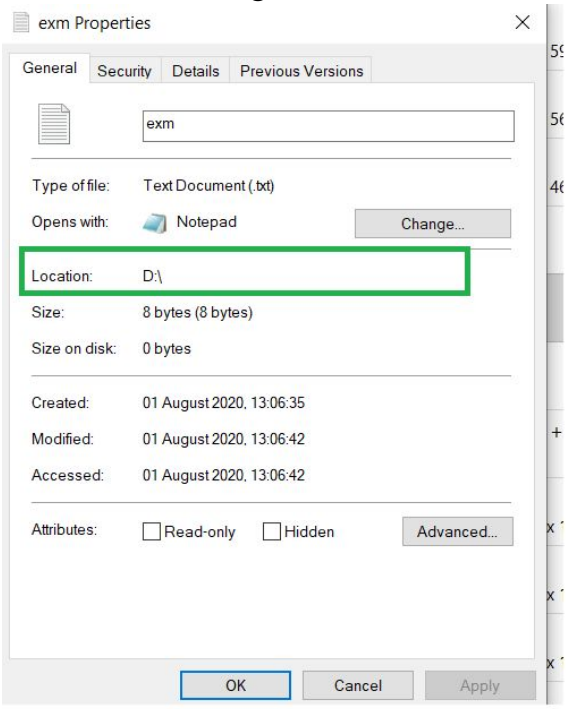
Project x1: Manipulating Files and folders.

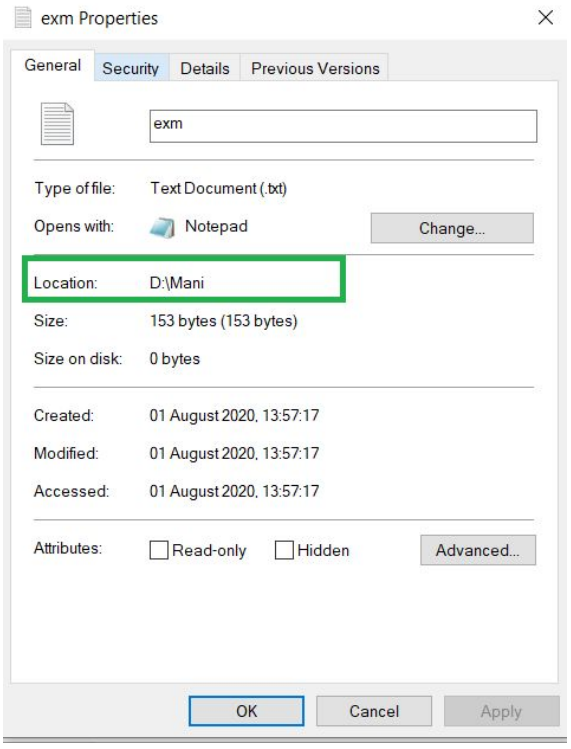
Moving a file or folder

Here we will be seeing how we can move a file or folder from one location to the other. But first let's see what happens when we try to move a file from one place to another. You will see, when you move a file, it simply **changes the path/location of the file**.

For example, right now the exm.txt file is in my D drive. If I open the properties of this file this is looking like this:



Look at the path here. And now when i move it to a folder named "Mani' in my d drive, then it will look like this:



That simply means, if you move a file from one place to another, then the location of the file changes. Now when you have to work with paths and location of files in the device then the **module OS** is going to help you.

But before moving ahead, let's see **what is a module?**
A **module** It is a **python file** with a set **of predefined functions, classes & variables.** This means some usable functions, classes etc are already written in some python file and these files have been made available to be used publicly. These python files are called modules.

Just like functions, modules are also of two types. One of which is a built-in module. Which you directly import and use. Now when you import any module you simply make it available to your program. After which you can use any of the functions or classes of the module. But the modules which aren't built-in, we will have to install them also.

Coming back to **os,** it is a built-in module. This means when you downloaded python, this module got downloaded with that and now to use it, you don't need to install it, simply import and use. Along with os

we will need another module named **shutil.** Shutil is another built module of python which is used for different operations on files and collections of files such as moving, copying etc.

Now to import these two modules, simply run the following code:

```
import os, shutil
```

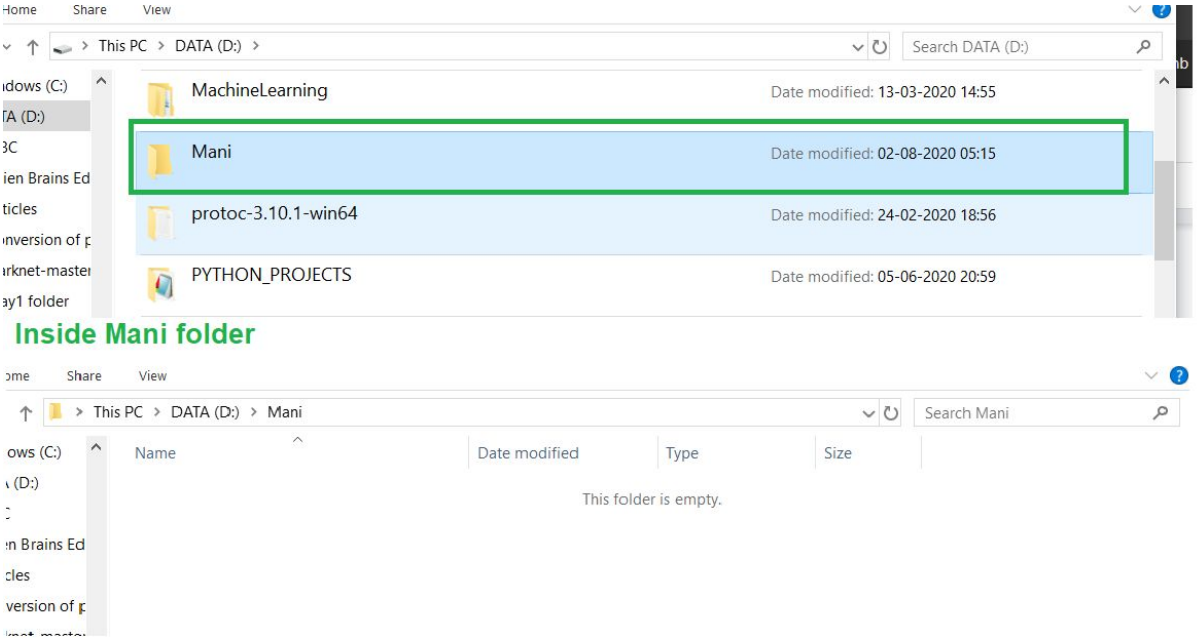
Now the function which is going to be helping you for moving a file is **shutil.move().** This function moves a file from one location to another. The syntax for this function is **shutil.move("source_path", destination_path")**. Here the file will be moved from the source path to the destination path. And this will be done by altering the path of the file as we have seen earlier.

Now let's say, you want to move the exm.txt file from D drive to a new folder named Mani in D drive. So you will have to create this new folder. For that we will be using **os.mkdir()**. This creates a new directory of the path given as input.

So to create a new folder named mani in the D drive, we will have to run the following code.

```
os.mkdir("D:\\Mani\\")
```

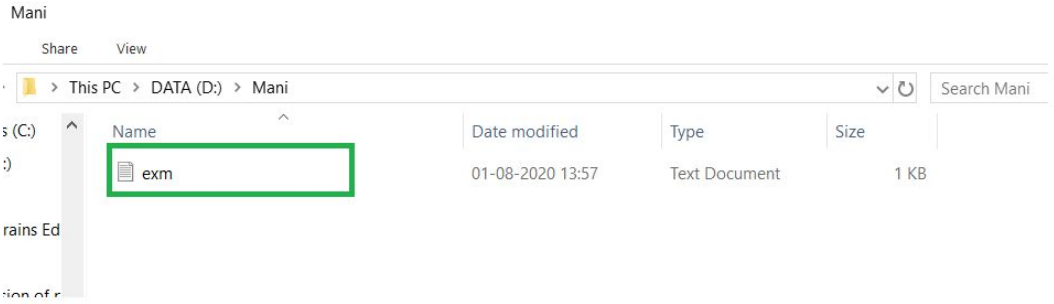
After this you will be able to see a new folder(like in the picture) in your D drive which is empty.



Now to move the exm.txt file to the Mani folder, just run the following line of code.

```
shutil.move("D:\\exm.txt","D:\\Mani\\exm.txt")
```

Once you do this, you will have exm.txt in your Mani folder which was earlier empty. Here is what your Mani folder would look like.



Let's take another case where you want to move multiple files. That means here you want to do one task again and again.The above line of code will be executed again and again. So here we will be using **loops.**

Before going deeper into this, let's understand **what a loop is.** Loops are used to **repeat the same/similar code** a number of times. There are various kinds of loops. One of them is **for loop.**

The **syntax** for for loop is like this:

```
for i in sequence:
    do all this task.
```

Here ‘i’ is an iterator. This iterator is nothing but a variable whose value keeps changing. The iterator travels/iterates over the sequence given here. For each value in the sequence ‘i’ will be holding that value. Now this loop starts with the first value in the list/sequence. For this value, the code inside the loop runs. We put those lines of code inside the loop which we want to repeat again and again. Once for the first value, when the code will get completed, the loop will reach to the beginning and change the value of ‘i’. ‘i’ will hold the 2nd value of the sequence now. And the whole stuff will work for this value and nextly ‘i’ will hold the third value. This will continue till the end of the sequence.

For example:

If i run this code:

```
l=['a','b','c','d']
for i in l:
    print(l)
    print("**************************")
```

The output will be:

```
a
**************************
b
**************************
c
**************************
d
**************************
```

Now coming back to our problem where we have to move multiple files, first of all, we will have to make a list of all the files whom we want to move. Then we will iterate over them using a for loop. And for each of the files we will be moving the file which is nothing but the iterator.

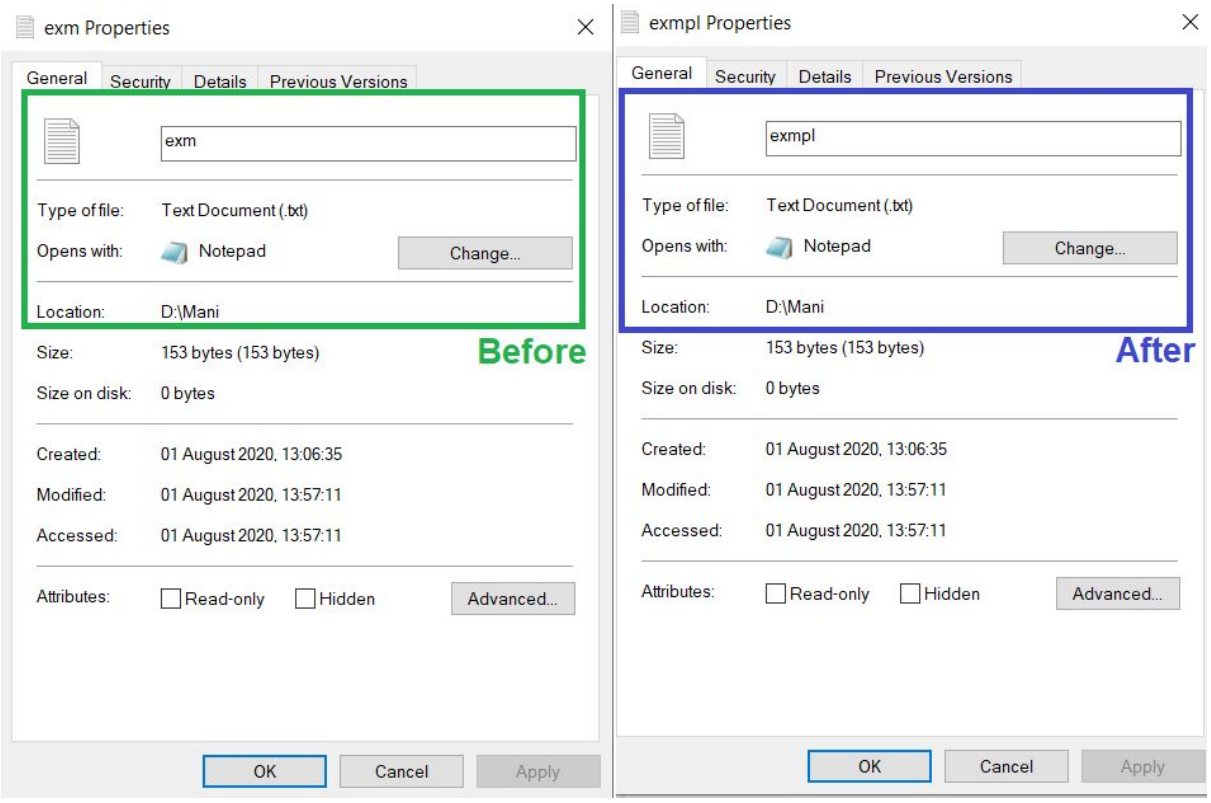
The line of code to do this is:

```
file_list=
["D:\\4th Semester Result.pdf","D:\\5th Semester Result.pdf",
"D:\\6th Semester Result.pdf"]
for file in file_list:
```

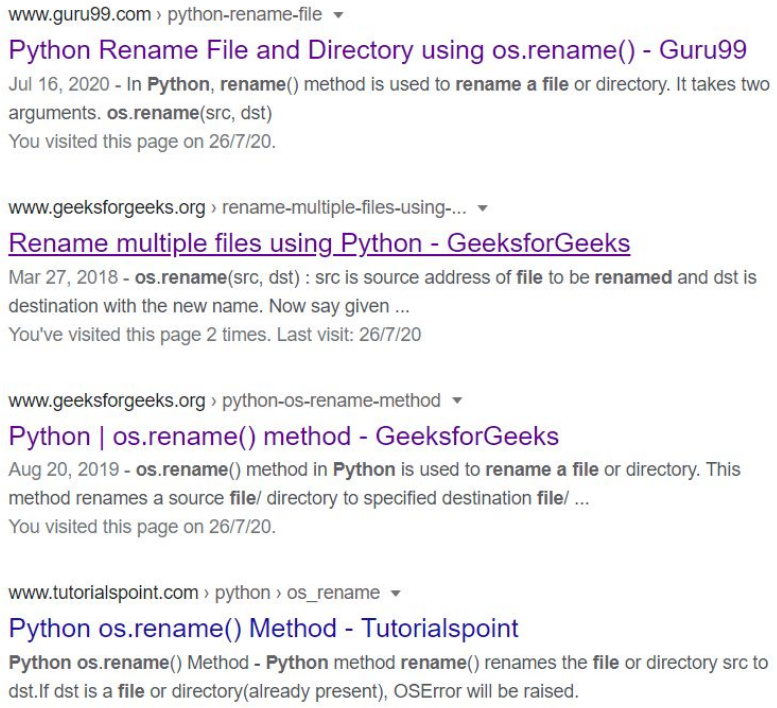
```
shutil.copy(file,"D:\\Mani\\")
```

Rename a file or folder

When you rename a file, you simply change it's path. For example, when I renamed the file from exm.txt to example.tt here is the change of path that took place.



So again something where we have to mess with the path of a file. Definitely we will be working with the module **os.** Now you know you have to use the module os but let's say you don't know how to use the module then in this scenario the best option with you is to search for it over the internet. If i search ***how to rename a file in python with os,*** then this is the result that i got.



Now on clicking on any one of the results above, I will get to know that I can use the function **os.rename().** The function os.rename() changes the name of the file or folder from the old to a new one.
The **syntax** to use this function is

os.rename(“old_path_of_the_file”,”new_path_of_the_file”)

Now to rename exm.txt to example.txt, I will be running the following code.

```
os.rename(“D:\\Mani\\exm.txt”,”D:\\Mani\\exmpl.txt”)
```

To make things clearer let’s take another use case here **use case 2**, let’s say I have two files in the D drive, 2nd Sem_res.jpeg and 3rd Sem_res.jpeg. I want to rename them to 2nd Semester Result.jpeg and 3rd Semester Result.jpeg respectively. Here we need to make the Sem_res.jpeg to Semester Result.jpeg. But also we will have to retrieve the 1st part of the file name from the file itself. What I mean is we will have to convert the path from D:\\2nd Sem_res.jpeg to D:\\2nd Semester Result.jpeg and same with the second file. Here the part ‘D:\\2nd’ is to be retrieved from the name of the file itself. To do this we will have to

take help from **string splitting.** So now let's first understand what is string splitting.

Splitting is a process of **splitting a string into a list of elements**. The **syntax** for string splitting is str.split("separator").This will split the string "str" across the separator.

For an example, if we apply the splitting to the string str likke below.

str= "I live in India."
str.split('i')
Then the output will be this: [" I", "ve ", "n ", "nd", "a."]

This also removes the separator 'i'.

Now coming back to use case 2. Here we want to repeat the task of os.remove more than one time so we will be enclosing the input files into a list and run a for loop over the list of files. Each time inside the loop we will be calling os.remove. But we saw that the new path has to be retrieved dynamically. I.e. when the code is running and for this we will be taking the help of string splitting.

Let's try to understand this first for a single file.

For the 1st file, the path is D:\\2nd Sem_res.jpeg and we want to change it to D:\\2nd Semester Result.jpeg. Now there is a space in D:\\2nd Sem_res.jpeg which is dividing the path in 2 parts. So if we split this path across a space so we will be having two elements enclosed in a list(say l) [" D:\\2nd" "Sem_res.jpeg"]. From here I just want to take the first element and replace the second element with Semester Result.jpeg. We can do this by using list indexing. First element is nothing but l[0] which returns D:\\2nd. Now if somehow we can add Semester Result.jpeg to the end of l[0], my task is done. For this we will have to take the help of **string concatenation.**

Let's see what string concatenation is. String concatenation is a method of **connecting two or more strings together**.

For example, if we have 2 strings str1 and str2 as below:

str1=”hello”
str2=”world”

Now to concatenate them i need to do following step,

str3=str1+str2

This will make the 3rd string str3 as
str3=helloworld.

Now coming to my problem, if I simply write l[0] +”Semester Result.jpeg” I will get my new path as D:\\2nd Semester Result.jpeg. Also with the path D:\\3rd Sem_res.jpeg we will get D:\\2nd Semester Result.jpeg. Problem solved. Once I have got the new path, now I simply rename the files using os.rename(). We just need to do all these things inside a loop.

Now let me just accumulate everything and write the code for you.
To do the use case 2, run the following code.

```
re_files=["D:\\2nd Sem_res.jpeg","D:\\3rd Sem_res.jpeg"]
for i in re_files:
    j=i.split(" ")#splitting across a space
    new_path=j[0]+' Semester Result.jpeg'
    os.rename(i,new_path)
```

Once the code has been executed, your files will be renamed.

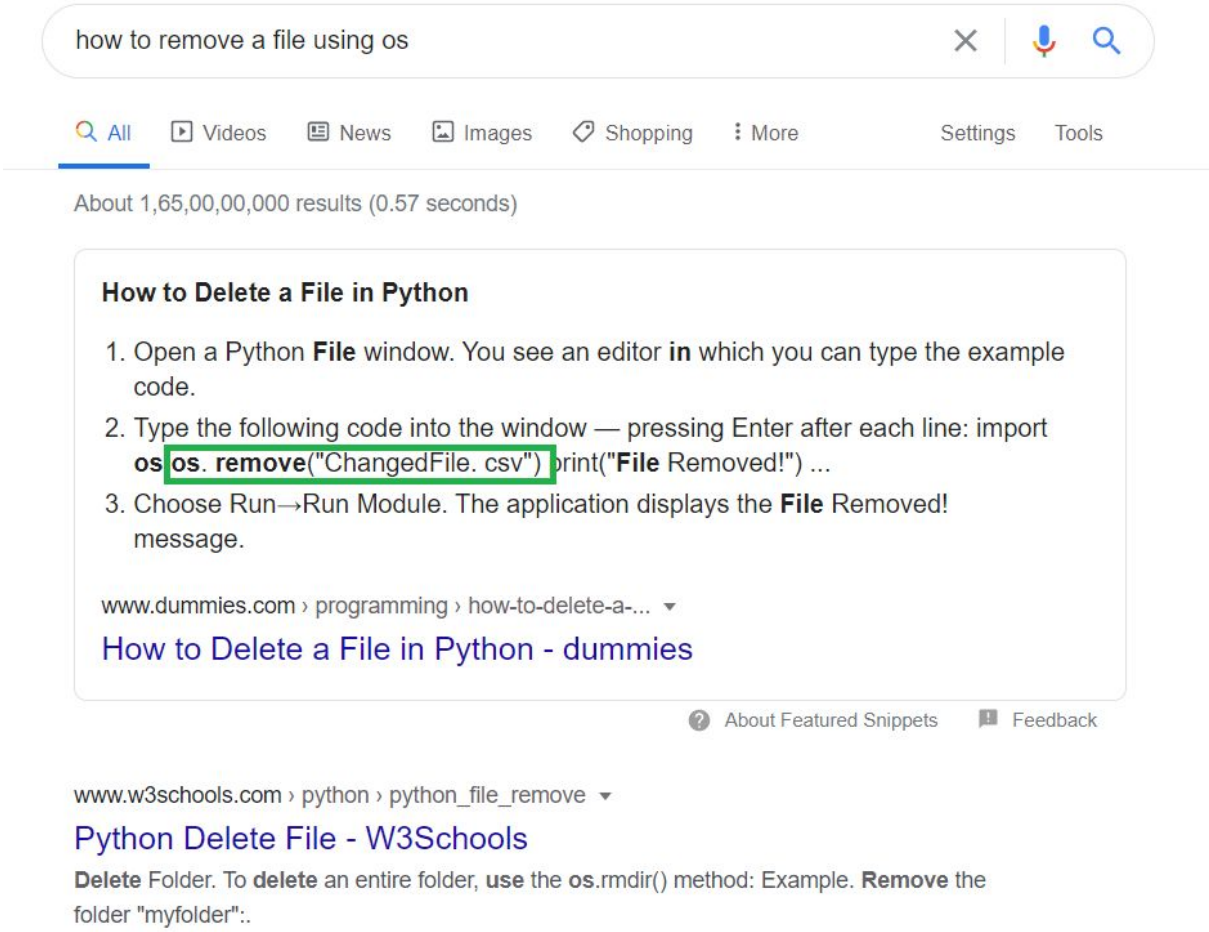
BONUS POINT: Yes you’re thinking right, os.rename() can also be used to move files and folders from one location to the other. Just replaced the source_path with old_path and destination_path with new_path.

Removing a file

To remove a file, we simply need to do one thing, make the path of the existing file free and available for other files. Again something related to path. Which means we will again have to use the module **os.** Now again if you have reached to this conclusion, the next step for you is to do a

search over the internet for *how to remove a file using os.*

I am sharing my search result here.



That means we have to use **os.remove().** This function removes the given file from the device. But it doesn't send it to the recycle bin. It just removes it.

The syntax for os.remove is as followed.

> **os.remove(path_of_the_file_to_be_removed):** removes the mentioned file.

So to remove the file exmpl.txt, run the following line of code.

```
os.remove("D:\\Mani\\exmpl.txt")
```

After running this code, the file will no longer be in the device.