with affordable price within this market has a famous Jalebi shop those Jalebi are really mouth-watering item I must say that is someone is planning to visit Delhi then definitely have a visit in Chandni Chowk.

Taj Mahal

Marvelous beautiful and stunning words can never be enough to describe the beauty of the Taj Mahal at the moment when I had a visit in Taj Mahal I got the positive wise which brought the piece of my mind. I found Most of the tourist are foreigner at this monument.



Agra Fort



Agra Fort is a historical Fort in the city of Agra in India there I smell the fragrance of ancient time even from there I had a view of Taj monument.

Meena Bazaar

Meena Bazaar it's a best place for shopping, especially for women. It has a wide variety of sarees, shoes, decorative articles etc. but it is most popular for Petha (Panchi Petha). I had discovered a wide variety of Petha. example:- Pan Petha, Grapes petha, Mix fruit Petha etc.



Baba Baghel Singh Gurudwara

Baba Baghel Singh Gurudwara is situated in Delhi. This is the first time in my life that I had a visit in gurudwara really I am glad that I had been to this place because I came to know about the Sikh culture. In this Gurudwara someone or the other donate some amount of food on daily basis. The most astonishing fact in this Gurudwara, I find that there is a queue of sikh community those who all are collecting the shoes of other those who all are visiting the Gurudwara.



LOTUS TEMPLE

The Lotus temple derives its name from its design the way the temple has been designed is really appreciated I guess the most tourists are being attracted toward this temple just because of its structural looks like a Lotus .

when I had enter in the temple really I found the piece of my mind no one can attend a single word inside the temple because the voice is going to be echo show all the instructions are given before entering into the temple .



FOOD 🍌 🍌

It's a mixed Fruit **Petha** (Panchi Petha) .
It is famous Agra , really it's taste is just awesome.





In Chandni Chowk this dish is known as

Tandoori Potato Chilli. It's taste just can't

express it in words. It's that awesome.

It's a Soyabean Chaap .

It's a daily basis receipe in Delhi.



Here the laddoos for what the Delhi is famous for.

I find this dish as a unique one .

The name of this dish is **Dahi Wale**.



MEMORABLE DAYS IN MY LIFE

7. Implementing coding practices in Python using PEP8.

CODE :-

```
#Indentation
age = {
    'Allen': 18,
    'suzy': 20,
    'isabella': 19,
}
print(age)

#Naming Convention
name = 'Pooja Pramanik'
first_name, last_name = name.split()
print(last_name, first_name, sep=', ')

#Code layout
total = (10
        + 20
        - 5)
print(total)

#Whitespace Around Binary Operators
lists = [1, 2, 3]
print(lists)

x = 5
y = 6
print(x, y)

#Comment
Name = 'Pooja Pramanik' # Student Name
print(Name)
```

OUTPUT :-

```
{'Allen': 18, 'suzy': 20, 'isabella': 19}
Pramanik, Pooja
25
[1, 2, 3]
5 6
Pooja Pramanik
```

PEP 8 STYLE GUIDE

GROUP No. 8

ANNE VERONICA	1
PRIYA GUPTA	20
SHUBH PATEL	60
BHAVANA PRAJAPATI	66
POOJA PRAMANIK	67
HEETA TALAVIYA	92
GLORY LITHIYAL	95
NITESH GUPTA	111
AMAN UPADHYAY	132
DEEPAK KESHRI	134
SURYASEN VISHWAKARMA	139

PEP8

PEP8 is a style guide for python code.

- PEP stands for Python Enhancement Proposal, and they describe and document the way python language evolves.
- It was written in 2001 by Guido van Rossum, Barry Warsaw, and Nick Coghlan.
- A PEP is a document that describes new features proposed for Python and documents aspects of Python, like design and style, for the community.
- They also provide a reference point (and a standard) for the pythonic way to write code

→ It also has a lot of programming recommendations and useful tips on various topics, which aim to improve readability and reliability of your code.

→ PEP8 features:-

1. Plugin architecture: Adding new checks is easy.
2. Parseable output: Jump to error location in your editor.
3. Small: Just one Python file, requires only stdlib. You can use just the pep8.py file for this purpose.

Naming Conventions

Naming Conventions:

1.Variable

2.Function

3.Class

4.Method

5.Constant

6.Module

7.Package

,

Variable: A variable is created the moment you first assign a value to it

```
#Wrong Way to Initialize or assigning a name to a variable  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
1variable=2 #Variable name started with a number (Wrong Way)  
print(1variable)
```

```
File "<ipython-input-1-d1860915d72c>", line 5  
    1variable=2  
    ^  
SyntaxError: invalid syntax
```

```
#Wrong Way to Initialize or assigning a name to a variable  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
x='Bhavana' #Variable name is too common and not intuitive (Not a Good Way)  
print(x)
```

Bhavana

```
#Wrong Way to Initialize or assigning a name to a variable  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
first_name='Bhavana' #Variable name is self-explanatory and has a readability, and it is seperated using underscores  
print(x)
```

Bhavana

Function: A function is a block of code which only runs when it is called.

```
#Wrong Way to Initialize or assigning a name to a function  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
def ^function(): #Function name should not be started with a Number or special characters  
    print("Not a correct way to represent a function name")  
  
^function()
```

```
File "<ipython-input-5-5f84f1733e34>", line 5  
    def ^function():  
        ^  
SyntaxError: invalid syntax
```

```
#Wrong Way to Initialize or assigning a name to a function  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
def x(): #Function name is too generic and it can create a confusion in enterprise programming  
    print("Function Name is too generic, you can use it but it is not recommended as it is not self-explanatory and intuitive")  
  
x()
```

Function Name is too generic, you can use it but it is not recommended as it is not self-explanatory and intuitive

```
#Wrong Way to Initialize or assigning a name to a function  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
def display_function(): #Function name is self explanatory  
    print("Function Name is self explanatory, name can be more intuitive in case of proper functionality")  
  
display_function()
```

Function Name is self explanatory, name can be more intuitive in case of proper functionality

Class: class definitions begin with a class keyword.

```
#Wrong Way to Initialize or assigning a name to a class  
#Name Should not start with a number  
#Name should be intuitive and not too common.  
  
1class x:  
def display_function(): #Function name is self explanatory  
    print("Function Name is self explanatory, name can be more intuitive in case of proper functionality")  
  
display_function()
```

File "<ipython-input-9-0547726683a1>", line 5

```
1class x:  
  ^
```

SyntaxError: invalid syntax

```
class Employee:  
    def accept(self):  
        print("Enter Id:")  
        self.Id=int(input())  
        print("Enter Name:")  
        self.name= str(input())  
    def display(self):  
        print("ID: %d \nName: %s"%(self.Id,self.name))  
  
emp=Employee()  
emp.accept()  
emp.display()
```

```
Enter Id:  
66  
Enter Name:  
bhavana  
ID: 66  
Name: bhavana
```

Method: A Python method is a label that you can call on an object; it is a piece of code to execute on that object.

```
#Wrong Way to Initialize or assigning a name to a method  
#Name Should not start with a number  
#Name should be intuitive and not too common.
```

```
1class Method:  
    def display(self):  
        print("This is method function. ")  
  
c = Method()  
c.display()
```

File "<ipython-input-26-3e88b14da450>", line 6

```
1class Method:
```

^

SyntaxError: invalid syntax

```
#Wrong Way to Initialize or assigning a name to a class  
#Name Should not start with a number  
#Name should be intuitive and not too common
```

```
class Product:  
    def __init__(self):  
        self.prod_id = input("Enter the Product ID: ")  
        self.prod_name = input("Enter the Product Name: ")  
        self.total_no = int(input("Enter the total no. of Items Purchase: "))  
        self.unit_price=float(input("Enter the unit Price: "))  
    def display(self):  
        print("Total Price of %d units of Product %s is: %0.2f" %(self.total_no,self.prod_name,self.total_no*self.unit_price))  
  
p1=Product()  
p1.display()
```

```
Enter the Product ID: A20134  
Enter the Product Name: Chocolate  
Enter the total no. of Items Purchase: 7  
Enter the unit Price: 75.50  
Total Price of 7 units of Product Chocolate is: 528.50
```

Constant: A constant is a type of variable whose value cannot be changed.

```
pi = 3.14                #pi is constant
radius=5
print("Area of circle: %0.2f" %(pi*radius*radius))
```

Area of circle: 78.50

Modules: Modules refer to a file containing Python statements and definitions.

```
# to import standard module math

import math
print("The value of pi is", math.pi)
```

The value of pi is 3.141592653589793

Packages: A package is basically a directory with Python files and a file with the name `__init__.py`



Code layout

WITHOUT SPACE

These conventions lead to text that you can read easily, like this:

This would become increasingly hard to read. For example have a look at the example below

```
howwillitlookifwedonothavethespace
```

WITH SPACE

Now here, we will use space and write it in regular English language, so it will be very easy to read.

```
How will it look if we do not have the space
```

Maximum line length and line breaking

PEP 8 guidelines suggest that each line of code (as well as comment lines) should be 79 characters wide or less. This is a common standard that is also used in other languages including R.

CORRECT

```
# Perform some math
a = 1+2
b = 3+4
c = a+b

# Read in and plot some
precip_timeseries = pd.readcsv("precip-2019.csv")
precip_timeseries.plot()
```

#WRONG

```
#Perform some math and do some things
a=1+2
b=3+4
c=a+b
data=pd.readcsv("precip-2019.csv")
data.plot()
```

Should a line break Before or After a Binary Operator

Here, it's harder to see which variable is being added and which is subtracted.

WRONG

```
Total = (Number 1+  
          Number 2-  
          Number 3)
```

You can immediately see which variable is being added or subtracted, as the operator is right next to the variable being operated on.

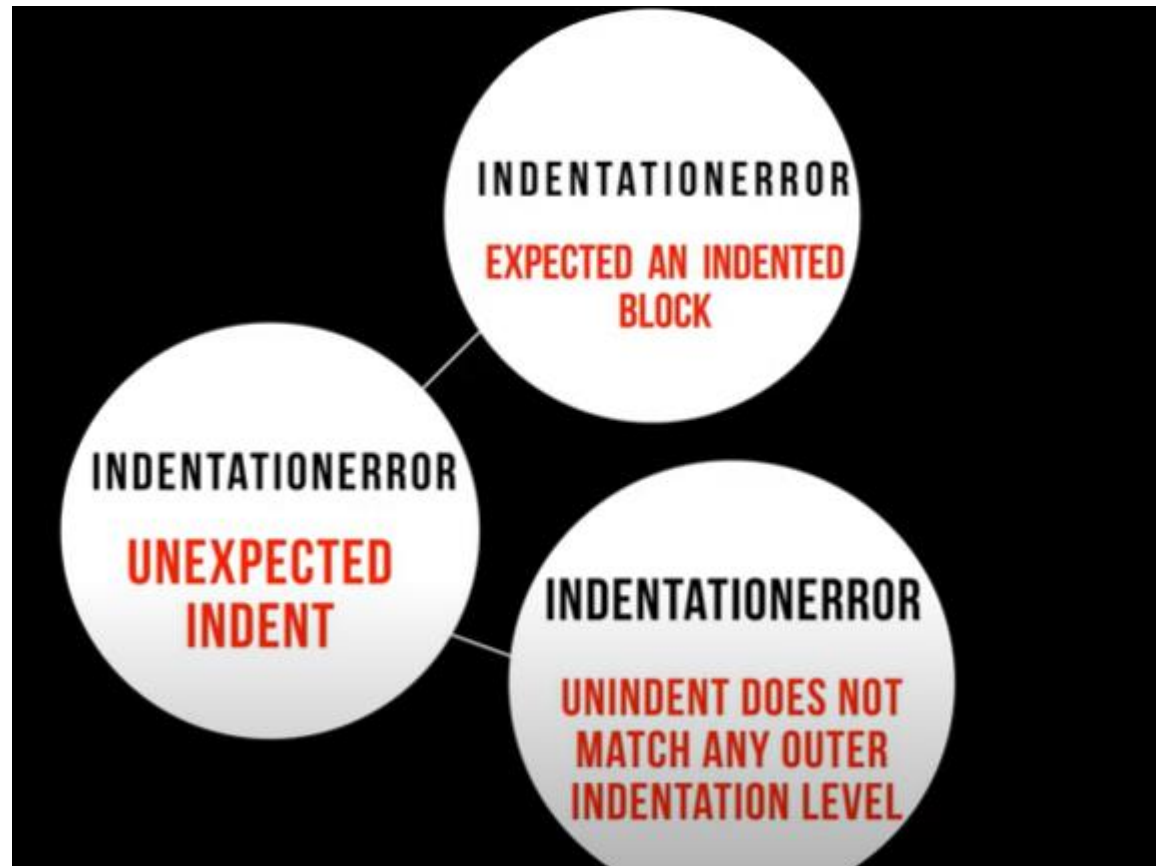
In the below Example

#CORRECT

```
Total = (Number 1  
          + Number 2  
          - Number 3)
```

Indentation

- Indentation is extremely important in Python.
- The Indentation level of lines of code in python determines how statements are grouped together.



1. Expected an indented block



```
x = 2
if x % 2 == 0:
    print("It is an even number")
```



```
File "<ipython-input-20-d2c95d58e212>", line 3
    print("It is an even number")
    ^
```

IndentationError: expected an indented block



```
x = 2
if x % 2 == 0:
    print("It is an even number")
```

It is an even number


2. Unexpected Indent



```
x = 2
if x % 2 == 0:
    print("It is an even number")
```



```
File "<ipython-input-24-a296ed44a7f2>", line 2
    if x % 2 == 0:
      ^
IndentationError: unexpected indent
```



```
x = 2
if x % 2 == 0:
    print("It is an even number")
```

```
It is an even number
```

3. Unindent does not match any outer indentation level

```
def greeting():  
    print("Greetings of the day")  
    return  
  
greeting()
```

File "<ipython-input-30-698032a46f85>", line 3
 return
 ^
IndentationError: unindent does not match any outer indentation level

```
def greeting():  
    print("Greetings of the day")  
    return  
  
greeting()
```

Greetings of the day

Tabs vs. Spaces

➤ Tabs vs. Spaces

The key indentation rules laid out by PEP 8 are the following:

- Use 4 consecutive spaces to indicate indentation.
- Prefer spaces over tabs.

➤ Indentation following line breaks

- Add a comment after the final condition. Due to syntax highlighting in most editors, this will separate the conditions from the nested code:

Not Recommended

```
▶ x = 5
  if (x > 3 and
      x < 10):
      print(x)
```

Recommended

```
▶ x = 5
  if (x > 3 and
      x < 10):
      # Both conditions satisfied
      print(x)
```

```
▶ x = 5
  if (x > 3 and
      x < 10):
      print(x)
```

Not Recommended



```
var = function(arg_one, arg_two,  
               arg_three, arg_four)
```

Recommended



```
var = function(  
    arg_one, arg_two,  
    arg_three, arg_four)
```

Not Recommended



```
def function(  
    arg_one, arg_two,  
    arg_three, arg_four):  
    return arg_one
```

Recommended

```
def function(  
    arg_one, arg_two,  
    arg_three, arg_four):  
    return arg_one
```

➤ Where to put the closing Braces

Not Recommended

```
list_of_numbers = [ 1, 2, 3,  
                    4, 5, 6,  
                    7, 8, 9]
```

1. Method

```
list_of_numbers = [  
    1, 2, 3,  
    4, 5, 6,  
    7, 8, 9  
]
```

2. Method

```
list_of_numbers = [  
    1, 2, 3,  
    4, 5, 6,  
    7, 8, 9  
]
```

COMMENTS:

Comments are lines that exist in computer programs that are ignored by compilers and interpreters.

Comment begins with a hash mark (#)

Generally, comment looks like this:

this a comment

Because comment does not execute ,when you will run program you will not see any indication of the comment there.

BLOCK COMMENTS:

Each line of block comments starts with a `#` and a single space

Paragraphs inside a block comment are separated by a line containing a single `#`.

Anti-pattern

```
#This comment needs a space  
def print_name(self):  
    print(self.name)
```

Best practice

```
# Comment is correct now  
def print_name(self):  
    print(self.name)
```

INLINE COMMENTS:

Inline comment should be separated by at least two spaces from the comment.

They should start with a `#` and a single space

Inline comments are unnecessary and in fact distracting if they state the obvious

Anti-pattern

```
def print_name(self):  
    print(self.name) #This comment needs a space
```

Best practice

```
def print_name(self):  
    print(self.name) # Comment is correct now
```

DOCSTRING COMMENTS:

A docstring is added as a comment string right below the function, module, or object

RULES:

A docstring is either a single line, or a multi-line comment

In latter case, the first line is short description, and after the first line an empty line follows

This is a basic example of what it looks like:

```
def add(value1, value2):  
    """Calculate the sum of value1 and value2."""  
    return value1 + value2
```

In the Python interactive help system, the docstring is then made available via the `__doc__` attribute.

```
>>> print add.__doc__  
Calculate the sum of value1 and value2.
```

Inline Comments vs Block Comments

Inline comments look like this

```
x = x + 1          # Compensate for border
```

While block comments look like this

```
# Compensate for border.  These comments  
# often cover multiple lines.  
x = x + 1
```

Whitespace in Expressions and Statements

1) Whitespace Around Binary Operators

Surround the following binary operators with a single space on either side:

- Assignment operators (=, +=, -=, and so forth)
- Comparisons (==, !=, >, <, >=, <=) and (is, is not, in, not in)
- Booleans (and, not, or)

Note: When = is used to assign a default value to a function argument, do not surround it with spaces.

Python

```
# Recommended
def function(default_parameter=5):
    # ...

# Not recommended
def function(default_parameter = 5):
    # ...
```

- Adding space when there is more than one operator in a statement.

Python

Recommended

```
y = x**2 + 5
```

```
z = (x+y) * (x-y)
```

Not Recommended

```
y = x ** 2 + 5
```

```
z = (x + y) * (x - y)
```

- Adding space to if statements where there are multiple conditions.

Python

Not recommended

```
if x > 5 and x % 2 == 0:
```

```
    print('x is larger than 5 and divisible by 2!')
```

Python

Recommended

```
if x>5 and x%2==0:
```

```
    print('x is larger than 5 and divisible by 2!')
```

Note : Use the same amount of whitespace either side of the operator.

The following is not acceptable :

Python

```
# Definitely do not do this!  
if x >5 and x% 2== 0:  
    print('x is larger than 5 and divisible by 2!')
```

When to Avoid Adding Whitespace

- Trailing space
- Immediately inside parentheses, brackets, or braces:

Python

```
# Recommended  
my_list = [1, 2, 3]  
  
# Not recommended  
my_list = [ 1, 2, 3, ]
```

- Before a comma, semicolon, or colon:

Python

```
x = 5  
y = 6  
  
# Recommended  
print(x, y)  
  
# Not recommended  
print(x , y)
```


Before the open parenthesis that starts the argument list of a function call:

```
Python

def double(x):
    return x * 2

# Recommended
double(3)

# Not recommended
double (3)
```

Before the open bracket that starts an index or slice:

```
Python

# Recommended
list[3]

# Not recommended
list [3]
```

- Between a trailing comma and a closing parenthesis:

Python

```
# Recommended  
tuple = (1,)
```

```
# Not recommended  
tuple = (1, )
```

- To align assignment operators:

Python

```
# Recommended  
var1 = 5  
var2 = 6  
some_long_var = 7
```

```
# Not recommended  
var1      = 5  
var2      = 6  
some_long_var = 7
```

Programming Recommendations

❖ Two Programming Recommendations by PEP-8

A) # Not recommended

```
my_bool = 6 > 5
if my_bool == True:
    return '6 is bigger than 5'
```

B) # Recommended

```
if my_bool:
    return '6 is bigger than 5'
```

In the above program B is recommended over A by the PEP-8

C) # Not recommended

```
my_list = []
if not len(my_list):
    print('List is empty!')
```

D) # Recommended

```
my_list = []
```

```
if not my_list:
```

```
    print('List is empty!')
```

In the above program D is recommended over C by the PEP-8

Q. When to Ignore PEP-8

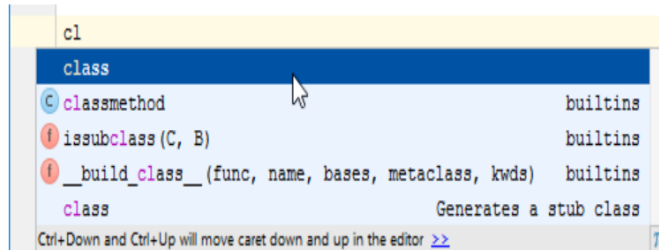
ANSWER: NEVER

Though, there are some guidelines in PEP-8 that are inconvenient in some instances:

- Complying with PEP-8
 - Code surrounding
 - Code compatibility

Tips and Tricks to Help Ensure Your Code Follows PEP 8

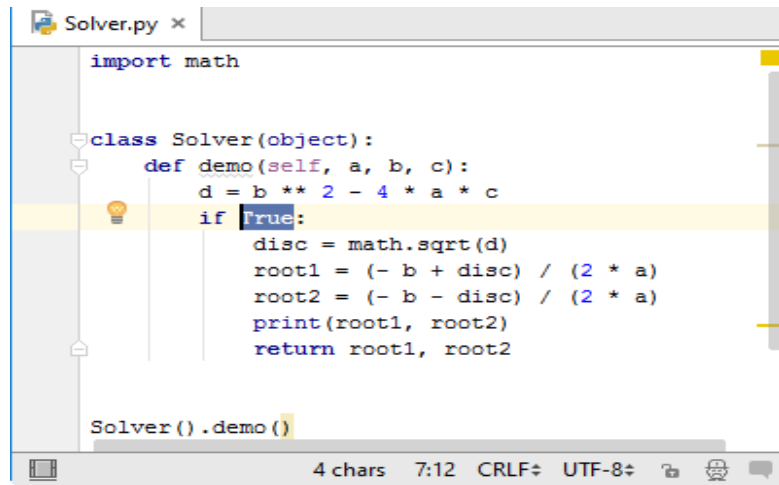
Highlighting code style violations:



(Refer to Code Completion page of the product documentation for details.)

Generating Source code:

Select `if` option from the suggestion list. As you see, PyCharm automatically adds `if True:` and indents the selected lines:



```
import math

class Solver(object):
    def demo(self, a, b, c):
        d = b ** 2 - 4 * a * c
        if True:
            disc = math.sqrt(d)
            root1 = (- b + disc) / (2 * a)
            root2 = (- b - disc) / (2 * a)
            print(root1, root2)
            return root1, root2

Solver().demo()
```

Linter-python-pep8 package

This linter-python-pep8 plugin or Linter provides an interface to pep8. It will be used with files that have the Python syntax.

Installation:

Before using this plugin, you should make sure that pep8 is installed on your system. You can follow following instructions to install pep8:

Install python.

Install pep8 by typing the following in a terminal:

```
pip install pep8
```

Black

Black can be installed by running `pip install black`. It requires Python 3.6.0+ to run. Once Black is installed, you will have a new command-line tool called `black` available to you in your shell, and you're ready to start.

```
$ pip install black
```

Format a Single File:

Let's look at this simple example: here are my two python functions in my python file called `sample_code.py`.

```
def add(a, b):  
    answer = a + b  
  
    return answer  
  
def sub(c ,  
d):  
    answer = c - d  
  
    return answer
```


You can use `black sample_code.py` in the terminal to change the format. After running Black, you will see the following output:

```
reformatted sample_code.py
All done! ✨💎✨
1 file reformatted.
```

Then you open `sample_code.py` to see formatted python code:

```
def add(a, b):
    answer = a + b

    return answer

def sub(c, d):
    answer = c - d

    return answer
```

Example of code and layout.

With space and without space.

WITH SPACE.

*ex- MY NAME IS NITESH

WITHOUT SPACE.

*ex- MYNAMEISNITESH

Maximum line length and line breaking.

Recommended

Python

```
def function(arg_one, arg_two,  
            arg_three, arg_four):  
    return arg_one
```

• Ex-

Not Recommended

Python

```
from mypkg import example1, \  
    example2, example3
```

• Ex-

Should a line break Before or After A Binary Operator.

- Ex-

Python

```
# Recommended
total = (first_variable
        + second_variable
        - third_variable)
```

- Ex-

Python

```
# Not Recommended
total = (first_variable +
        second_variable -
        third_variable)
```

Example of comments.

block comment.

Anti-practice.

Example

```
#This is a comment  
print("Hello, World!")
```

Best-practice.

Example

```
#This is a comment  
#written in  
#more than just one line  
print("Hello, World!")
```

Inline comments.

Anti-pattern.

Python

```
x = 5 # This is an inline comment
```

Best practice.

Python

```
x = 'John Smith' # Student Name
```

Documentation string comment.

```
"""Return a foobang
```

```
Optional plotz says to  
froblicate the bizbaz first.
```

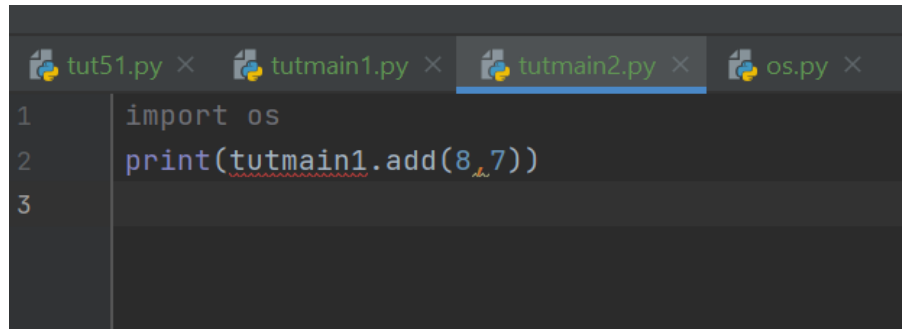
```
"""
```

EXAMPLE OF NAMING CONVENTION

NAMING MODULE WITH HELP OF

PEP8

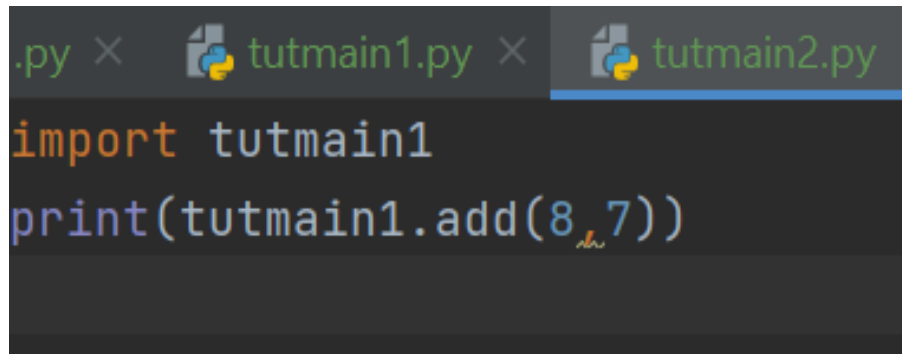
Not recommended



A screenshot of a code editor with four tabs: tut51.py, tutmain1.py, tutmain2.py (selected), and os.py. The code in the selected tab is:

```
1 import os
2 print(tutmain1.add(8,7))
3
```

Recommended



A screenshot of a code editor with three tabs: .py (partially visible), tutmain1.py, and tutmain2.py (selected). The code in the selected tab is:

```
import tutmain1
print(tutmain1.add(8,7))
```


NAMING VARIABLE WITH HELP OF PEP8

Variable:

```
>>> # Not recommended
>>> x = 'John Smith'
>>> y, z = x.split()
>>> print(z, y, sep=', ')
'Smith, John'
```

```
>>>
```

```
>>> # Recommended
>>> name = 'John Smith'
>>> first_name, last_name = name.split()
>>> print(last_name, first_name, sep=', ')
'Smith, John'
```

EXAMPLES OF INDENTATION

❖ METHODS OF WHERE TO PUT CLOSING BRACES:-

Recommended

```
list_of_flowers = [  
    rose,sunflower,  
    marigold,jasmine,  
    tulips,lavender  
]
```

```
list_of_flowers = [  
    rose,sunflower,  
    marigold,jasmine,  
    tulips,lavender  
]
```

❖ Methods for following line breaks

Recommended

```
▶ x=10  
  if (x < 15 and  
      x > 5):  
      #Both the conditions satisfied  
      print(x)
```

```
▶ x=10  
  if (x < 15 and  
      x > 5):  
      print(x)
```

EXAMPLE OF WHITESPACING

1. Adding space when there is more than one operator in a statement.



#recommended

```
b = a**8 + 5
```

```
c = (a+b) * (a-b)
```



#not recommended

```
b = a ** 8 + 5
```

```
c = (a + b) * (a - b)
```

1. Adding space to if statements where there are multiple conditions.

```
#Recommended  
if x>8 and x%2== 0:  
    print('x is larger than 8 and divisible by 2!')
```

```
#not Recommended  
if x > 8 and x % 2 == 0:  
    print('x is larger than 8 and divisible by 2!')
```