# Modules:

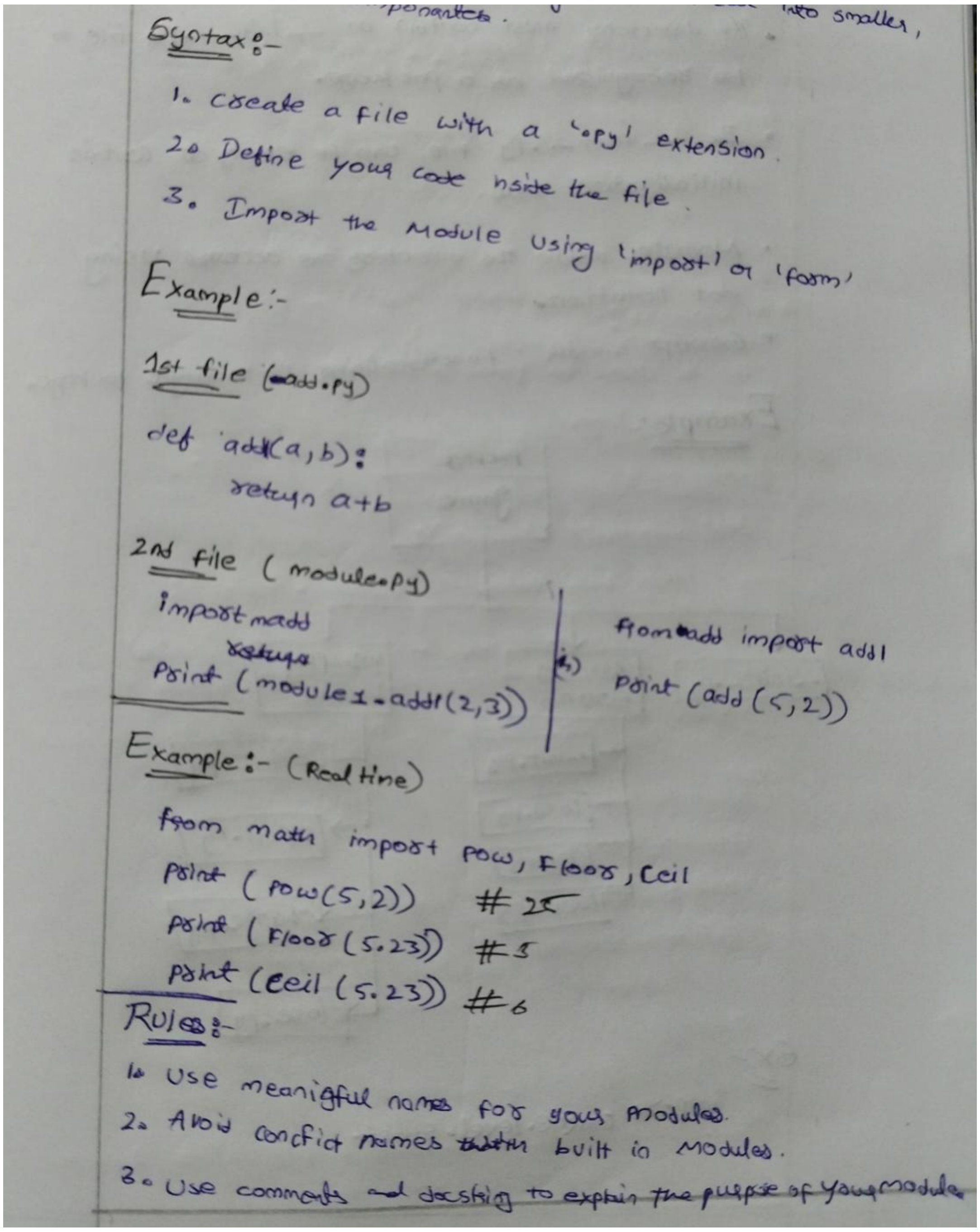
**Python Module** is a file that contains built-in functions, classes,**its** and variables. There are many **Python modules**, each with its specific work.

In this article, we will cover all about Python modules, such as How to create our own simple module, Import Python modules, From statements in Python, we can use the alias to rename the module, etc.

## What is Python Module

A Python module is a file containing Python definitions and statements. A module can define functions, classes, and variables. A module can also include runnable code.

Grouping related code into a module makes the code easier to understand and use. It also makes the code logically organized.



ponantes. into smaller,

Syntax:-

1. Create a file with a '.py' extension.
2. Define your code inside the file.
3. Import the Module using 'import' or 'form'

Example:-

1st file (add.py)

```
def add(a,b):
    return a+b
```

2nd file (module.py)

```
import madd
       ~~return~~
Print (module1.addl(2,3))
```

(ii)

```
from add import addl
Print (add (5,2))
```

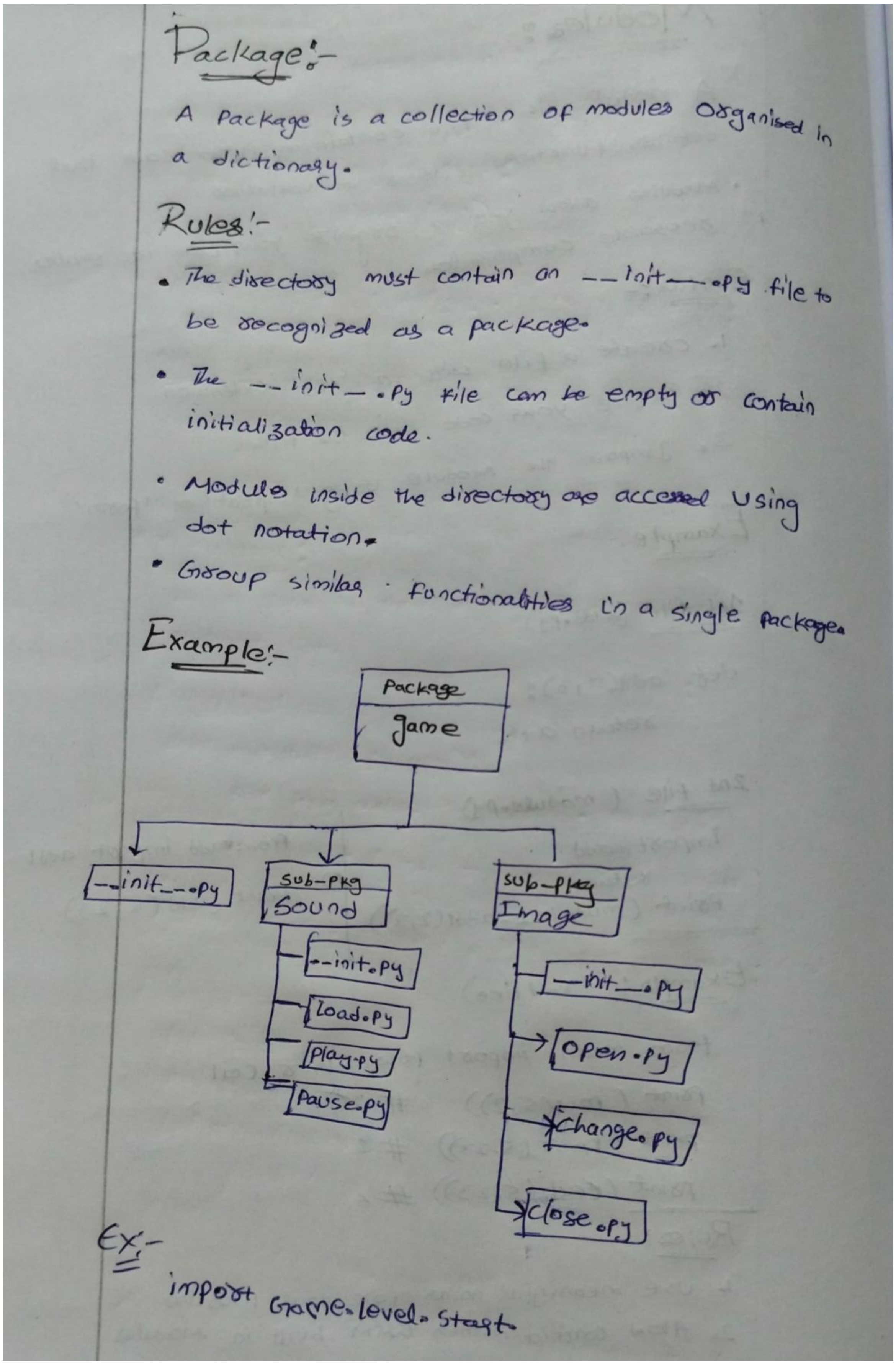
Example:- (Real time)

```
from math import pow, Floor, Ceil
Print (pow(5,2))       # 25
Print (Floor (5.23))   # 5
Print (Ceil (5.23))    # 6
```

Rules:-

1. Use meaningful names for your Modules.
2. Avoid conflict names ~~that~~ built in Modules.
3. Use comments and docstring to explain the purpose of your module

## Packages:



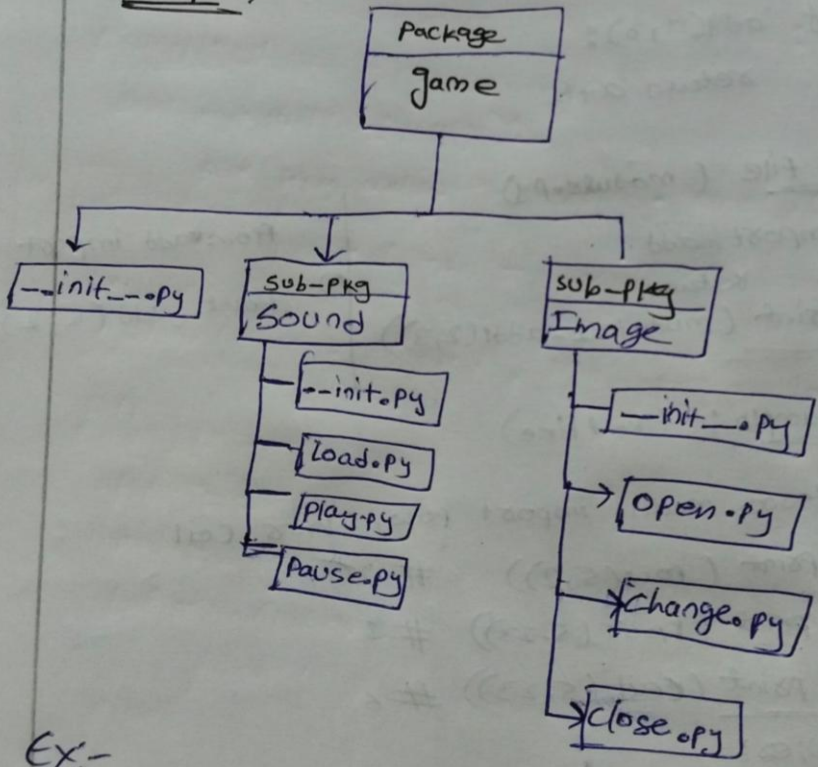
Package:-

A Package is a collection of modules Organised in a dictionary.

Rules:-

- The directory must contain an __init__.py file to be recognized as a package.
- The __init__.py file can be empty or contain initialization code.
- Modules inside the directory are accessed using dot notation.
- Group similar functionalities in a single package.

Example:-



Ex:-

import Game.level.Start

Example:- Startwith() method functionality

```
S = "Python is COOL"
Char = "py"
def start_with(s, char):
    temp = " "
    for i in range(len(char)):
        temp += s[i]

    if temp == char:
        return True
    else:
        return False
Print(Start_with(s, char))
```

Special Symbols Used for Passing arguments:-

- *args (Non.keyword Argument)
- ** kwargs (keyword Arguments)

*args:- it is Used to pass a non-keyworded, variable-length argument list. (to Acess Multiple values)

Example:1:-

```
def myFun(*argv):
    for arg in argv:
        Print(arg)
myFun('Hello', 'Welocme', 'to', 'GFG').
```

O/P:-

```
Hello
welcome
to
GFG
```

Example-2:-

```
def myFun(arg1, *argv):
        print("1st Arg:", arg1)
        for arg in argv:
            print(arg)

myFun('Hello', 'welcome', 'to', 'GFG')
```

O/P:-

```
1st Arg: Hello
welcome
to
GFG
```

** kwargs:-

It is used to pass a keyworded, variable-length argument list.

Example-1:-

```
def myFun(**kwargs):
    for key, value in kwargs.items():
        print("%s == %s" . (key, value))
myFun(First='Greek', mid='For', last='Geeks')
```

O/P:-

```
First == Geeks
mid == for
last == Geeks.
```