

# FINAL YEAR INTERNSHIP REPORT

Submitted in partial fulfillment of the award of the degree  
of  
**BACHELOR OF TECHNOLOGY**  
In  
**INSTRUMENTATION AND ELECTRONICS ENGINEERING**



**Submitted by**

ANKITA JENA : 1701106046

SWADESH KUMAR NATH : 1701106028

Under the guidance of

**Mohit Varu**

GIRLSSCRIPT TECHNOLOGIES PRIVATE LIMITED

**DEPARTMENT OF INSTRUMENTATION AND ELECTRONICS ENGINEERING  
COLLEGE OF ENGINEERING AND TECHNOLOGY, BHUBANESWAR  
BIJU PATNAIK UNIVERSITY OF TECHNOLOGY, ROURKELA-769015, ODISHA,  
INDIA, 2017-2021**



## DECLARATION

We, the undersigned, declared that the project entitled “**BRAIN HEMORRHAGE ESTIMATION WITH PYTHON**” being submitted in partial fulfillment for the award of Bachelor of Technology Degree, Electronics and Instrumentation Engineering, affiliated to Biju Pattnaik of University of Technology, is the work carried out by us.

**ANKITA JENA** :

**SWADESH KUMAR NATH** :



## CERTIFICATE

This is to certify that the project entitled **“BRAIN HEMORRHAGE ESTIMATION USING PYTHON”** submitted by the following students for the award of the degree of Bachelor of Technology in the Department of Electronics and Instrumentation Engineering is a bonafide record of work carried out by them under my supervision and guidance. In my opinion the project report is in the standard of fulfilling all the requirements for the award of the degree of Bachelor of Technology in Electronics and Instrumentation Engineering.

**1. SWADESH KUMAR NATH:**

**2. ANKITA JENA:**



**Supervisor**

**Mohit Varu**

**Managing Director,**

**GirlScript Technologies Pvt. Ltd.**

**Email: mohit@girlscript.tech**

**HOD**

**(E&I dept)**

## **ACKNOWLEDGEMENT**

First of all, we would like to thank the Almighty whose blessings bestowed upon us has made us capable of handling such nature of work with sheer delicacy and responsibility. We also thank **Mr. Naresh Chandra Naik**, Asst. Professor, Instrumentation and Electronics Engineering for his continued drive for better quality in everything that happens at CET. This report is a dedicated towards that greater goal.

We are also thankful to **Dr. Sribatsa Behera** and all our Professors without whose guidance we could not have come up with this project.

Last but not the least, we offer our acknowledgement to our fellow friends and all those who have contributed in development of this report into something worth to be remembered.

## **ABSTRACT**

Brain hemorrhage is a serious category of head injury that can have a fatal impact on brain function and performance. But sometimes the identification of cerebral hemorrhage cannot be known immediately. So far, the identification of cerebral hemorrhage is done through CT Scan image observation that requires special skills. Therefore, we need a certain method that can segment the CT Scan image quickly and automated. The goal is to obtain the image segmentation of brain bleeding more quickly and accurately. So patients with cerebral hemorrhage can immediately obtain medical treatment in accordance with the needs. The preprocessing process of CT Scan image starts from the preprocessing phase of the CT Scan image using color filtering, erosion and dilation methods. This stage is done to clarify the cerebral hemorrhage and eliminate the noise contained in the image. Then performed watershed and cropping segmentation to separate the skull bones of the skull with brain tissue. The next step is to improve the image quality using median filtering. Then the image is again segmented using the threshold method to separate the image of cerebral hemorrhage as the observed object. Last performed the calculation of area and volume percentage of bleeding in the brain.

## **CONTENTS**

<b>SL.NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
1.	List of Figures	...
2.	Introduction	1
3.	Topics covered during internship	2
4.	Basic principle of CT scan	37
5.	Proposed Solution	38
6.	Head CT- Scan Image	39
7.	Color Filtering	39
8.	Erode	39
9.	Dilate	40
10.	Watershed	40
11.	Crop	41
12.	Median Filtering	41
13.	Threshold Binary	41
14.	Count Pixel of Brain Area & Brain Hemorrhage Area	42
15.	Calculating Volume and Volume Percentage	42
16.	Experimental Code	43
17.	Experimental Result	45
18.	Conclusion	46
19.	References	47

## **LIST OF FIGURES**

<b>FIG.NO.</b>	<b>TITLE</b>	<b>PAGE NO.</b>
1.	Modern CT Scanner	6
2.	Tomogram slices with thickness of 5 and 10 mm	7
3.	Schematic diagram of the algorithm	8
4.	The head image in axial plane	9
5.	Result of Color Filtering	10
6.	Result of Erode Operations	10
7.	Result of Dilate Operation	11
8.	Result of Crop Operations	12
9.	Result of Median Filtering	12
10.	Output image of Threshold Binary: (a)Brain (b)Brain Hemorrhage	12
11.	Result of count pixel of brain area and brain hemorrhage area	13

## **INTRODUCTION**

Brain injury is one of many dangers that cause death in humans. Of all the cases of head injuries according to data in the United States last about 49% caused by motorcycle accidents and fall is the second common cause. Head injuries are most commonly found at the age of 15 to 24 years, and twice as large in men as compared to women. The medical process uniquely helps doctors and health professionals to perform special therapies based on the consequences of the patient's pathology condition.



**Fig.1. Modern CT Scanner [5]**

There are many terms used to describe or classify patients with head injuries. In the last year we use the terms "open" and "closed", as well as coup and counter coup. However, the term will be difficult to describe the degree of severity of head injury. The current head injury weights are defined based on the Gaslow coma scale. The terms mil, moderate, and severe head injury are useful in relation to the assessment of parameters for therapy and outcome along the treatment continuum. However, there should be no assumption that mild head injury will result in mild problems or no problems in the patient.



## TOPICS COVERED DURING INTERNSHIP

### WEEK 1 -

DAY AND DATE	TOPICS COVERED
DAY-1(08.02.2021)	<ul style="list-style-type: none"> <li>• Python Installation</li> <li>• Features of Python</li> <li>• Print Statement</li> <li>• Variables and Memory</li> <li>• Datatypes</li> <li>• Typecasting</li> <li>• Arithmetic Operations in Python</li> </ul>
DAY-2(09.02.2021)	<ul style="list-style-type: none"> <li>• Day 1 Problems Discussions</li> <li>• Arithmetic Operations(Continued)</li> <li>• UserInputs</li> <li>• String</li> <li>• EscapeSequence</li> <li>• Indexing &amp; Slicing inPython</li> <li>• String Formatting &amp;Operations</li> <li>• Sample Program to print a GreetingsMessage</li> </ul>
DAY-3(10.02.2021)	<ul style="list-style-type: none"> <li>• Day 2 Sample Greetings ProblemDiscussion</li> <li>• Conditionals in Python</li> <li>• ifstatement</li> <li>• TernaryOperator</li> <li>• LogicalOperators</li> <li>• BooleanAlgebra</li> <li>• in and passOperators</li> <li>• For Loops(Intuition)</li> <li>• PythonLists</li> </ul>
DAY-4(11.02.2021)	<ul style="list-style-type: none"> <li>• Lists</li> <li>• Operations on Lists</li> <li>• Tuples</li> </ul>
DAY-5(12.02.2021)	<ul style="list-style-type: none"> <li>• Dictionaries</li> <li>• Sets</li> <li>• Operations on Dictionaries</li> </ul>

## **Brief Description:**

Python is an interpreted, object oriented high level language which was developed in 1980 by Guido van Rossum at the National Research Institute for Mathematics and Computer Science in the Netherlands as a successor of ABC language capable of exception handling and interfacing.

### **Why Python?**

- High Level Programming Language.
- We don't need to know about its architecture to code.
- English like Language.
- We can use it anywhere ranging from Data Scienceto Development to Scraping to testing etc.
- Packages and Methods Everywhere and forEverything.
- Free and Open Source, Portable.

### **print() Statement:**

def: The print() function prints the specified message to the screen, or other standard output device.

Basic Syntax :

print(object(s), sep = separator, end = end)

### **Variables and Memory:**

def: Variables are containers for storing data values such as texts and numbers etc.

- Unlike other programming languages, Python has no command for declaring a variable.
- A variable is created the moment you first assign a value to it.
- Variables do not need to be declared with any particular type, and can even change type after they have been set. That's why Python is called as Dynamically Typed Language.

### **Variables Naming Convention:**

A variable in Python can have a short name (like x and y) or a more descriptive name (age, myName, total\_area).

Rules for Python variables:

- A variable name must start with a letter or the underscore character
- A variable name cannot start with a number.
- A variable name can only contain alpha-numeric characters and underscores (A-z, 0-9, and \_).
- Variable names are case-sensitive (age, Age and AGE are three different variables).
- We cannot use a reserved keyword as variable name.

### **Python Keywords:**

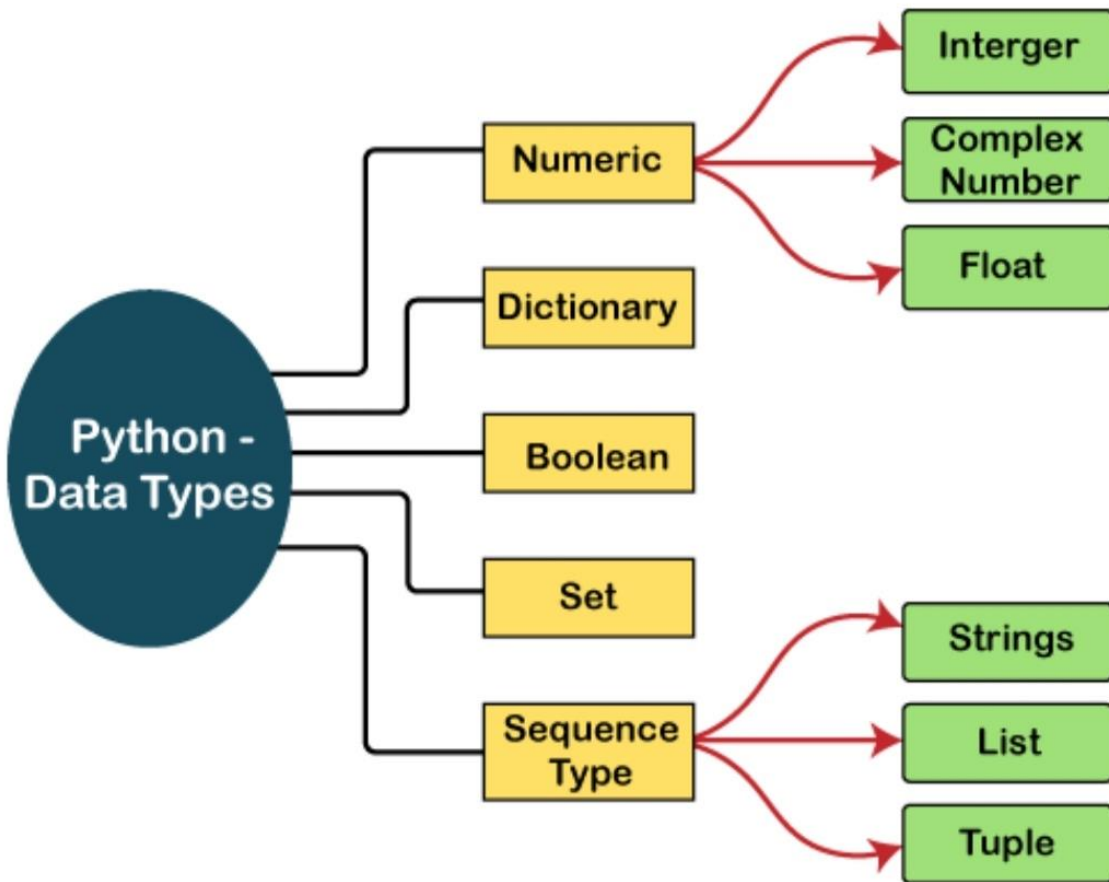
- Keywords are the reserved words in Python.
- We cannot use a keyword as a variable name, function name or any other Identifier. They are used to define the syntax and structure of the Python language.
- In Python, keywords are case sensitive.

There are 33 keywords in Python 3.7.x, the numbers can vary over time.

### Data types in Python:

The Basic Datatypes in Python are:

- Numeric
- Dictionary
- Boolean
- Set
- Sequence Type



### Numeric Data type:

The Different Numeric Data types in Python are :

- **Integer**  
*eg – 1,28,886, etc.*

- **Float**

*eg – 2.3, 3.7, 0.0000001, etc.*

- **Complex**

*eg - 6 + 8j, 5 + 0.3j, etc.*

### **Dictionary Data type:**

def: It's an unordered collection of data values. It's basically stores data in the form of key: value` pair.

- Each key-value pair in a Dictionary is separated by a colon :, whereas each key is separated by a 'comma'.
- We can initialize a set object using `dict()`.

### **Boolean Datatype:**

Boolean Variable are of Two Types (Values) :

- True
- False
- Any numerical value apart from 0 is regarded as `True`, or else `False`.
- We can initialize a set object using `bool()`.

### **Set Datatype:**

def: Set is an unordered collection of data type that is iterable, mutable and has no duplicate elements.

- We can initialize a set object using `set()`.

### **Sequence Type Datatype:**

It is known as Sequence type Datatypes as the storage locations are allocated in a continuous manner in the memory and can be accessed easily even if we know the address of a single entity.

There types are:

- Strings

*eg - "Swadesh", "Little", etc.*

- Lists

*eg - [], [1,2,3], [1, "Santa"], etc.*

- Tuples

*eg - (), (1,2,3), (1, "abcdefgh"), etc.*

## **DATA STRUCTURE IN PYTHON**

### **1. LISTS:**

- Ordered collection of data.
- Supports similar slicing and indexing functionalities as in the case of Strings.
- They are mutable.
- The list is changeable, meaning that we can change, add, and remove items in a list after it has been created.

- They can be of any data type. It can also hold elements of different datatypes at the same time.
- List items are indexed, the first item has index [0], the second item has index [1] etc.

Example-

- `my_list = ['one', 'two', 'three', 4, 5]`
- `len(my_list)` would output 5
- We can delete elements from a list by using `Del list_name[index_val]`

## **2. DICTIONARIES:**

- Dictionaries are used to store data values in key:value pairs.
- Dictionaries are changeable, meaning that we can change, add or remove items after the dictionary has been created.
- Dictionaries cannot have two items with the same key. The values will be overwritten.
- We can access the items of a dictionary by referring to its key name, inside square brackets. The syntax is-
- `thisdict= {"brand": "Ford", "model": "Mustang", "year": 1964}`  
`x = thisdict["model"]` will give "Mustang" as output.
- The `keys()` method will return a list of all the keys in the dictionary.  
`x = thisdict.keys()`

## **3. TUPLES:**

- Tuples are used to store multiple items in a single variable.
- A tuple is a collection which is ordered and **unchangeable** and can store duplicate values.

Ex: `thistuple= ("apple", "banana", "cherry")`  
`print(thistuple)`

- Tuples are unchangeable, meaning that we cannot change, add or remove items after the tuple has been created.
- Indexing and slicing, everything's same like that in strings and lists.

## **4. SETS:**

- Sets are used to store multiple items in a single variable.
- A set is a collection which is both *unordered* and *unindexed*.

- Sets are written with curly brackets.
- Set items are unordered, unchangeable, and do not allow duplicate values.
- Sets are unchangeable, meaning that we cannot change the items after the set has been created, but we can add new items.
- Basic Syntax-

```
x=set()
x.add(1)
x = { 1 }
x.add(1) - This would make no change in x now
```

## **WEEK 2-**

<b>DAY-6(15.02.2021)</b>	<ul style="list-style-type: none"> <li>• Difference between == and isoperator</li> <li>• What does the if name == "main":do?</li> <li>• What is Python MainFunction?</li> <li>• Functions</li> </ul>
<b>DAY-7(16.02.2021)</b>	<ul style="list-style-type: none"> <li>• PythonArguments</li> <li>• Python ArbitraryArguments</li> <li>• Recursion</li> <li>• What are Python Built-InFunctions?</li> <li>• Python ZipFunction</li> <li>• PythonIterators</li> <li>• Python LambdaFunctions</li> </ul>
<b>DAY-8(17.02.2021)</b>	<ul style="list-style-type: none"> <li>• Python MapFunction</li> <li>• Python FilterFunction</li> </ul>
<b>DAY-9(18.02.2021)</b>	<ul style="list-style-type: none"> <li>• Returning Multiple Values from PythonFunction.</li> <li>• Importing Modules in Python.</li> <li>• Error Handling inPython.</li> </ul>

### **Python Zip Function:**

The zip() function returns a zip object, which is an iterator of tuples where the first item in each passed iterator is paired together, and then the second item in each passed iterator are paired together etc.

```
Syntax-zip(iterator1, iterator2, iterator3 ...)
Ex : a = ("John", "Charles", "Swadesh")
b = ("Jenny", "Hayley", "Ankita", "Vicky")
x = zip(a, b)
```

## **Python Iterators:**

- An iterator is an object that contains a countable number of values.
- An iterator is an object that can be iterated upon, meaning that we can traverse through
- all the values.
- In Python, an iterator is an object which implements the iterator protocol, which consist of the methods `siter()` and `next()`.

```
• Ex: mytuple = ("apple", "banana", "cherry")
myit = iter(mytuple)
print(next(myit))
print(next(myit))
print(next(myit))
```

## **Python Lambda Functions:**

A lambda function is a small anonymous function.

A lambda function can take any number of arguments, but can only have one expression.

Lambda functions can take any number of arguments

### **Syntax-**

```
lambda arguments : expression
x = lambda a : a + 10
print (x (5))
```

## **Python Map Function:**

The `map()` function executes a specified function for each item in an iterable. The item is sent to the function as a parameter.

### **Syntax - map(function, iterables)**

```
Ex: def myfunc(a, b):
    return a + b
```

```
x = map(myfunc, ('apple', 'banana', 'cherry'), ('orange', 'lemon', 'pineapple'))
print(x)
```

## **Python Filter Function:**

The `filter()` function returns an iterator where the items are filtered through a function to test if the item is accepted or not.

### **Syntax - filter(function, iterable)**

```
Ex - ages = [5, 12, 17, 18, 24, 32]
```

```
def myFunc(x):
    if x < 18:
        return False
```

```
    else:  
        return True  
adults = filter(myFunc, ages) for x in adults:  
    print(x)
```

### **Importing Modules in Python :**

Import in python is similar to #include header file in C/C++. Python modules can get access to code from another module by importing the file/function using import. The import statement is the most common way of invoking the import machinery.

### **Error Handling in Python:**

Error in Python can be of two types i.e. Syntax errors and Exceptions. Errors are the problems in a program due to which the program will stop the execution. On the other hand, exceptions are raised when some internal events occur which changes the normal flow of the program.

#### **Syntax Error:**

As the name suggest this error is caused by wrong syntax in the code. It leads to the termination of the program

#### **Exceptions:**

Exceptions are raised when the program is syntactically correct but the code resulted in an error. This error does not stop the execution of the program, however, it changes the normal flow of the program. For example the ZeroDivisionError is an Exception. To prevent this, you can handle a possible exception by enclosing your statements inside a try-except statement.

### **WEEK 3-**



<b>DAY-10(22.02.2021)</b>	<ul style="list-style-type: none"> <li>• File Handling in Python.</li> <li>• Introduction to OOPs using Python</li> <li>• Introduction to OOPs o Declaring Class</li> </ul>
<b>DAY-11(23.02.2021)</b>	<ul style="list-style-type: none"> <li>• Introduction to OOPs using Python <ul style="list-style-type: none"> <li>o Intro to OOPs</li> <li>o Declaring Class</li> <li>o Class Methods</li> <li>o Static Methods</li> <li>o Decorators</li> </ul> </li> </ul>
<b>DAY-12(25.02.2021)</b>	<ul style="list-style-type: none"> <li>• OOPs using Python <ul style="list-style-type: none"> <li>o Magic or DUNDER Methods</li> <li>o self vs class</li> <li>o Inheritance</li> <li>o Polymorphism</li> </ul> </li> </ul>
<b>DAY-13(26.02.2021)</b>	<ul style="list-style-type: none"> <li>• OOPs using Python <ul style="list-style-type: none"> <li>o Object Overriding (Re)</li> <li>o Abstraction (Implementation Hiding)</li> <li>o Encapsulation (Information Hiding)</li> </ul> </li> </ul>

## **Introduction to OOPs**

Object-Oriented Programming (OOP) is the term used to describe a programming approach based on objects and classes. The object-oriented paradigm allows us to organize software as a collection of objects that consist of both data and behavior. This is in contrast to conventional functional programming practice that only loosely connects data and behavior.

## **Declaring Class**

Class creates a user-defined data structure, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A class is like a blueprint for an object. Some points on Python class:

- Classes are created by keyword class.
- Attributes are the variables that belong to a class.
- Attributes are always public and can be accessed using the dot (.) operator.

## **Instance Methods**

Instance methods are functions that are defined inside a class and can only be called from an instance of that class. Just like `__init__()`, an instance method's first parameter is always self.

## **Class Methods**

- A class method is a method which is bound to the class and not the object of the class.
- They have the access to the state of the class as it takes a class parameter that points to the class and

not the object instance.

- It can modify a class state that would apply across all the instances of the class.

For example it can modify a class variable that will be applicable to all the instances.

Syntax : class C(object):

@classmethod

def fun(cls, arg1, arg2, ...): ....

fun: function that needs to be converted into a class method

returns: a class method for function.

## **Static Methods**

- A static method is also a method which is bound to the class and not the object of the class.
- A static method can't access or modify class state.
- Static methods do not know about class state. These methods are used to do some utility tasks by taking some parameters.

- @staticmethod decorator is used here.

- Syntax : class my\_class:

@staticmethod

11 def function\_name(arguments):

#Function Body

return value

## **Decorators**

Python has an interesting feature called Decorators to add functionality to an existing code. This is also called metaprogramming because a part of the program tries to modify another part of the program at compile time.

## **WEEK 4-**

<b>DAY-13(01.03.2021)</b>	<ul style="list-style-type: none"> <li>•Revision: Assignment Topics</li> <li>•Self Keyword</li> <li>• Magic Methods</li> <li>•Polymorphism</li> <li>•Method Overloading</li> <li>•Method Overriding</li> <li>•Abstraction</li> <li>•Encapsulation</li> <li>•Inheritance</li> </ul>
<b>DAY-13(02.03.2021)</b>	<ul style="list-style-type: none"> <li>•Assigning of 1st Project</li> </ul>

### **Self Keyword**

Self represents the instance of the class. By using the “self” keyword we can access the attributes and methods of the class in python. It binds the attributes with the given arguments.

The reason we need to use self is because Python does not use the @ syntax to refer to instance attributes. Python decided to do methods in a way that makes the instance to which the method belongs be passed automatically, but not received automatically: the first parameter of methods is the instance the method is called on.

### **Difference between self and cls keyword:**

- The difference between the keywords self and cls reside only in the method type.
- If the created method is an instance method then the reserved word self has to be used, but if the method is a class method then the keyword cls must be used.
- Finally, if the method is a static method then none of those words will be used because as said before, static methods are self-contained and do not have access to the instance or class variables nor to the instance or class methods

### **Dunder or Magic Methods**

Dunder or magic methods in python are the methods having two prefix and suffix underscores in the method name. Dunder here means “Double Under (Underscores)”. These are commonly used for operator overloading. Few examples for magic methods are: \_\_init\_\_, \_\_add\_\_, \_\_len\_\_, \_\_repr\_\_ etc.

### **INHERITANCE**

- Inheritance allows us to define a class that inherits all the methods and properties from another class.
- The class which inherits all the properties is called a child or derived class. And the class from which the properties are being derived is called parent or base class.
- Inheriting all the properties refers to the ability of a child class to access all variables and methods

defined inside a parent class.

- It provides reusability of a code.
- Different forms of inheritance include : single inheritance, multiple inheritance, multi-level inheritance, Hierarchical inheritance and hybrid inheritance.

## **POLYMORPHISM**

- The word polymorphism means having many forms.
- In programming, polymorphism means same function name (but different signatures) being used for different types.
- It is also useful in creating different classes which will have class methods with same name. That helps in re using a lot of code and decreases code complexity.

## **Method Overloading**

Method overloading is the class having methods that are of the same name with different arguments. Arguments different will be based on a number of arguments and types of arguments. It is used in a single class. It is also used to write the code clarity as well as reduce complexity.

## **Method Overriding**

Method Overriding is the method having the same name with the same arguments. It is implemented with inheritance also. It is mostly used for memory reducing processes.

## **ABSTRACTION (Implementation Hiding)**

- Abstraction in Python is the process of hiding the real implementation of an application from the user and emphasizing only on usage of it.
- For example, consider we bought a new electronic gadget. Along with the gadget, you get a user guide, instructing how to use the application, but this user guide has no info regarding the internal working of the gadget.

## **ENCAPSULATION (Information Hiding)**

- Encapsulation is one of the fundamental concepts in object-oriented programming (OOP).
- It describes the idea of wrapping data and the methods that work on data within one unit.
- This puts restrictions on accessing variables and methods directly and can prevent the accidental modification of data.
- To prevent accidental change, an object's variable can only be changed by an object's method. Those types of variables are known as private variable. A class is an example of encapsulation as it encapsulates all the data that is member functions, variables, etc.

## **PROJECT 1: SNAKES & LADDERS**

Here we had to design a snakes and ladders game in python which will give the output in the form of text and with the help of print statement will show the positions of a particular player and the results.

### **STRUCTURE OF THE GAME:**

- The game will only be text output based, i.e. no GUI will be shown to the user. Instead, it will use print statements to display the position of a particular player, end results, etc.
- There are only 2 players in the game and no spectators are allowed.
- The first thing that the game will ask the input will be the names of the players.
- The input should first ask the name of Player 1 and then after giving input, it should ask the name of Player 2.
- The game will alternate between these 2 players.
- For a particular player's chance, it will ask to input his/her chance. If Player 1 enters " roll " (excluding quotes), then a random number will be generated between 1 and 6 (both inclusive).
- The output will be shown and then the final position will be displayed each time a player plays his/her chance.
- The player starts at 0 and has to reach the final position (in our case, it's 100).
- The position of snakes and ladders are stored in start: end format where start denotes the position of ladder/snake and end denotes the value that it reaches.

Position of snakes: { 17 : 7, 54 : 34, 62 : 19, 98 : 79 }

Position of ladders: { 3 : 38, 24 : 33, 42 : 93, 72 : 84 }

### **CODE :**

```
import random
import time
print("###Welcome to Snakes and Ladders Game###\n")
plr1=input("Enter the name of player 1 : ")
plr2=input("Enter the name of player 2 : ")
print("\nThere are two modes here. For Manual mode you can enter the
value between 1-19 but in auto mode you need to type 'roll'.\n")
mode=input("Which mode you want to play, 'manual' or 'auto' mode : ")
print("\n###Let's start the Game###")
snake={17 : 7, 54 : 34, 62 : 19, 98 : 79}
ladder={3 : 38, 24 : 33, 42 : 93, 72 : 84}

def snake_check(s1,s2):
    wrng1,wrng2="  ", "  "
    try:
        s1=snake[s1]
        wrng1="player 1 got snakebite"
    except:
        pass
    try:
        s2=snake[s2]
        wrng2="player 2 got snakebite"
```

```

except:
    pass
return s1,s2,wrng1,wrng2

def ladder_check(s1,s2):
    wrng1,wrng2=" _____"," _____"
    try:
        s1=ladder[s1]
        wrng1="player 1 got the ladder"
    except:
        pass
    try:
        s2=ladder[s2]
        wrng2="player 2 got the ladder"
    except:
        pass
    return s1,s2,wrng1,wrng2

def roll1(s2):
    import random
    u=input("Player 1 : ")
    #u="roll"
    d=random.randint(1,6)
    if u=="quit":
        s2=100
    elif u=="roll":
        print(f"You got a {d} ")
    return d,s2

def roll2(s1):
    import random
    u=input("Player 2 : ")
    #u="roll"
    e=random.randint(1,6)
    if u=="quit":
        s1=100
    elif u=="roll":
        print(f"You got a {e} ")
    return e,s1

def roll1_man(s2):
    u=input("Player 1 : ")
    try:
        d=int(u)
        if d<20 and d>0:

```

```

        pass
    else:
        print("##invalid input...##")
        d=0
except:
    pass
if u=="quit":
    d=0
    s2=100
else:
    print(f"You got a {d} ")
return d,s2

def roll2_man(s1):
    u=input("Player 2 : ")
    try:
        e=int(u)
        if e<20 and e>0:
            pass
        else:
            print("##invalid input...##")
            e=0
    except:
        pass
    if u=="quit":
        e=0
        s1=100
    else:
        print(f"You got a {e} ")
    return e,s1

def win_check():
    if s1==100 or s2==100:
        if s1==100:
            winner=plr1
        elif s2==100:
            winner=plr2
        print(f"\n{winner} won the game")
        print("\n###_Game Successfully completed_###")
        quit()
    else:
        winner=None
    return winner

if __name__=="__main__":
    winner=None
    s1,s2=0,0

```

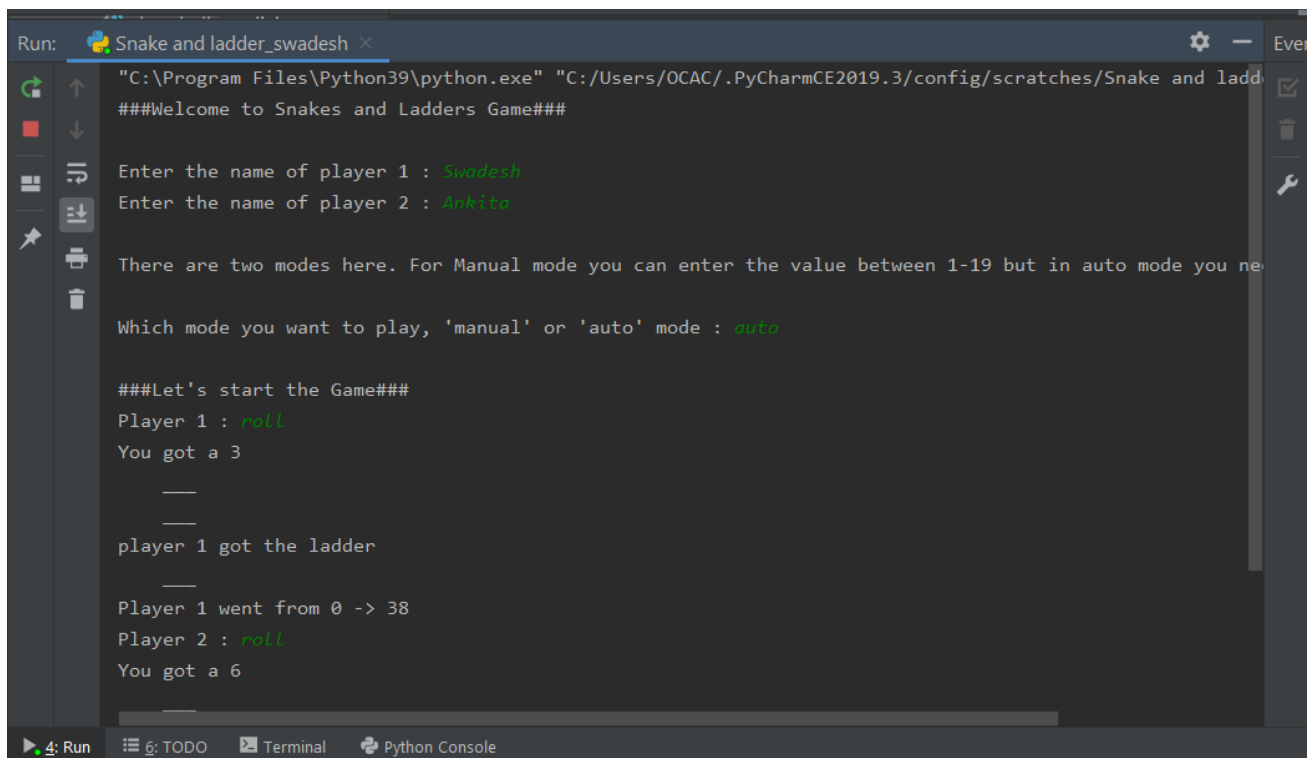
```

while True:
    if winner!=plr1 or winner!=plr2:
        s1past,s2past=s1,s2
        if mode=="auto":
            y,s2=roll1(s2)
        elif mode=="manual":
            y,s2=roll1_man(s2)
        s1+=y
        if s1>100:
            s1-=y
        s1,s2,wrng1,wrng2=snake_check(s1,s2)
        print(f"{wrng1}\n{wrng2}")
        s1,s2,wrng1,wrng2=ladder_check(s1,s2)
        print(f"{wrng1}\n{wrng2}")
        print(f"Player 1 went from {s1past} -> {s1}")
        winner=win_check()
        if mode=="auto":
            z,s1=roll2(s1)
        elif mode=="manual":
            z,s1=roll2_man(s1)
        s2+=z
        if s2>100:
            s2-=z
        s1,s2,wrng1,wrng2=snake_check(s1,s2)
        print(f"{wrng1}\n{wrng2}")
        s1,s2,wrng1,wrng2=ladder_check(s1,s2)
        print(f"{wrng1}\n{wrng2}")
        print(f"Player 2 went from {s2past} -> {s2}\n- - - - -")
        winner=win_check()
    else:
        pass

```



## OUTPUT



```

Run: Snake and ladder_swadesh x
"C:\Program Files\Python39\python.exe" "C:/Users/OCAC/.PyCharmCE2019.3/config/scratches/Snake and ladder_swadesh.py"
###Welcome to Snakes and Ladders Game###

Enter the name of player 1 : Swadesh
Enter the name of player 2 : Ankita

There are two modes here. For Manual mode you can enter the value between 1-19 but in auto mode you need not.

Which mode you want to play, 'manual' or 'auto' mode : auto

###Let's start the Game###
Player 1 : roll
You got a 3

---

player 1 got the ladder

---

Player 1 went from 0 -> 38
Player 2 : roll
You got a 6

```

## PROJECT 2: TIC TAC TOE

Here we have designed a two players tic-tac-toe game, which we will be able to play against the computer.

Initially, we'll make an empty game board and then we'll take inputs from the players and we'll check for the winning condition and if the whole board gets filled, the results will be declared as "You won" or "You lost".

### STEPS INVOLVED IN DESIGNING GAME:

- Designed the board.( The Tic Tac Toe game board.)
- Declared the possible moves.
- Program the winning combinations.
- Wrote a function to display the board.
- Used the random library, to program the Computer's moves.
- Wrote a function to check whether the move is valid.
  
- Wrote a function to check whether we can win the game.
- Wrote a function to make the move on the board.
- Wrote a function for the Computer and program its future moves.
- Wrote a function to check for overflow.
- Wrote the main method.
  - i. Ask the user to enter their moves.
  - ii. Call the required functions to run the game.
  - iii. Display the results.

**CODE :**

```

import random
import time

#1-designing the board
tb={1:' ',2:' ',3:' ',4:' ',5:' ',6:' ',7:' ',8:' ',9:' '}
bd_keys=[]
print("This is the Board:\n 1|2|3 \n---+---\n 4|5|6 \n---+---\n 7|8|9\n")

#4-display the board
def view(bd):
    print(" "+bd[1]+"|" +bd[2]+"|" +bd[3])
    print("---+---")
    print(" " + bd[4] + "|" + bd[5] + "|" + bd[6])
    print("---+---")
    print(" " + bd[7] + "|" + bd[8] + "|" + bd[9]+"\n")

#checking the validity of the move
def check_valid():
    if tb[move]==' ':
        valid=True
    else:
        valid=False
        print("Invalid Move...\nThis place is already filled.\n")
    return valid

#win checking
def win_check(bd):
    if bd[1]==bd[2]==bd[3]!=' ' or bd[4]==bd[5]==bd[6]!=' ' or
bd[7]==bd[8]==bd[9]!=' ' or bd[1]==bd[4]==bd[7]!=' ' or
bd[2]==bd[5]==bd[8]!=' ' or bd[3]==bd[6]==bd[9]!=' ' or
bd[1]==bd[5]==bd[9]!=' ' or bd[3]==bd[5]==bd[7]!=' ' :
        return True
    else:
        return False

#altering the 'turn' into x and o
def alter(a):
    if a=="x":
        return "o"
    elif a=="o":
        return "x"

if __name__=="__main__":
    choice=input("Pick 'x' or 'o': ")
    if choice=="x":
        turn="x"

```

```

else:
    turn="o"
count=0
view(tb)
for i in range(5):

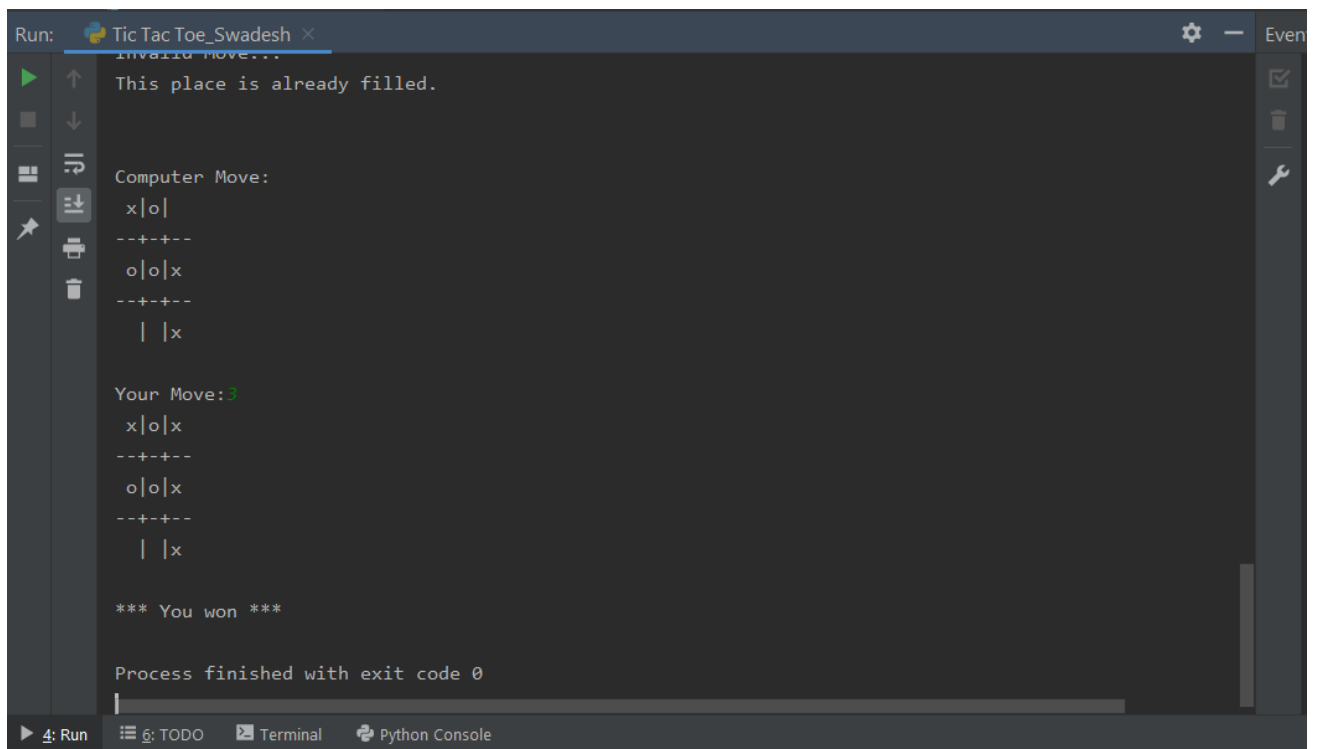
    while True:
        move=int(input("Your Move:"))
        vld=check_valid()
        if vld==True:
            tb[move] = turn
            count += 1
            break
        else:
            continue
    view(tb)
    if count>=9:
        print('\n*** Tie ***\n__Game Over__')
        break
    turn=alter(turn)

    if win_check(tb)==True:
        print('*** You won ***')
        break
    time.sleep(1)

    while True:
        print('\nComputer Move:')
        move=random.randint(1,9)
        vld=check_valid()
        if vld==True:
            tb[move] = turn
            count += 1
            break
        else:
            continue
    view(tb)
    turn=alter(turn)
    if win_check(tb) == True:
        print('*** You Lose ***')
        break

```

## OUTPUT



```
Run: Tic Tac Toe_Swadesh X
invalid move...
This place is already filled.

Computer Move:
x|o|
---+---
o|o|x
---+---
| |x

Your Move:
x|o|x
---+---
o|o|x
---+---
| |x

*** You won ***

Process finished with exit code 0
```

The screenshot shows a code editor window with a tab titled 'Tic Tac Toe\_Swadesh'. The 'Run' console is open, displaying the output of a Python program. The output shows a Tic Tac Toe game where the computer moves first, followed by the user. The user wins the game. The console also shows an 'invalid move...' message and 'Process finished with exit code 0'. The editor's interface includes a sidebar with icons for Run, Up, Down, Run and Debug, Print, and Delete, and a bottom status bar with '4: Run', '6: TODO', 'Terminal', and 'Python Console'.

## WEEK 5 –

DAY AND DATE	TOPICS COVERED
15 <sup>TH</sup> MARCH	<ul style="list-style-type: none"> <li>• Anaconda installation</li> <li>• Introduction</li> </ul>
16 <sup>TH</sup> MARCH	<ul style="list-style-type: none"> <li>• Data analysis</li> <li>• Numpy</li> <li>• Steps to create Numpy array</li> <li>• Mathematical function function on Numpy array</li> <li>• Itering over Rows and columns</li> <li>• String operation</li> </ul>
17 <sup>TH</sup> MARCH	<ul style="list-style-type: none"> <li>• Sorting using Numpy</li> <li>• Pandas dataframe and Series</li> </ul>
18 <sup>TH</sup> MARCH	<ul style="list-style-type: none"> <li>• Matplotlib</li> <li>• Seaborn library</li> </ul>
19 <sup>TH</sup> MARCH	<ul style="list-style-type: none"> <li>• OpenCV</li> </ul>

## NUMPY LIBRARY-

NumPy is a general-purpose array-processing package. It provides a high-performance multi-dimensional array object, and tools for working with these arrays. It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows NumPy to seamlessly and speedily integrate with a wide variety of databases.

### Arrays in NumPy:

NumPy's main object is the homogeneous multidimensional array.

- It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers.
- In NumPy dimensions are called *axes*. The number of axes is *rank*.
- NumPy's array class is called **ndarray**. It is also known by the alias **array**.

### Array creation:

There are various ways to create arrays in NumPy.

- For example, you can create an array from a regular Python list or **tuple** using the **array** function. The type of the resulting array is deduced from the type of the elements in the sequences.
- Often, the elements of an array are originally unknown, but its size is known. Hence, NumPy offers several functions to create arrays with **initial placeholder content**. These minimize the necessity of growing arrays, an expensive operation.

**For example:** np.zeros, np.ones, np.full, np.empty, etc.

- To create sequences of numbers, NumPy provides a function analogous to range that returns arrays instead of lists.
- **arange:** returns evenly spaced values within a given interval. **step** size is specified.
- **linspace:** returns evenly spaced values within a given interval. **num** no. of elements are returned.
- **Reshaping array:** We can use **reshape** method to reshape an array. Consider an array with shape (a1, a2, a3, ..., aN). We can reshape and convert it into another array with shape (b1, b2, b3, ..., bM). The only required condition is:  $a1 \times a2 \times a3 \dots \times aN = b1 \times b2 \times b3 \dots \times bM$ . (i.e original size of array remains unchanged.)
- **Flatten array:** We can use **flatten** method to get a copy of array collapsed into **one dimension**. It accepts *order* argument. Default value is 'C' (for row-major order). Use 'F' for column major order.

### Array Indexing:

Knowing the basics of array indexing is important for analysing and manipulating the array object. NumPy offers many ways to do array indexing.

- **Slicing:** Just like lists in python, NumPy arrays can be sliced. As arrays can be multidimensional, you need to specify a slice for each dimension of the array.
- **Integer array indexing:** In this method, lists are passed for indexing for each dimension. One to one mapping of corresponding elements is done to construct a new arbitrary array.
- **Boolean array indexing:** This method is used when we want to pick elements from array which satisfy some condition.

### Basic operations:

Plethora of built-in arithmetic functions are provided in NumPy.

- **Operations on single array:** We can use overloaded arithmetic operators to do element-wise operation on array to create a new array. In case of +=, -=, \*= operators, the existing array is modified.
  1. **Unary operators:** Many unary operations are provided as a method of **ndarray** class. This includes sum, min, max, etc. These functions can also be applied row-wise or column-wise by setting an axis parameter.
  2. **Binary operators:** These operations apply on array elementwise and a new array is created. You can use all basic arithmetic operators like +, -, /, , etc. *In case of +=, -=, = operators, the existing array is modified.*

### Sorting array:

There is a simple **np.sort** method for sorting NumPy arrays.

## PANDAS LIBRARY-

**Pandas DataFrame:** is two-dimensional size-mutable, potentially heterogeneous tabular data structure with labeled axes (rows and columns). A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. Pandas DataFrame consists of three principal components, the **data**, **rows**, and **columns**.

The diagram illustrates a Pandas DataFrame as a table with 6 rows and 5 columns. The columns are labeled 'Name', 'Team', 'Number', 'Position', and 'Age'. The rows are indexed from 0 to 6. Arrows point from the labels 'Columns' and 'Rows' to their respective axes. A pink box highlights a subset of the data, specifically the 'Number' and 'Position' columns for rows 2 through 6, labeled 'Data'.

	Name	Team	Number	Position	Age
0	Avery Bradley	Boston Celtics	0.0	PG	25.0
1	John Holland	Boston Celtics	30.0	SG	27.0
2	Jonas Jerebko	Boston Celtics	8.0	PF	29.0
3	Jordan Mickey	Boston Celtics	NaN	PF	21.0
4	Terry Rozier	Boston Celtics	12.0	PG	22.0
5	Jared Sullinger	Boston Celtics	7.0	C	NaN
6	Evan Turner	Boston Celtics	11.0	SG	27.0

We will get a brief insight on all these basic operation which can be performed on Pandas DataFrame :

- Creating a DataFrame
- Dealing with Rows and Columns
- Indexing and Selecting Data
- Working with Missing Data
- Iterating over rows and columns

### *Creating a Pandas DataFrame*

In the real world, a Pandas DataFrame will be created by loading the datasets from existing storage, storage can be SQL Database, CSV file, and Excel file. Pandas. DataFrame can be created from the lists, dictionary, and from a list of dictionary.

etc. Dataframe can be created in different ways here are some ways by which we create a dataframe:

**Creating a dataframe using List:** DataFrame can be created using a single list or a list of lists.

**Creating a dataframe from dict of ndarrays / lists:** To create DataFrame from dict of ndarray/list, all the ndarray must be of same length. If index is passed then the length index should be equal to the length of arrays. If no index is passed, then by default, index will be range(n) where n is the array length.

# Python code demonstrate creating dataframe from dict narray / lists  
 # By default addresses.

```
import pandas as pd
```

# initialise data of lists.

```
data = {'Name':['Tom', 'nick', 'krish', 'jack'],  
        'Age':[20, 21, 19, 18]}
```

# Create DataFrame

```
df = pd.DataFrame(data)
```

# Print the output.

```
print(df)
```

Run on IDE

**Output:**

	Name	Age
0	Tom	20
1	nick	21
2	krish	19
3	jack	18

Dealing with rows and columns:

A Data frame is a two-dimensional data structure, i.e., data is aligned in a tabular fashion in rows and columns. We can perform basic operations on rows/columns like selecting, deleting, adding, and renaming.

Column Selection:

In Order to select a column in Pandas DataFrame, we can either access the columns by calling them by their columns name.

Row Selection:

Pandas provide a unique method to retrieve rows from a Data frame. DataFrame.loc[] method is used to retrieve rows from Pandas DataFrame. Rows can also be selected by passing integer location to an iloc[] function.

Indexing and Selecting Data:

Indexing in pandas means simply selecting particular rows and columns of data from a DataFrame. Indexing could mean selecting all the rows and some of the columns, some of the rows and all of the columns, or some of each of the rows and columns. Indexing can also be known as Subset Selection.

Indexing a DataFrame using indexing operator []:



Indexing operator is used to refer to the square brackets following an object. The `.loc` and `.iloc` indexers also use the indexing operator to make selections. In this indexing operator to refer to `df[]`.

Selecting a single columns: In order to select a single column, we simply put the name of the column in-between the brackets.

Indexing a DataFrame using `.loc[...]`:

This function selects data by the label of the rows and columns. The `df.loc` indexer selects data in a different way than just the indexing operator. It can select subsets of rows or columns. It can also simultaneously select subsets of rows and columns.

Selecting a single row: In order to select a single row using `.loc[]`, we put a single row label in a `.loc` function.

Iterating over rows and columns:

Iteration is a general term for taking each item of something, one after another. Pandas DataFrame consists of rows and columns so, in order to iterate over dataframe, we have to iterate a dataframe like a dictionary.

Iterating over rows :

In order to iterate over rows, we can use three function `iteritems()`, `iterrows()`, `itertuples()` . These three function will help in iteration over rows.

Iterating over Columns :

In order to iterate over columns, we need to create a list of dataframe columns and then iterating through that list to pull out the dataframe columns.

Pandas Series:

Pandas Series is a one-dimensional labeled array capable of holding data of any type (integer, string, float, python objects, etc.). The axis labels are collectively called index. Pandas Series is nothing but a column in an excel sheet.

Basic operations which can be performed on Pandas Series :

- Creating a Series
- Accessing element of Series
- Indexing and Selecting Data in Series
- Binary operation on Series
- Conversion Operation on Series

## MATPLOTLIB-

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. It was introduced by John Hunter in the year 2002.

One of the greatest benefits of visualization is that it allows us visual access to huge amounts of data in easily digestible visuals. Matplotlib consists of several plots like line, bar, scatter, histogram etc.

Basic plots in Matplotlib :

Matplotlib comes with a wide variety of plots. Plots helps to understand trends, patterns, and to make correlations. They're typically instruments for reasoning about quantitative information. Some of the sample plots are covered here.

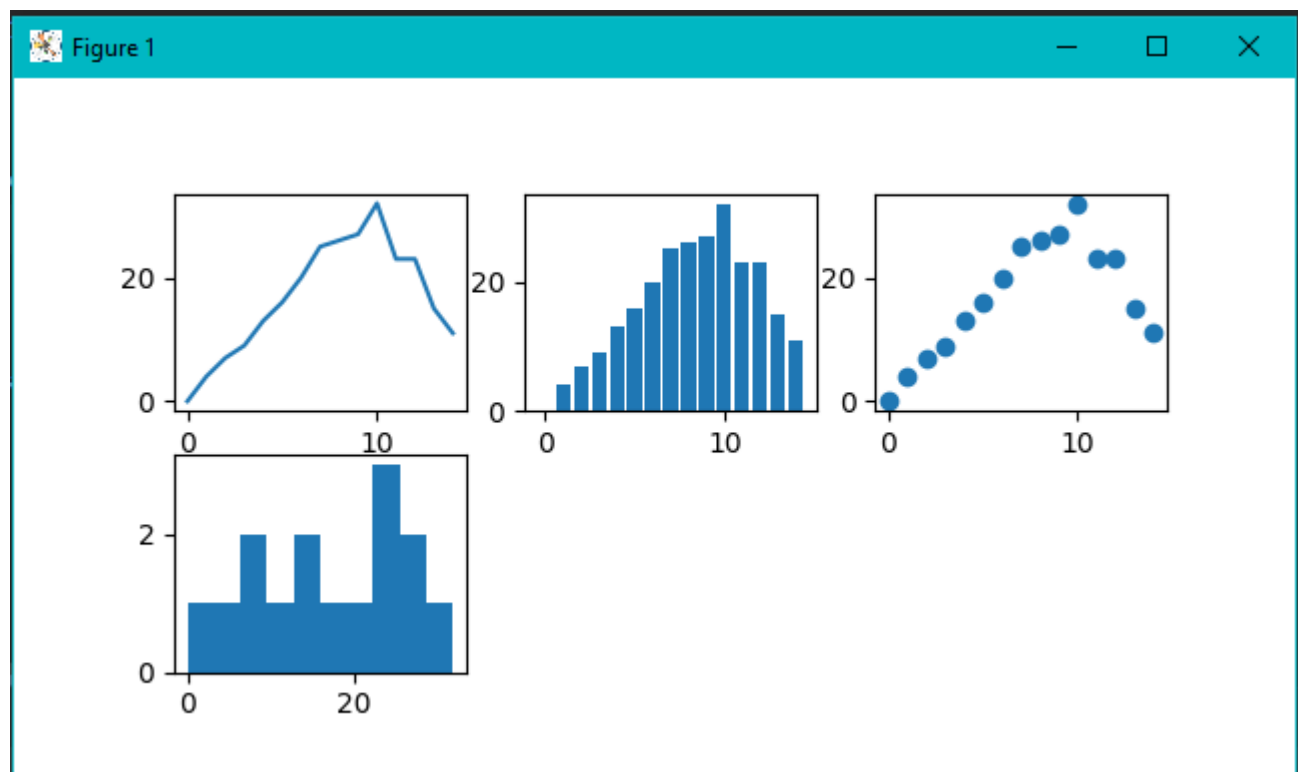
**Example:**

```
from matplotlib import pyplot as plt

time=[0,1,2,3,4,5,6,7,8,9,10,11,12,13,14]
distance=[0,4,7,9,13,16,20,25,26,27,32,23,23,15,11]

plt.subplot(3,3,1)
plt.plot(time,distance)
plt.subplot(3,3,2)
plt.bar(time,distance)
plt.subplot(3,3,3)
plt.scatter(time,distance)
plt.subplot(3,3,4)
plt.hist(distance)
plt.show()
```

**Output:**



## ASSIGNMENT-

The animated simulation of a projectile motion of a ball.

Code:

```
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.animation import FuncAnimation

"""
45' inclination makes both horizontal and longitudinal speed
 $\cos 45^\circ u = \sin 45^\circ u = 0.707u$ ... Here  $u=100$  is given so  $100 \cdot \cos 45^\circ = 70.7$ 
for inclination other than  $45^\circ$  speed will be different for horizontal and
longitudinal direction.
here i assumed the ball losses its speed with each collision with ground
by a value "elst=0.8".
"""

u=100
angle=60
t1=0
s=0
elst=0.7
ux=u*np.cos(np.deg2rad(angle))
uy=u*np.sin(np.deg2rad(angle))
def traj(t):
    global ux,uy,t1,s
    y=uy*(t-t1)-.5*9.8*((t-t1)**2)
    if y>0:
        pass
    elif y<0:
        y=0
        s += ux * (t - t1)
        t1=t
        uy=uy*elst
        ux=ux*elst
    x=s+ux*(t-t1)
    #print(t, ' //// ', t1, ' //// ', uy, ' //// ', x, ' //// ', y)
    return x,y

def init():
    ax.set_xlim(-200,2200)
    ax.set_ylim(-200,400)
    return ln,

fig, ax = plt.subplots()
xdata, ydata = [], []
```

```

ln, = plt.plot([], [], "ro")

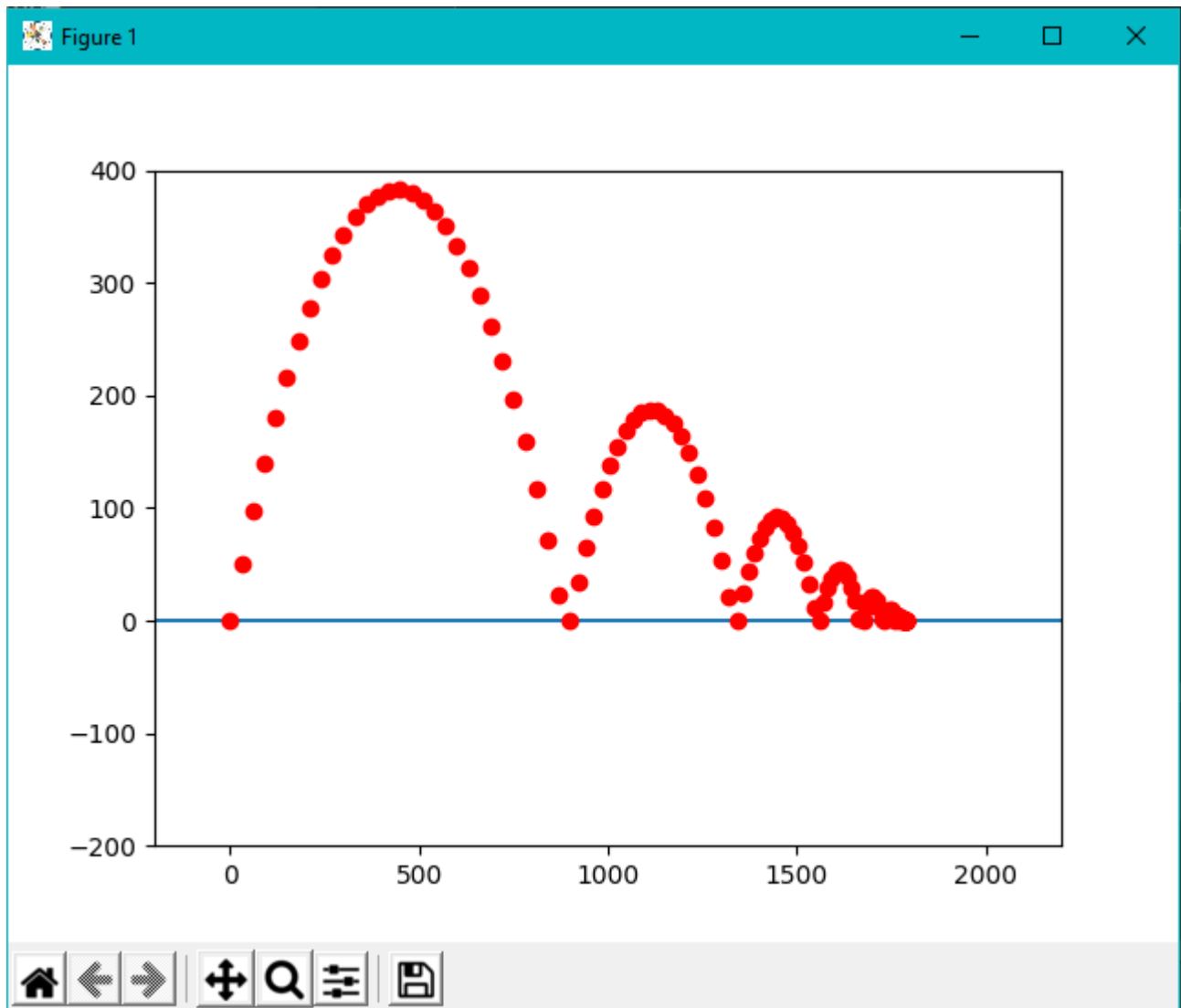
# # for ground-line...
g = np.arange(-250, 2200, 1)
w=0*g
l = plt.plot(g,w)

def update(frame):
    x,y=traj(frame)
    print(frame, ' //// ',t1, ' //// ',u, ' //// ',y)
    xdata.append(x)
    ydata.append(y)
    ln.set_data(xdata, ydata)
    return ln,

ani = FuncAnimation(fig, update, frames=np.linspace(0, 3000,
5000),init_func=init, interval=20, blit=True)
plt.show()

```

Output:



## Introduction to Seaborn

Seaborn is an amazing visualization library for statistical graphics plotting in Python. It provides beautiful default styles and color palettes to make statistical plots more attractive. It is built on the top of [matplotlib](#) library and also closely integrated to the data structures from [pandas](#). Seaborn aims to make visualization the central part of exploring and understanding data. It provides dataset-oriented APIs, so that we can switch between different visual representations for same variables for better understanding of dataset.

Different categories of plot in Seaborn

Plots are basically used for visualizing the relationship between variables. Those variables can be either be completely numerical or a category like a group, class or division. Seaborn divides plot into the below categories –

- **Relational plots:** This plot is used to understand the relation between two variables.
- **Categorical plots:** This plot deals with categorical variables and how they can be visualized.
- **Distribution plots:** This plot is used for examining univariate and bivariate distributions
- **Regression plots:** The regression plots in seaborn are primarily intended to add a visual guide that helps to emphasize patterns in a dataset during exploratory data analyses.

- **Matrix plots:** A matrix plot is an array of scatterplots.
- **Multi-plot grids:** It is an useful approach is to draw multiple instances of the same plot on different subsets of the dataset.

### **Some basic plots using seaborn**

**Dist plot :** Seaborn dist plot is used to plot a histogram, with some other variations like kdeplot and rugplot.

**Line plot :** The line plot is one of the most basic plot in seaborn library. This plot is mainly used to visualize the data in form of some time series, i.e. in continuous manner.

**Lmplot :** The Lmplot is another most basic plot. It shows a line representing a linear regression model along with data points on the 2D-space and x and y can be set as the horizontal and vertical labels respectively.

## **INTRODUCTION TO OPENCV-**

OpenCV is a huge open-source library for computer vision, machine learning, and image processing. OpenCV supports a wide variety of programming languages. It can process images and videos to identify objects, faces, or even the handwriting of a human. When it is integrated with various libraries, such as Numpy which is a highly optimized library for numerical operations, then the number of weapons increases in your Arsenal i.e whatever operations one can do in Numpy can be combined with OpenCV.

**Applications of OpenCV:** There are lots of applications which are solved using OpenCV, some of them are listed below

- face recognition
- Automated inspection and surveillance
- number of people – count (foot traffic in a mall, etc)
- Vehicle counting on highways along with their speeds
- Interactive art installations
- Anamoly (defect) detection in the manufacturing process (the odd defective products)
- Street view image stitching
- Video/image search and retrieval
- Robot and driver-less car navigation and control
- object recognition
- Medical image analysis
- Movies – 3D structure from motion
- TV Channels advertisement recognition

### **OpenCV Functionality:**

- Image/video I/O, processing, display (core, imgproc, highgui)
- Object/feature detection (objdetect, features2d, nonfree)
- Geometry-based monocular or stereo computer vision (calib3d, stitching, videostab)
- Computational photography (photo, video, superres)
- Machine learning & clustering (ml, flann)
- CUDA acceleration (gpu)

## Image-Processing:

Image processing is a method to perform some operations on an image, in order to get an enhanced image and or to extract some useful information from it. If we talk about the basic definition of image processing then **“Image processing is the analysis and manipulation of a digitized image, especially in order to improve its quality”**.

### Digital-Image:

An image may be defined as a two-dimensional function  $f(x, y)$ , where  $x$  and  $y$  are spatial(plane) coordinates, and the amplitude of fat any pair of coordinates  $(x, y)$  is called the intensity or grey level of the image at that point.

In another word An image is nothing more than a two-dimensional matrix (3-D in case of coloured images) which is defined by the mathematical function  $f(x, y)$  at any point is giving the pixel value at that point of an image, the pixel value describes how bright that pixel is, and what colour it should be.

Image processing is basically signal processing in which input is an image and output is image or characteristics according to requirement associated with that image.

### Image processing basically includes the following three steps:

1. Importing the image
2. Analysing and manipulating the image
3. Output in which result can be altered image or report that is based on image analysis

## Web Scraping

Web page scraping can be done using multiple tools or using different frameworks in Python. There are variety of options available for scraping data from a web page, each suiting different needs.

First, let's understand the difference between web-scraping and web-crawling. Web crawling is used to index the information on the page using bots also known as Crawlers. On the other hand, Web-scraping is an automated way of extracting the information/content using bots also known as Scrapers.

Let see some most commonly used web Scraping tools for Python3 :

1. Urllib2
2. Requests
3. BeautifulSoup
4. Lxml
5. Selenium
6. MechanicalSoup

Among all the available frameworks/ tools, only urllib2 come pre-installed with Python. So all other tools need to be installed, if needed. Let's discuss all these tools in detail.

1. **Urllib2** : Urllib2 is a python module used for fetching URL's. It offers a very simple interface, in the form of urlopen function, which is capable of fetching URL's using different protocols like HTTP, FTP etc.
2. **Requests** : Requests does not come pre-installed with Python. Requests allows to send HTTP/1.1 requests. One can add headers, form data, multipart files and parameters with simple Python dictionaries and access the response data in the same way. Installing requests

can be done using pip.

3. **BeautifulSoup** : BeautifulSoup is a parsing library that can use different parsers. BeautifulSoup's default parser comes from Python's standard library. It creates a parse tree that can be used to extract data from HTML; a toolkit for dissecting a document and extracting what you need. It automatically converts incoming documents to Unicode and outgoing documents to UTF-8.

pip can be used to install BeautifulSoup :  
 pip install beautifulsoup4

4. **Lxml** : Lxml is a high-performance, production-quality HTML and XML parsing library. If the user need speed, then go for Lxml. Lxml has many modules and one of the module is etree , which is responsible for creating elements and structure using these elements.

One can start using lxml by installing it as a python package using pip tool :  
 pip install lxml

5. **Selenium** : Some websites use javascript to serve content. For example, they might wait until you scroll down on the page or click a button before loading certain content. For these websites, selenium is needed. Selenium is a tool that automates browsers, also known as web-drivers. It also comes with Python bindings for controlling it right from your application.

pip package is used to install selenium :  
 pip install selenium

6. **MechanicalSoup** : MechanicalSoup is a Python library for automating interaction with websites. It automatically stores and sends cookies, follows redirects, and can follow links and submit forms. It doesn't do JavaScript.

One can use following command to install MechanicalSoup :  
 pip install MechanicalSoup

## ASSIGNMENT-

To scrape data from an online site and collect required information and covert it into DATAFRAME.  
 Code:

```
import pandas as pd
from bs4 import BeautifulSoup
import requests

url='https://www.amazon.in/OnePlus-Mirror-Black-128GB-Storage/product-reviews/B07DJHV6VZ/ref=cm_cr_dp_d_show_all_btm?ie=UTF8&reviewerType=all_reviews'
page=requests.get(url)
#source_data=BeautifulSoup(page.content,'html.parser').find(class_='a-section a-spacing-none reviews-content a-size-base').findAll('div',class_="a-section review aok-relative")
source_data=BeautifulSoup(page.content,'html.parser')
data=source_data.findAll('div',class_="a-section review aok-relative")
names,ratigs,rv_ttls,con_rvs=[],[],[],[]
```



```

for part in data:
    try:
        name=part.a.text
        #print(name)
    except:
        print('...')
    try:
        rating=part.find('span',class_='a-icon-alt').text[:3]
        #print(rating)
    except:
        print('...')
    try:
        sht_rv=part.find(class_='a-size-base a-link-normal review-title
a-color-base review-title-content a-text-bold').text.lstrip().rstrip()
        #print(sht_rv)
    except:
        print('...')
    try:
        rv_con=part.find('span',class_='a-size-base review-text review-
text-content').span.text.rstrip().lstrip()
        # print(rv_con)
    except:
        print('...')
    names.append(name)
    ratigs.append(rating)
    rv_ttls.append(sht_rv)
    con_rvs.append(rv_con)

dict_data={'Name':names,'Ratings':ratigs,'Review Title':rv_ttls,'Review
Content':con_rvs}
print(dict_data,'\n.....\n')
framed_data=pd.DataFrame(dict_data)
print(framed_data)
framed_data.to_csv('Oneplus_Amazon_Reviews.csv')

```

Output:

```

Run: amazon_scrapping x
"C:\Program Files\Python39\python.exe" C:/Users/OCAC/.PyCharmCE2019.3/config/scratches/amazon_scrapping.py
{'Name': ['Tanmay Shukla', 'Surbhi Garg', 'klknow', 'abdulkadir garari', 'Anshu K.', 'nagaraj s.', 'Regidi Mahesh Dora', 'Aakash Sinha', 'Mumtaz Alam', 'Vaibhav Narvaria'], 'Review': ['I got this phone on Friday evening.Pros:Great ...', 'Camera quality is not upto the mark. I visited...', 'I charged the phone completely out of the box ...', 'One plus 6 was costing 28k during the big bill...', 'Good build.amazing battery life. ( Minimum 3 d...', 'Phone is simply superb in all aspects...low li...', 'Two days back I received 6T and had many probl...', 'This has by far been the best phone I have eve...', 'Battery-Extremely PoorCamera-SatisfactoryLook-...', 'After replace the handset issue was resolved. ...'], 'Content': ['...', '...', '...', '...', '...', '...', '...', '...', '...', '...'], '': ['...', '...', '...', '...', '...', '...', '...', '...', '...', '...']}

[10 rows x 4 columns]

Process finished with exit code 0

```

## ASSIGNMENT-2

Scraping data from a weather-site and plotting a graph of temperature for last 7 days.

Code:

```

from bs4 import BeautifulSoup
import matplotlib.pyplot as plt
import pandas as pd
import requests
page=requests.get("https://forecast.weather.gov/MapClick.php?lat=34.0536&lon=-118.2455#.XyFbdJ4zaM8")
soup=BeautifulSoup(page.content,'html.parser')
week=soup.find('div',id="seven-day-forecast-container")
items=week.findAll('div',class_="tombstone-container")
time,short_description,temperature=[],[],[]

#print(items[1])
for data in items:
    day=data.find(class_="period-name").text
    time.append(day)
    sht_note=data.find(class_="short-desc").text
    short_description.append(sht_note)
    temperature.append(data.find(class_="temp").text)
data_final={'Time':time,'Short':short_description,'Temperature':temperature}
framed=pd.DataFrame(data_final)
print(framed)
framed.to_csv('weather.csv')
#print(temperature)
#print(time)

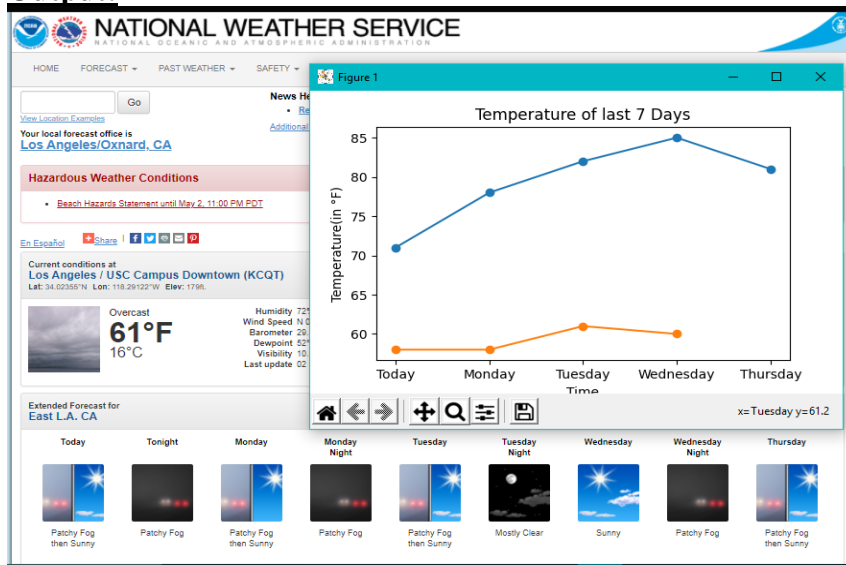
```

```

#-----
maxtemp,mintemp,day=[],[],[]      # Extended project to visualize the
data in graph format
try:n=int(len(temperature)/2)
except:n=int(len(temperature)+1/2)
for i in range(n+1):
    day.append(time[2*i])
for j in range(len(temperature)):
    b=temperature[j].split()
    temp=b[1]
    #temp=temperature[j].strip('HighLowF:°')
    if b[0]=='High:':
        maxtemp.append(int(temp))
    else:
        mintemp.append(int(temp))
daymin=day[:len(mintemp)]
daymax=day[:len(maxtemp)]
fig,ax=plt.subplots(figsize=(6,4))
ax.plot(daymax,maxtemp,marker='o')
ax.plot(daymin,mintemp,marker='o')
ax.set(title='Temperature of last 7 Days',
        xlabel='Time',
        ylabel='Temperature(in °F)')
plt.show()

```

### Output:



## **BASIC PRINCIPLE OF CT SCAN**

A **CT scan**, or **computed tomography scan** is a medical imaging procedure that uses computer-processed combinations of many X-ray measurements taken from different angles to produce cross-sectional (tomographic) images (virtual "slices") of specific areas of a scanned object, allowing the user to see inside the object without cutting.[4]

Internal structure of an object can be reconstructed from multiple projections of the object. X ray beam moves around the patient in a circular path providing a 3D display of the intracranial anatomy built up from a vertical series of transverse series tomograms.

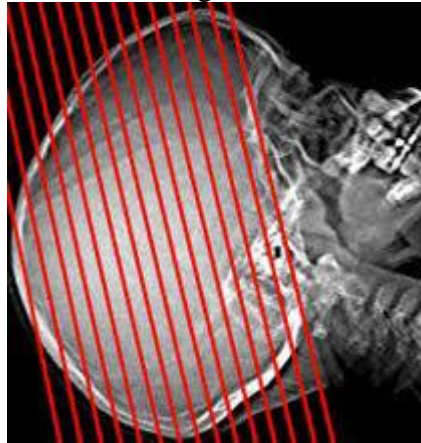


Fig.2. Tomogram slices with thickness of 5 and 10 mm [8]

CT image segmentation Scan head is one of the important efforts undertaken to determine the health conditions in the human brain. In general, segmentation done manually by experts or radiologists, takes a very long time and often finds the error of segmentation results, because the human error itself. Therefore, we need a certain method that can segment the image of the brain quickly and automated. The goal is to obtain the image segmentation of brain bleeding more quickly and accurately. So patients with cerebral hemorrhage can immediately obtain medical treatment in accordance with the needs.

CT Scan is a very useful digital imaging technique, because this technique can provide clear information of the human anatomy. The CT scan scans the nucleus of hydrogen, which is the largest atom in the human body. CT Scan image segmentation becomes the preferred diagnostic method for knowing injuries to the brain, as well as for assessing potential surgical risks or levels of brain treatment.

## **PROPOSED SOLUTION**

In general, segmentation done manually by experts, takes a very long time and often finds the error in segmentation result, because the human error itself. Therefore, we need a certain method that can segment the image of the brain quickly and automated. The goal is to obtain the image segmentation more quickly and accurately.

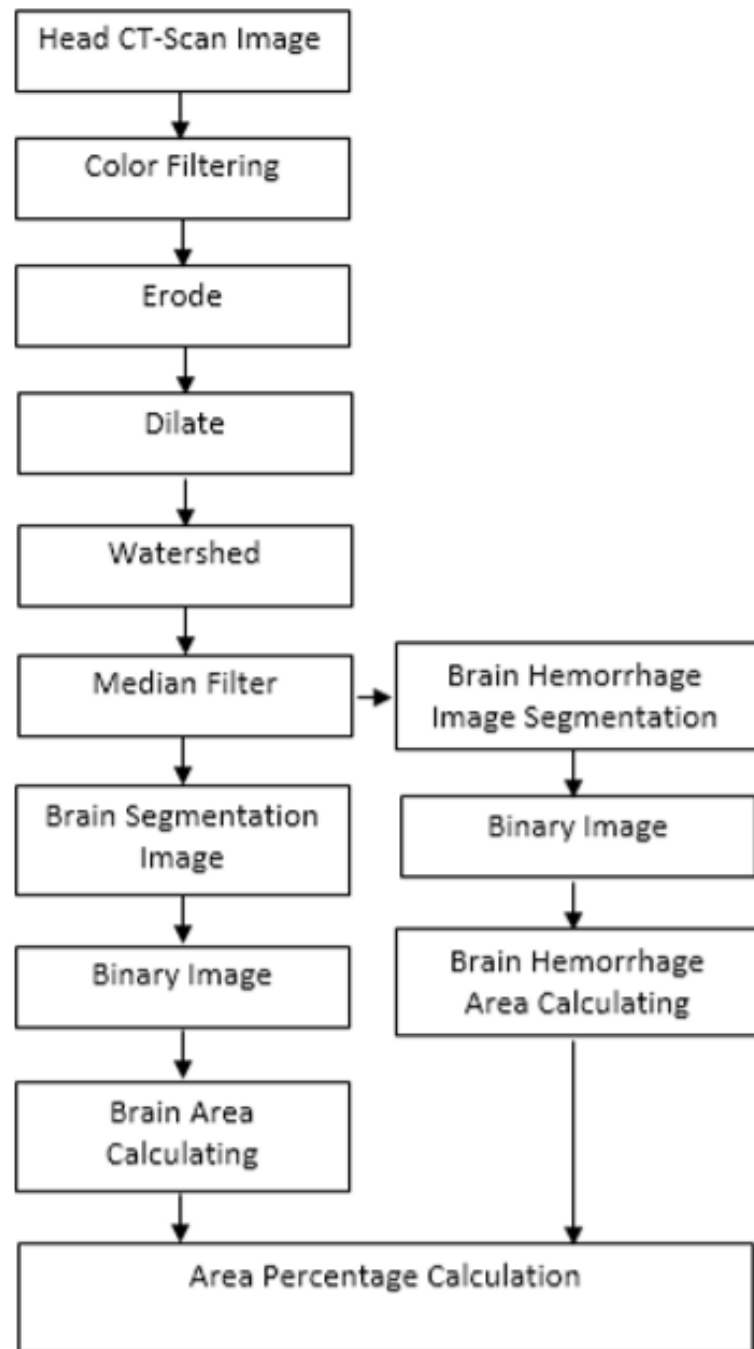


Fig.3. Schematic diagram of the algorithm [1]

Here in this project an attempt has been made to calculate the brain hemorrhage volume by the help of image processing. In this section, we represent the procedures developed segmentation of head CT Scan. The procedure for head CT Scan segmentation which involves color filtering, erosion, dilation, watershed segmentation, crop, median filtering, threshold. The procedure are graphically described in a schematic diagram shown in Fig.3.

### A. Head CT- Scan Image

Almost all CT-Scans were performed in axial plane [2]. The axial plane is selected for clearer image results. In this research, we used axial plane head CT-Scan image. Size of input image is 800 x 662 pixels. Fig. 2 represents the head images in axial plane view.

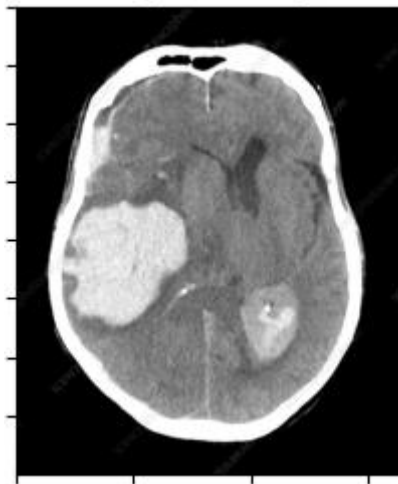


Fig.4. The head image in axial plane [9]

### B. Color Filtering

Color Filtering is an image processing technique used to manipulate an image based on a specific color. The way it works is by comparing the color components of each pixel image with a specific color. If the color corresponds to the specific color of the pixel color component it is left alone. However, if the color does not match the specific color of the pixel color component is changed to the background color, usually a black color. The colors used in Color Filtering can be represented in different color spaces. Fig. 3 represents the result of color filtering.

### C. Erode

Erosion or erosion is one of the image morphology operations that calculates the local minimum value based on the kernel or structuring element area. Image morphology operation is an image processing technique based on the shape or morphology of an image feature. The kernel or structuring element is an  $m \times n$ -sized matrix that has a central point. In general, the erosion process performed on an image produces smaller objects and eliminates the object points as part of the background based on the kernel used. Fig. 4 represents the result of erode operations.

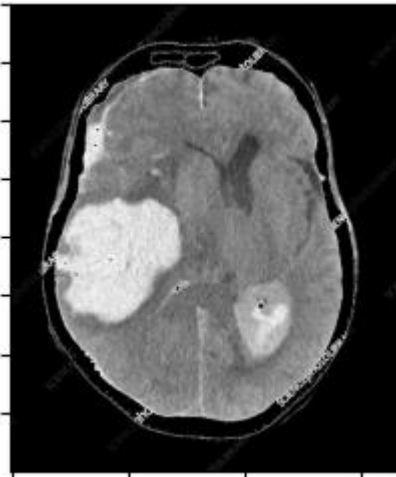


Fig.5. Result of Color Filtering

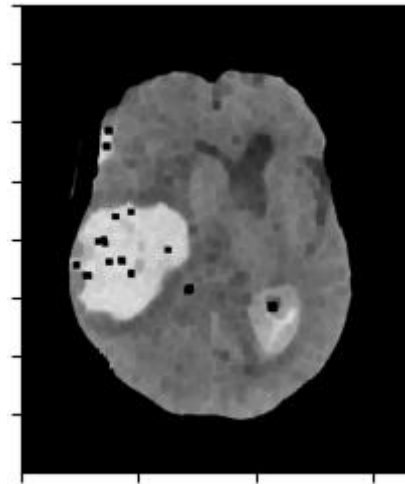


Fig.6. Result of Erode Operations

### D. Dilate

Dilation is the opposite of the erosion process. Dilation or dilation is one of the image morphology operations that calculates the local maximum value based on the kernel or structuring element area. In general, dilation processes performed on an image produce larger objects and merge the object points into parts of the object based on the kernel used. Fig. 5 represents the result of dilate operations.

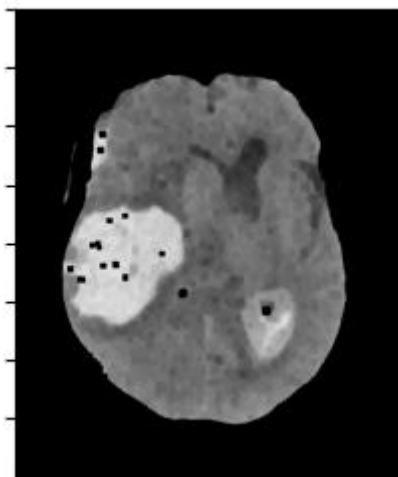


Fig.7. Result of Dilate Operations

### E. Watershed

The concept of Watershed transformation is to assume an image is a three-dimensional form of  $x$  and  $y$  position with each level of color it possesses. The position of  $x$  and  $y$  is the basic plane and pixel color level, which in this case is the ash image (gray level) is an altitude with the assumption that the value that is closer to the white color has a higher height. Assuming the shape of topography, then there are three kinds of points are: 1. The point that is the minimum region 2. The point that is where if a drop of water is dropped, the water will fall down to a certain minimum 3. The point which is where if water is dropped, the water has the possibility to fall into one of the minimum positions (uncertainly falling to a minimum point, but may fall to a certain minimum point or other minimum point). Suppose a gray image of level  $f(x, y)$  is considered a topographic surface  $S$ , where each gray level is considered to be considered terrain elevation, and hill areas correspond to the intended region,

as well as valleys or basins showing a minimum. Suppose that every minimum  $m_1(f)$  is full of holes and the topographic surface  $S$  is depicted vertically into a lake, assumed at a constant velocity. Water will flow and fill the surface. During this filling process, water will come from two or more different minima.

## F. Crop

Subsequently, image will be crop in segmented area (land area in watershed). The output of this step is brain area (without skull). First find the contours with OpenCV's findContours and create a mask with drawContours. Finally copy the masked original image to the new image, which means only the areas of the contours will be copied. Fig. 7 represents the result of crop operations.

## G. Median Filtering

Median filtering is a non-linear method used to eliminate noise in the image. It is widely used because it is very effective for removing noise but still retaining edges. The median filter works by moving through pixels to pixels in an image, replacing each value with the median value of neighboring pixels. Median Filtering is calculated by first sorting all pixel values from window to numerical sequence, and then replacing pixels with pixel middle values. How to find the above median value is:

1. Read the pixel value to be processed along with its neighbor pixels.
2. Sort the pixel values from the smallest to the largest.
3. Select the value in the middle for the new value for pixels (x, y).

Thus, the median filter eliminates very different pixel values with its neighboring pixels. The use of the median filter itself also has a drawback that the processed image will appear slightly blurred or blurred. Fig. 8 represents the result of median filtering.

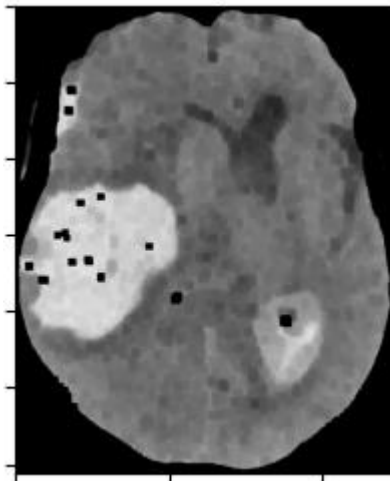


Fig.9. Output image of Crop operation

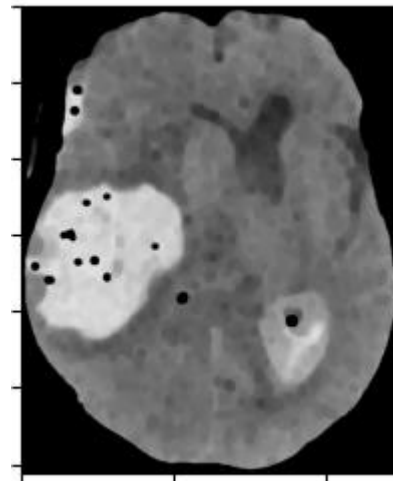


Fig.10. Output image of Median filtering

## H. Threshold Binary

Thresholding is a binary technique used to convert gray images into binary images. Thresholding can be used in the image segmentation process to identify and separate the desired object from the background based on the gray level distribution or image texture. The thresholding process uses a threshold value to change the pixel value of the grayscale image to black or white. Threshold consists of several types, in this research type of threshold used is Threshold Binary. In the binary threshold, if the pixel value in the image is larger than the threshold value, the pixel value will be replaced by the maxval value, otherwise if the pixel image value is smaller than the threshold then the pixel value will



be replaced by 0 (black). Fig. 9 represents the result of threshold binary.

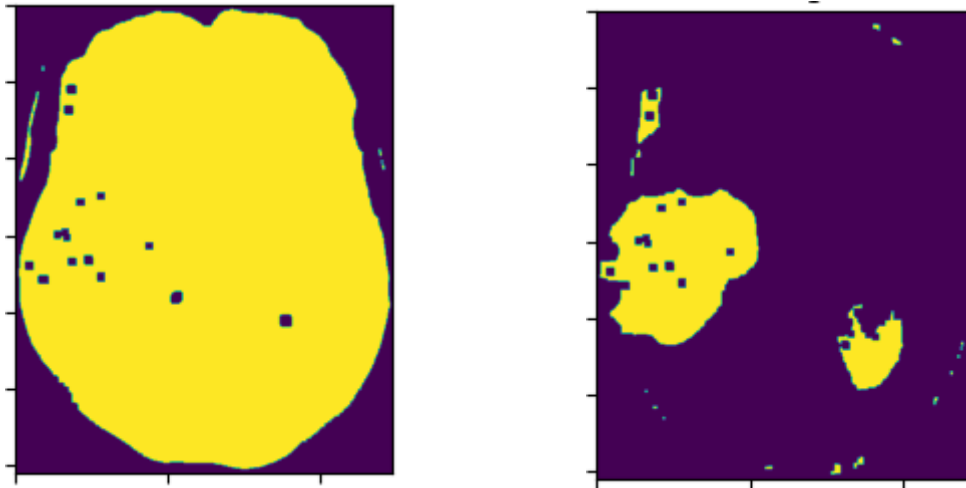


Fig.11. Output image of Threshold Binary: (a)Brain (b)Brain Hemorrhage

### I. Count Pixel of Brain Area & Brain Hemorrhage Area

The calculation of the area of the brain begins by searching for pixels that have a gray value not equal to 0 of the global threshold process. The number of pixels is the area of the brain. The calculation of the area of brain hemorrhage begins by looking for pixels that have a gray value not equal to 0 from the threshold region process. The number of pixels is the area of brain hemorrhage. Fig. 12 represents the result of count pixel of brain area and brain hemorrhage area.

### J. Calculating Volume and Volume Percentage

At this stage the calculation of the volume of bleeding in the brain. The calculation of the volume of brain hemorrhage begins by finding the extent of bleeding on each CT-Scan layer. In this final project CT-Scan image obtained from the hospital amounted to 200 images with a thickness of 0.5 mm. The following are the steps to calculate the volume of bleeding:

1. Calculate the area of brain hemorrhage from the 1st layer to the nth layer by counting the number of pixels on each layer.
2. Perform calculations to convert pixel values into mm units.
3. Get information about the distance between layers / CT Scan image slices. This information will be used as high (t) in volume calculation.
4. Illustrate the use of layer and slice distance on CT-Scan
5. Once we get the width of layer 1 to layer n.
6. Use the volume count formula to get the volume between layers.

$$\text{Volume} = \sum_{i=1}^{n-1} (A_i \cdot t)$$

Information:

V = Volume (mm<sup>3</sup>)

A<sub>i</sub> = Wide Area (mm<sup>2</sup>)

n = Multiple Layers

i = 1, 2, 3, ... n

t = Height (mm)

## EXPERIMENTAL CODE

```

import cv2
import numpy as np
import matplotlib.pyplot as plt

plt.figure(0)
img=cv2.imread("hmrq_brain.jpg")
#img=cv2.imread('normal_brain.jpg')
plt.subplot(2,2,1)
plt.imshow(img)
plt.title('Original Image')

# color filter.....
dim_grey = np.array([0,0,0])
light_grey= np.array([250,250,250])
mask = cv2.inRange(img,dim_grey,light_grey)
clr_fltr=cv2.bitwise_and(img,img,mask=mask)
plt.subplot(2,2,2)
plt.imshow(clr_fltr)
plt.title('color filter')

# Erosion.....
kernel=np.ones((3,3),np.uint8)
ersn=cv2.erode(clr_fltr,kernel,iterations=5)
plt.subplot(2,2,3)
plt.imshow(ersn)
plt.title('Erosion')

# Dialation.....
dltn=cv2.dilate(ersn,kernel,iterations=1)
plt.subplot(2,2,4)
plt.imshow(dltn)
plt.title('Dilation')

plt.figure(1)
# Watershed.....

# cropping.....
gray = cv2.cvtColor(dltn, cv2.COLOR_BGR2GRAY)
gray = cv2.blur(gray, (11,11))
thresh = cv2.threshold(gray, 0, 255, cv2.THRESH_BINARY)[1]
contours,hierarchy = cv2.findContours(thresh, 1, 2)
for cnt in contours:
    x,y,w,h = cv2.boundingRect(cnt)
    if w>100 and h>50:
        break
crop = dltn[y:y+h, x:x+w]

```

```

plt.subplot(2,2,1)
plt.imshow(crop)
plt.title('crop')

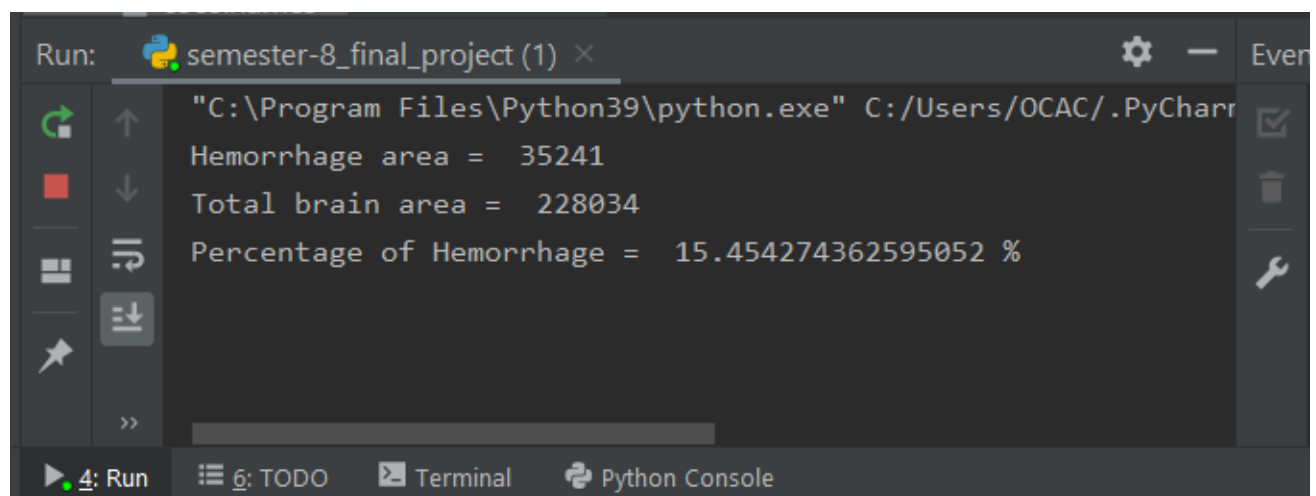
# Median.....
median = cv2.medianBlur(crop, 1)
plt.subplot(2,2,2)
plt.imshow(median)
plt.title('Median')

# Threshold Binary.....
median = cv2.cvtColor(median,cv2.COLOR_BGR2GRAY)
ret1,threshold1 = cv2.threshold(median,150,255,cv2.THRESH_BINARY)
ret3,threshold1 =
cv2.threshold(threshold1,50,255,cv2.THRESH_OTSU+cv2.THRESH_BINARY)
ret4,threshold1 =
cv2.threshold(threshold1,50,255,cv2.THRESH_OTSU+cv2.THRESH_BINARY)
ret2,threshold2 = cv2.threshold(median,0,255,cv2.THRESH_BINARY)
plt.subplot(2,2,3)
plt.imshow(threshold1)
plt.title('Hemorrhage area')
plt.subplot(2,2,4)
plt.imshow(threshold2)
plt.title('Brain Area')

# area count.....
area1 = cv2.countNonZero(threshold1)
area2 = cv2.countNonZero(threshold2)
print("Hemorrhage area = ",area1)
print('Total brain area = ',area2)
print('Percentage of Hemorrhage = ',area1*100/area2,'%')
plt.show()

```

Output:



```

Run: semester-8_final_project (1) x
"C:\Program Files\Python39\python.exe" C:/Users/OCAC/.PyCharm
Hemorrhage area = 35241
Total brain area = 228034
Percentage of Hemorrhage = 15.454274362595052 %

```

Fig.12. Result of count pixel of brain area and brain hemorrhage area

## **EXPERIMENTAL RESULT**

The proposed algorithm was tested using dataset axial head CT-Scan image from internet. The experimental steps consist of error value of wide brain segmentation, error value of wide brain hemorrhage segmentation, percentage area calculation. Segmentation and calculation testing were performed on 200 axial head CT Scan images.

A. Error Value of Wide Brain Segmentation

B. Error Value of Wide Brain Hemorrhage Segmentation

C. Percentage Area Calculation

The volume of bleeding is obtained by calculating the area of brain hemorrhage multiplied by the thickness. The thickness of head CT Scan is 0.5 mm. Then to calculation of the percentage of brain hemorrhage volume to the brain volume as a whole. The ratio of brain hemorrhage volume is calculated using the formula:

$$\text{Ratio} = \frac{\text{Brain Hemorrhage Volume}}{\text{Brain Volume}} \times 100 \%$$

## **CONCLUSION**

Based on the experimental results, before segmenting the CT Scan head, The process of skull cutting with watershed method still has limitations so we removed that process. Cutting done is still not perfect in some parts, especially in the last slice. Disadvantages in skull bone removal affect the calculation of brain area and bleeding area, so there are still some errors. From the system test obtained the calculation of brain area has some error.

## **REFERENCES**

- [1] Rizal Romadhoni Hidayatullah, Riyanto Sigit and Sigit Wasista "Segmentation of Head CT-Scan to Calculate Percentage of Brain Hemorrhage Volume' ", *International Electronics Symposium on Knowledge Creation and Intelligent Computing (IES-KCIC)*, 2017
- [2] Nouredine Rasha, Tarhini Khalil, Saleh Soha, "Segmentation and Extraction of Brain Injury Lesions from MRI Images: Matlab Implementatio", *International Conference on Advances in Biomedical Engineering (ICABME)*, 2015.
- [3] Dawngliana Malsawm, Deb Daizy, Handique Mousum, and Roy Sudipta, "Automatic Brain Tumor Segmentation in MRI: Hybridized Multilevel Thresholding and Level Set", *International Symposium on Advanced Computing and Communication (ISACC)*, 2015
- [4] [https://en.wikipedia.org/wiki/CT\\_scan#:~:text=A%20CT%20scan%2C%20or%20computed,of%20specific%20areas%20of%20a](https://en.wikipedia.org/wiki/CT_scan#:~:text=A%20CT%20scan%2C%20or%20computed,of%20specific%20areas%20of%20a)
- [5] [https://en.wikipedia.org/wiki/CT\\_scan#/media/File:UPMCEast\\_CTscan.jpg](https://en.wikipedia.org/wiki/CT_scan#/media/File:UPMCEast_CTscan.jpg)
- [6] [https://www.google.com/search?q=history+of+ct+scan&rlz=1C1CHBF\\_enIN916IN916&oq=history+of+ct+scan&aqs=chrome..69i57.9596j0j1&sourceid=chrome&ie=UTF-8](https://www.google.com/search?q=history+of+ct+scan&rlz=1C1CHBF_enIN916IN916&oq=history+of+ct+scan&aqs=chrome..69i57.9596j0j1&sourceid=chrome&ie=UTF-8) ,
- [7] <https://i.pinimg.com/originals/56/dc/4e/56dc4e9eaf119ea63161c4a01ba6f141.png> ,
- [8] <https://www.slideshare.net/Drraveesoni/ct-scan-basics-53553679>
- [9] <https://images.app.goo.gl/1Gu2pHPmc4Qg16gj7>



# CERTIFICATE OF COMPLETION

is presented to

**Ankita Jena**

for successfully completing the month long online training of

**Python ( Basic )**

**Anubha Maneshwar**

Founder

GirlScript Technologies Pvt. Ltd

**09-03-2021**

**Issued Date**



# CERTIFICATE OF COMPLETION

is presented to

**Ankita Jena**

for successfully completing the month long online training of

**Python ( Advanced )**

**Anubha Maneshwar**

Founder

GirlScript Technologies Pvt. Ltd

**15-04-2021**

**Issued Date**





# CERTIFICATE OF COMPLETION

is presented to

**SWADESH KUMAR NATH**

for successfully completing the month long online training of

Python ( Advanced )

---

**Anubha Maneshwar**

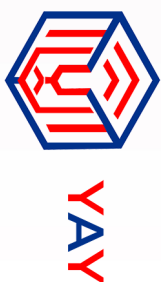
Founder

GirlScript Technologies Pvt. Ltd

---

**15-04-2021**

**Issued Date**



# CERTIFICATE OF COMPLETION

is presented to

**SWADESH KUMAR NATH**

for successfully completing the month long online training of

**Python ( Basic )**

---

**Anubha Maneshwar**

Founder

GirlScript Technologies Pvt. Ltd

---

**09-03-2021**

**Issued Date**

**TO WHOM SO EVER IT MAY CONCERN**

This is to certify that Ankita Jena, has undergone an internship on "Python and Data Science" from February 2021 to May 2021 for meeting his/her industrial internship requirements. He/She is being imparted industrial training and advanced software training by our training/production faculties to get exposure to the practical aspects in industries.

To the best of our knowledge and belief, he/she bears a good moral character.



Mohit Varu

Managing Director

GirlScript Technologies Pvt. Ltd.

Email : [mohit@girlscript.tech](mailto:mohit@girlscript.tech)

**TO WHOM SO EVER IT MAY CONCERN**

This is to certify that Swadesh Kumar Nath, has undergone an internship on "Python and Data Science" from February 2021 to May 2021 for meeting his/her industrial internship requirements. He/She is being imparted industrial training and advanced software training by our training/production faculties to get exposure to the practical aspects in industries.

To the best of our knowledge and belief, he/she bears a good moral character.



Mohit Varu

Managing Director

GirlScript Technologies Pvt. Ltd.

Email : [mohit@girlscript.tech](mailto:mohit@girlscript.tech)