11th

DAY

PYTHON

PROJECT

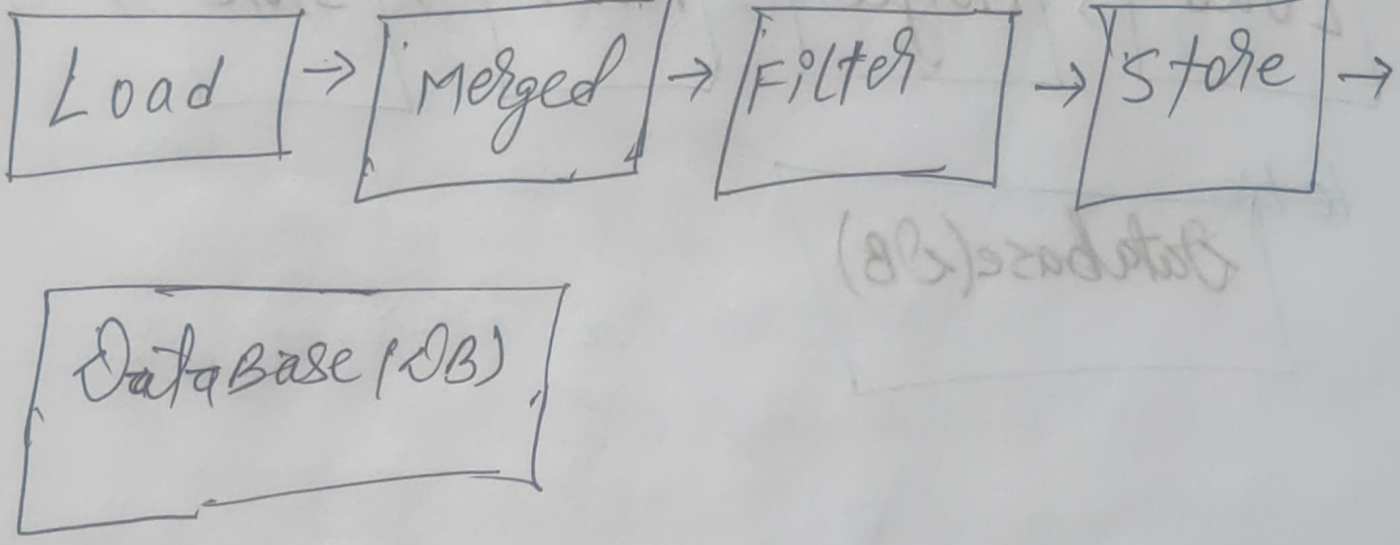
on

LOGGING

&

DEBUGGING

Logging in python



let's 1st create a new file over here with name test.py

my directory name → pythonproject123

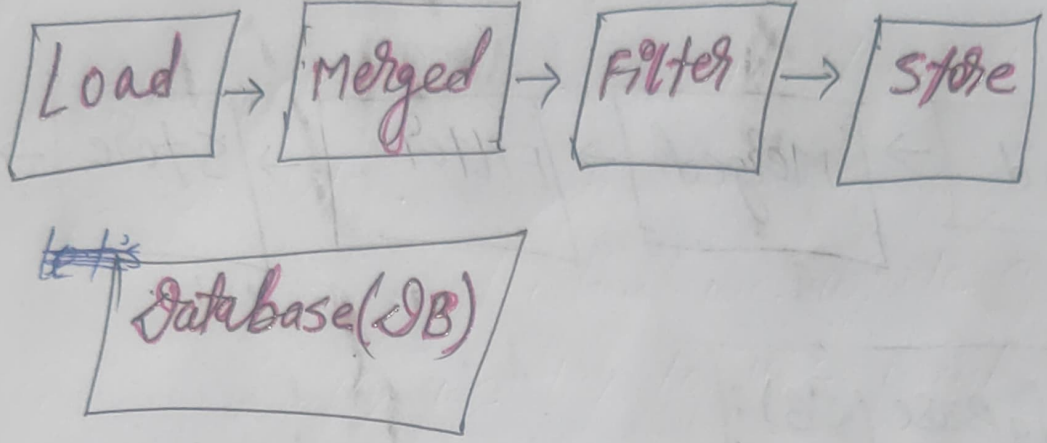
inside this file that I have created main.py

test.py

Hint:

Right click
select file
name it test.py

Let's understand meaning of Logging



Let's suppose,
I am trying to build a project there is situation that, build a very big project, that project, I have multiple steps very 1st step is to Load a data.
2nd step is to Merged some short of a data.
loading data from multiple sources & merged it.
Next, I have some requirement to FILTER some short of a on top of a data.

After that requirement to store these filter data inside some of the Databases (DB) going forward.

These are the steps, I am trying to follow.

when I am trying to load, merged, filter, store ~~not~~ into database (DB).

There is a very high chance that there is possibility that because of glitch in my code or because of some data file which I am trying to redeem, because of name ERror, because of Extension ERror.

my code will stop executing in between there is high possibility.

that case let's suppose there is a person who try to monitor these entire system, how that person get to know, in what step my system has failed or whether my system able to load a data properly or system was able to merged data properly. or filter data properly. & store

properly into Database (DB).

How that person monitoring these entire eco system will be able to find understand that where is a problem. or where is glitch.

In general when we wrote a code, generally we try to use 'print' statement.

To print something in between in Console. we all try to use it.

But, 'print' is not the standard way, so yes, we will be able to print something in a Console, which is run time Console. will be able to print something. & we will be check step by step fine here is a problem or my code has executed 2 line or 3rd line there is a issue.

So we use 'print' 'print' these statement multiple times.

In between a code to just understand that what kind of outcome that ~~I have~~

I am trying to received in between, but whenever we try to built a standard project or whenever we are going to walk with standard project that point of time we never recommend or ~~atte~~ accept a code where person has included a 'print' statement.

we never allow anyone to use print statement in case of real time code.

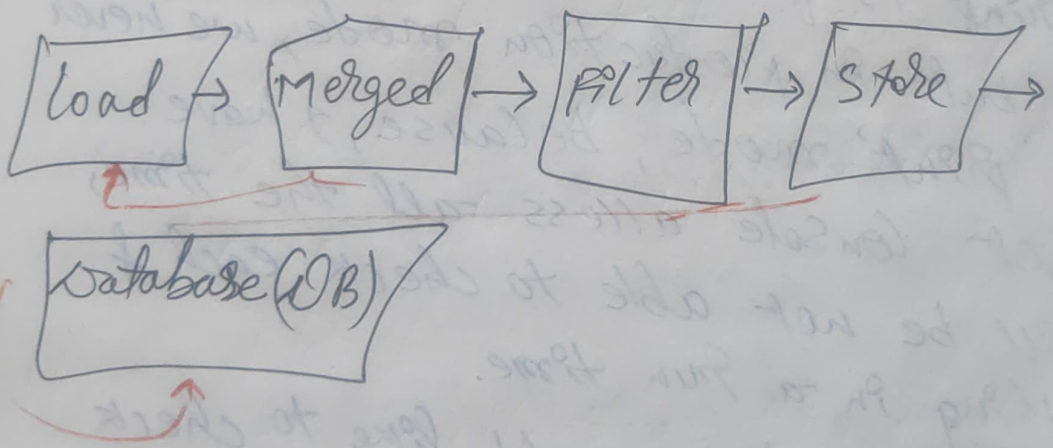
'print' is fine into development mode, but in production mode, we never accept 'print' mode, because, I have not get console access all the time, I will be not able to check each & everything in a run time.

may be I would like to check same thing after 2 days. or after a months what has happened with the system. why these system was not able execute this file.

I would like to investigate my code at that point of time.

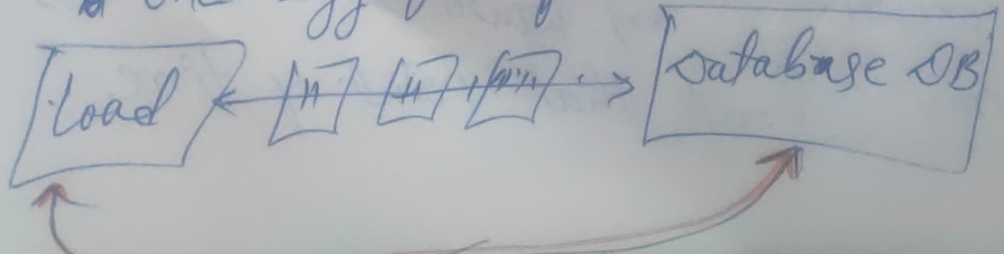
As I said 'print' statement is not going to work.
what is standard procedure, is to use logging.

* In case of file system I can try to create a txt file, I can try to store some short of a data, I can read out some short of a data. all those things we already discussed.



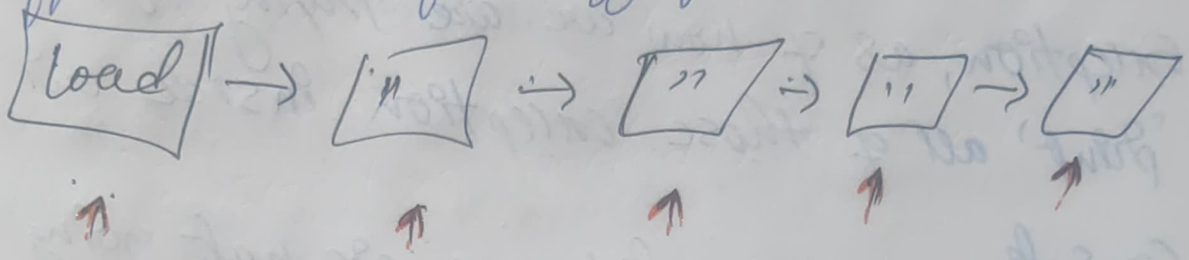
Now, here its extension of same thing.
Here what we try to do is, we try to create a file a logg file.

One Logg file for entire process



or

process wise we can create a different different kind of a logg file over here.



~~so~~ or we can say individual one by one. we can try to create a logg files & then inside that logg file, we try to print.

once load is complete, so once load will start, I can try to log that,

~~ok fine loading is starting.~~

ok fine loading is started.

Then once, I ended upon doing loading. loading ended with this file, this file name.

I can even try just a name, even I can try just time stamp my current system. whatever I would to write I can write, If there is some Error or Exception my code is

going to through.

we already talked about Exception.

If my code is throughing an Exception, as of now we are trying to 'print' all of those exception inside our Console.

But in reality, we are not going to 'print' all of those things inside a Console when I am going to deploy a project in a production environment.

At that point of time, what I will do is that, I will try to insert each & every data or information a Logging file.

It's exactly same thing instead of printing something inside my Console, ~~I am~~ what I am trying to do is, I am just trying to insert the same information inside a file, which is going to store itself in some system. in some hard disk of in some server.

so, that after or even a week or [illegible] months, I would like to initiate in investigation about my entire code. & the process of a data, I will be able to get a Evidence or prof of it, this is where Logging is very very important.

whenever we are going to write a code there are couple of things, we are suppose to keep in mind. that Exception Handling we are suppose to implement.

No one going to ask you, you should use exception Handling or not bydefault its a standard process that exception Handling, we are suppose to use.

Logging is an again standard procedure just like Exception Handling that we are suppose to use. with respect to any line of code that we are going to write. going forward.

that is a exception or that is basically default methdology that we all are going to use. & this is where Logging come to a picture.

* There is no Such different we are learning instead of using print statement, I am saying is that, whatever you are trying to print why we can't log it.

Because print will be available in just run time mode.

what if person is not able to access Console, what if current instance is not working.

In future if I would like to do Some short of a investigation, how I will be able to initiate those short of a investigation.

This is where logging come to a picture.

Inside Logging there are multiple ~~this we~~ things we will be able to find out.

multiple level of logging, we will be able to find out.

part-1- Test.py

*

: project

pythonproject123

main.py

test.py

External libraries.

```
import logging
logging.basicConfig(filename=
    "test.log")
logging.info("this is my
    very first code for
    logging")
```

*

pythonproject123

main.py

test.log

test.py

External libraries.

Let's understand:-

Let's create a one logging file.

Before start.

Come inside test.py & type. This I have already created.

---

* import Logging
  Logging.basicConfig(filename = "test.log")
  Logging.info("this is my very first code for logging")

Before starting, we have to import

* import Logging. Once import Logging, this next line try to Call Logging.ba

* Logging.basicConfig(filename = "test.log")
  Logging.info("this is my very first code for Logging").

Call loggin.basic Configuration, now here try to set a file name, in which file we would like

to store what.

So, filename = ~~test.py~~ "test.log")

So, here we can try to give a file name is = I have created "test.log." file.

I have created.

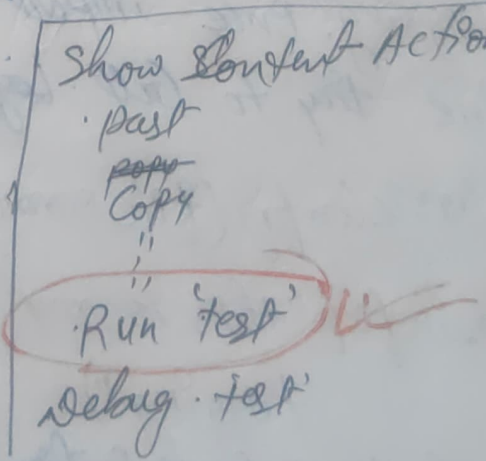
I will keep this things as it.

Then what I can do is, I can dump some short of information.

* logging.info("this is my very first code for logging")

logging.info & I can try to write some short of a information ("this is my very first code for logging")

Now after this 3 one code save it & just right click



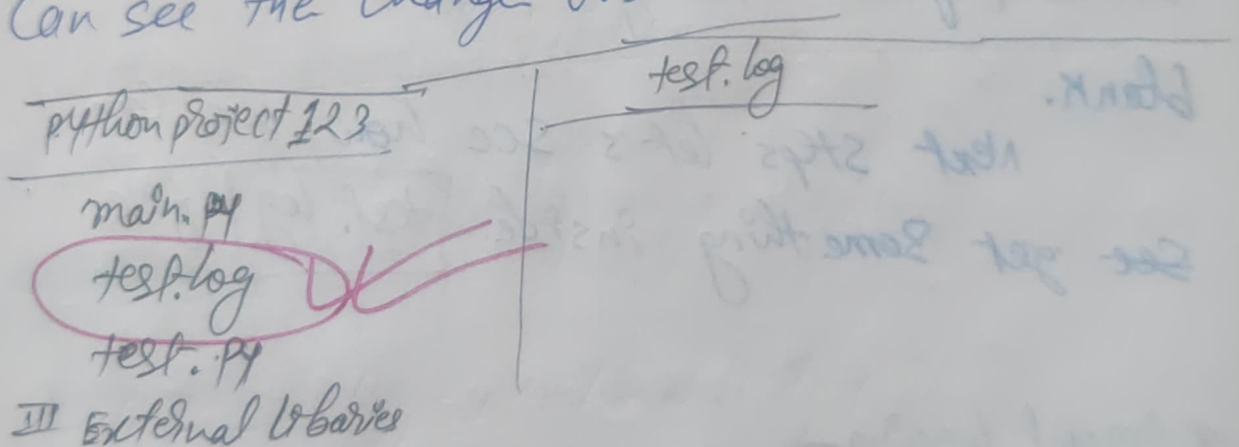
Right click 'Run test' execute it.

we can see down below. This ~~outcome~~ below.

```
C:\Users\Anaconda\envs\pythonproject123\
Process finished with exit code 0
```

Once, we execute that 3 lines code down below we can see this nothing outcome.

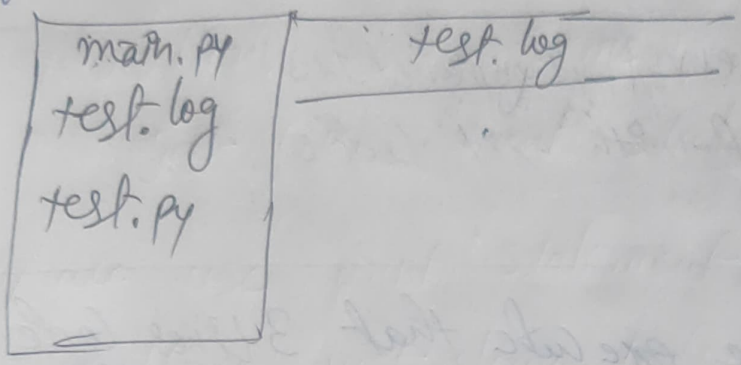
But after execute this code, we can see the change over here.



we can find out that by default inside our project directory, system has created

test.log

test.log file has created, but again if I click on test.log



If we click on test.log inside this we can't find anything inside, its totally blank.

Next steps let's see how we can ~~see~~ get something inside test.log file.

* import logging—

```
logging.basicConfig(filename="test.log", level=
                    logging.INFO)
logging.info("this is my very first code for
              logging")
```

(1)* INFO *

Let's try something that, how we can find something inside these Log file.

I can try to define

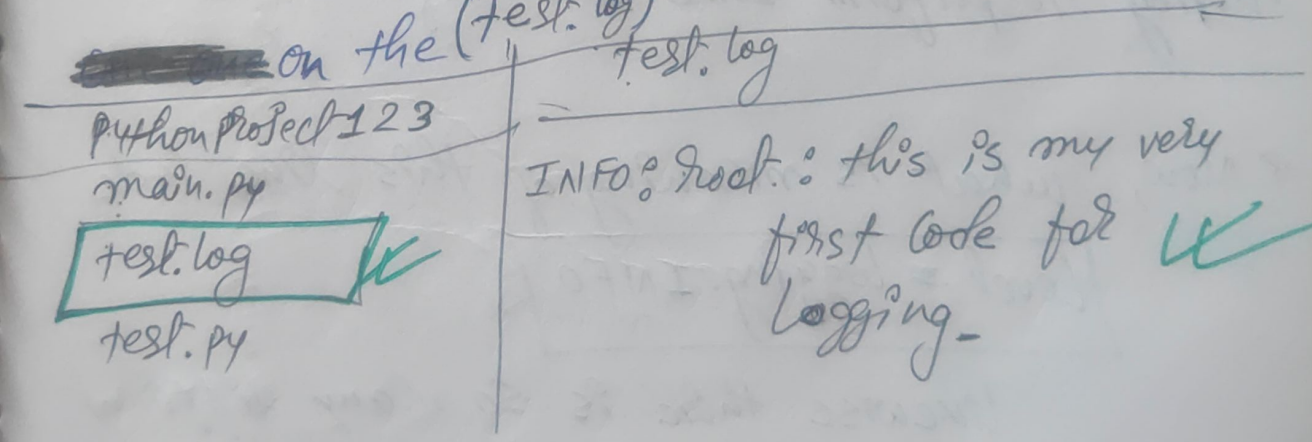
[redacted]

```
Level = logging.INFO
```

If we just this line to test.code & save it right click & execute the 'Run test'

As a outcome we can see not below just see in file section.

Now, after execute, if I will click on the (test.log)



If, I will go inside test.log, we will be able to see, something inside these message we can see

```
INFO: root: this is my very first code for logging.
```

This is something, which we will be able to find out.

Yes, I am able to ~~logg~~ logg, some information inside some of the file.

Just like 'print' statement.

If we use 'print' statement, it will show the outcome, just down below console.

But here what we are trying to do is, I have just created a ~~ff~~ file a log file, which is `test.log` & inside that, I am trying to perform some short of a operation.

* Now, what is meaning of this line code

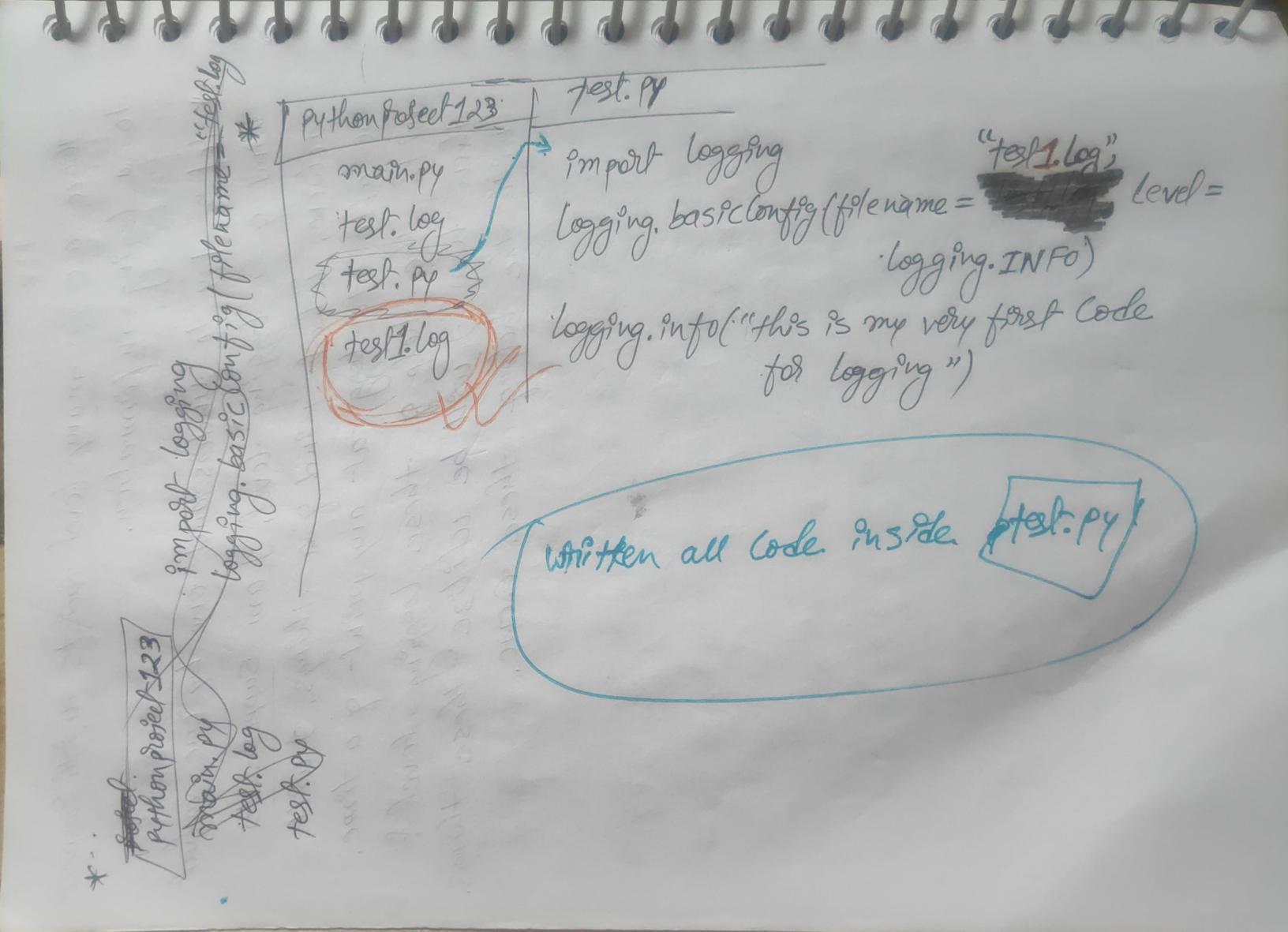
```
Level = logging.INFO
```

Likewise there is so, many different kind of a levels that we have, we have

1. warning level
2. Debug Level
3. ERROR level
4. Info level etc...

There are so many level exists with respect to a loging information.

* let's try to understand one by one what kind of a logging level, I am suppose to achieve, what kind of a things, I will be able to logg at any point of a time, with the help of these logging level & then how I will be utilize these things. with the help of these one.


Python Project 123
main.py
test.log
test.py
test1.log
test.py
import logging
logging.basicConfig(filename = "test1.log", level =
logging.INFO)
logging.info("this is my very first code
for logging")
written all code inside test.py
import logging
logging.basicConfig(filename = "test.log
Python project 123
main.py
test.log
test.py

So, here,

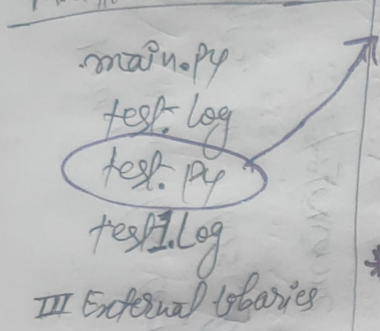
In this code, I have made some changes, in this line of code.

```
filename = "test1.log",
```

Save it right click 'Run test' execute it.

As a outcome we can see over here. It create a new file name as test1.Log As of now its blank.

These all code we are writing inside a test.PY



test.py

```
* import logging
  logging.basicConfig(filename="test1.log", level=logging.INFO)
  logging.info("this is my very first code for logging")
* L = [1,2,3,4,5,6,7,7]
  for i in L:
      if i == 2:
          logging.info(i)
```

```
* L = [1,2,3,4,5,6,7,7]
  for i in L:
      ''
      ''    (L)
```

Let's understand:-

Here what I will do is that, I am trying to create a 'list' inside a 'list' I am [redacted] going to write some short & g a code for this one to pass this entire 'list' [redacted]

test.py

This code inside test.py

```
L = [1,2,3,4,5,6,7,7]
for i in L:
    if i == 2:
        logging.info(i)
```

ctrl+s & right click 'Run test' execute it.

outcome go to check in inside

test1.log

test

```
INFO:root:this is my very first
code for logging
"
INFO:root:2
```

If we go inside after execute file & check inside test1.log we can find out (2) over here.

Note:-

* Instead of doing these `logging.info(i)` what, I could write, previously we are ~~wri~~ writing print statement.

Try to print this one into my Console, but these time, I am not looking for something inside my Console. or ~~below~~ down below as a outcome.

what I am looking for is, I am looking for something inside my file system, inside a file, it is suppose to store, some of the information.

* Let's understand :-

Here, may be ; if we would like check in entire 'List' over here.

* L = [1, 2, 3, 4, 5, 6, 7, 7]

```
for i in L :
    if i == 2 :
        Logging.info(L) ik
```

Here, earlier, I have used (i) now this time using [L] code written inside test.py.

Save it & right click & ['Run test'] once we execute it.

Go to the [test1.log] & we can outcome over here.

test1.log

INFO : root : this is my very ' ' ' —

INFO : ' ' ' " " ' "

test1.log

```
INFO:root:this is my very " " "
INFO:root: " " " " " "
INFO:root:2
INFO:root:this is my very first " "4
INFO:root:[1,2,3,4,5,6,7,7]
```

Yes, we can see we are able to store inside the particular log information.

* Now, if person, other person who is trying to monitor these entire thing, If they will try to check, these information may be after a week or after a year,

Yes, He/she can see inside these test1.log folder they will be able to find out each & every information.

This is the info level log that we are trying to kept inside a particular file.

Note :-

There are 5 kind of logg that, we will be find out :-

* Log Level - (1) INFO
* Log level - (2) Debug
* Log level - (3) warning
* Log Level - (4) ERROR
* Log Level - (5) Critical

[redacted] these are the 5 level of loggs, we will be able to find out inside a python. or other programming language.

At what situation what kind of Log level, I am going to used.

Definition :-

(1) Debug Log :- we all have seen code in the debugging mode, last class, whenever we have to perform some short of a investigation or dignostic on the top of a code. at that point of time, I am trying to use a debug level of log.

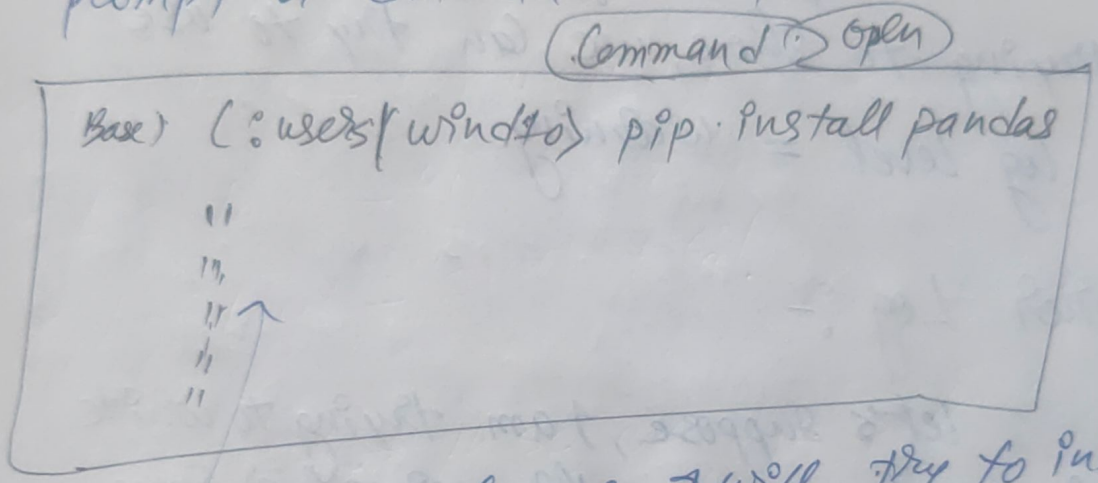
That OK fine this is the debugging code or something that I am trying to do as a debugging in that case whatever information, I have to store may be at that point of time, instead of calling logging INFO info.

I will end up calling debugg level of code.

(2) Warning Log :- Now if I am suppose to give a warning, at any point of time,

For example :-

If I open & run my Anaconda prompt or Command in pc.



Here, whenever I will try to install may be `pip install pandas` some libaries I am trying to store. at these point of time, it will trying to give you some logg information.

Sometime we must have seen that, it trying to give us a warning, Something is deplicated, these Software is going to change, from where we are getting those messages, because these message are store inside warning log.

file which we have created, where we are showing warning log over there everything will be written.

From there we are trying to showcase each & every ~~thing~~ information.

whatever ~~ty~~ type of information we are trying to show, we can try to use ~~log~~ a log level = warning.

(3) Error Log :-

Let's suppose, I am trying to write a code, inbetween something crash or some exception has raised, that point of time, I can call Error.

Instead of showing me an exception try to do one thing, logg these exception, store & save these exception inside a file, so that in ~~future~~ future, when I will do a investigation, I will get to know about that, what kind of Error I have received.

(4) Critical :- Now these is something called as a critical, whenever we have some short of a very important ERROR or very stevair ERROR at that point of time, we can try to call a Logg.

(5) INFO :- So, whatever info, means kind of print we can consider, whatever generic ~~infor~~ information, I have to store may be print operation, I have to store inside ~~any~~ some file at that point of time I will end up using info logg at any point of time.

Note :-

Here, inside python programming, we will be able to find out a numbering as well.

Number System as well.

(1) Debug :- For debug number is or severity level is 10

(2) INFO - For info ; it is 20

(3) warning - For warning ; it is 30

(4) ERROR :- For ERROR, it is 40

(5) Critical :- For critical it is 50.

These are the severity level, these are given to us for hierarchy.

* So, 10 means ; its a 'list' critical one.
20 means a little bit saviour
issue a thing that we are trying to logg.

30 means like a more than 20th one the info that we are trying to store, the kind of warning we would like to give a users.

40 means, yes saviour one.

50 means a very very saviour or critical one. that we are trying to store.

These are the number system allocated to logging file or information at any point of time, which we are going to use in mean time.

33.20

pythonproject123

- main.py
- test.log
- test.py
- test1.log

III External libraries

test.py

```
import logging
logging.basicConfig(filename="test1.log", level=logging.INFO)
logging.info("this is my very first code for logging")
L=[1,2,3,4,5,6,7,7]
for i in L:
    if i==2:
        logging.info(L)
```

1st part

2nd part

Logging.warning("this is a warning for a user that they have found out 2 in list")

Next line code Execute if & check on test1.log

New message created with warning.

*(2) WARNING *

Let's understand :-

Now, let's suppose, I have already given log level = info.

Now, test.py we will with Calling

* Logging.warning("this will try to load a warning message")

This one Logging.warning inside this we have written some message. ("this will " " , " " message")

Now right click & execute it.

test1.log

once, I execute this line code & go to test1.log & check that whether,

test1.log | Output

INFO : root : this is my very fesst code for logging.

INFO : root : " " " " "

INFO : "

INFO :

INFO :

INFO :

WARNING : root : this will try to load a warning message.

INFO : root : [1, 2, 3, 4, 5, 6, 7, 7]

once, we execute it, if we come & see into test1.log, we can see that it has logged some information, with tag 'WARNING'.

By default, I am just trying to log something, trying to store something inside my file we can see from top all INFO, INFO ....

But here its trying to log a message whatever we want as a 'warning'

At any point of time, if I would like to filter, filter out all the info, filter out all the warning, filter out all the debug. filter out all the Errors.

yes, we are able to filter out at any point of a time.

2nd part

Now again come to the test.py give some warning: that logging.warning("this is a warning for user that they have found out 2 in list") right click Run test execute it.

Go & check to the test1.log

test1.log

INFO:
warning: root: this will try to .....
INFO: root: [1, 2, 3, 4, 5, 6, 9, 7]
warning: this is a warning for a user that they have found out 2 in list.

we can see that, it has logged the information with the 'warning': what information or message, I would like to ~~persistence~~ ~~give~~ persisted, it is just persisted all the information in a particular file.

* (3) Error *

Project
pythonproject-123

main.py
test.log
~~test.py~~
test1.log

test.py

```
import logging
    ''
    ''
    ''
    '',
logging.warning("  .....  ")
logging.error("this is a error
        message ~~from~~ system")

L = [1,2,3,4,5,6,7,7]
for i in L:
    if i == 2:
    logging. ....
    logging.warning("this ... ...
                ... 2 ... in list)
```

Right click & execute it

Error :-

Let's suppose, I would like logg a Error.

* Logging.error("this is a error message from system")

This line code added.
Now execute it

Go to Check test1.log

Test1.log

outcome

INFO : root : this is my very ''
INFO : root :
''
''
warning :
INFO :
warning ''
ERROR :- root : this is a error message from System
INFO : root
warning : root

we can see a message with ERROR.

~~when we~~

* when we write 'print' statement, its just like a generic statement, we are trying to write.

But in these case what we are trying to do is, we are trying ~~seg~~ segregate, every kind of messages,

If this will happend that case it is going to be an ERROR.

If this is not going to happend I should give a warning.

If this happend, I should give an information info.

But end of the day the message we are trying to store, Nothing much.

we ~~Also~~ are just trying to logg some short of a information at these point of time.

Shutdown

Project
pythonproject123

test.py

```
import logging
import logging
"
"
"
if i == 2:
    logging.info(L)
    logging.warning("this ...
                    2 in list")

logging.shutdown()
```

let's suppose,
If I would like shutdown the logging, what I can do is

* ~~Logging : shout~~ logging.shutdown() I can call & right click execute it.

Once we execute it
go to check

test1.log

we can see over here is that,
after we execute `logging.shutdown()`
. we can see that it stop everything
we can see here nothing executed.

* whatever, I will execute after these
`logging.shutdown()` nothing will work
because, I have already given this
command.