Day-1 Task

Q1.Install Python on your Local Operating System

a)https://youtu.be/M5ILgNI0iXw (https://youtu.be/M5ILgNI0iXw) (python installation)

b)https://youtu.be/SBOFnzVuzOE (https://youtu.be/SBOFnzVuzOE) (vim Editor)

Ans: Installed

Q2.List Some Mega Projects Hosted in Python.

Ans:- a)Database Projects: Management Systems (Library, Hotel, Hospital, School) b)ML Projects: Classification c)AI Projects: Chatbots, face detection and recognition, speech recognition, Security d)Data Science Projects: Global Terrorism Analysis, Call Data Record Analysis

Q3.List Python Libraries Used in Different IT Domains

Ans:- Pandas, numpy, matplotlib, tkinter, sklearn, tensorflow.

In [7]:
```
#Q4. Create a Program and execute a script in Python

print("Hello World")

a = 45

b = 5

c = a+b

print( "Sum is:", c)
```

```
Hello World
Sum is: 50
```

Python Tasks - Day2

Q1. Install vim and execute a hello world script using powershell

a) Python installation link - https://youtu.be/M5ILgNI0iXw (first 20 minutes)

b) Vim Installation link - https://youtu.be/SBOFnzVuzOE

Ans:- vim hello.py

import os os.system("cls") print("\n\n\n\n\n\n") print("Hello World.Welcome to vim tutorial.".center(100)) print("\n\n\n\n\n\n")

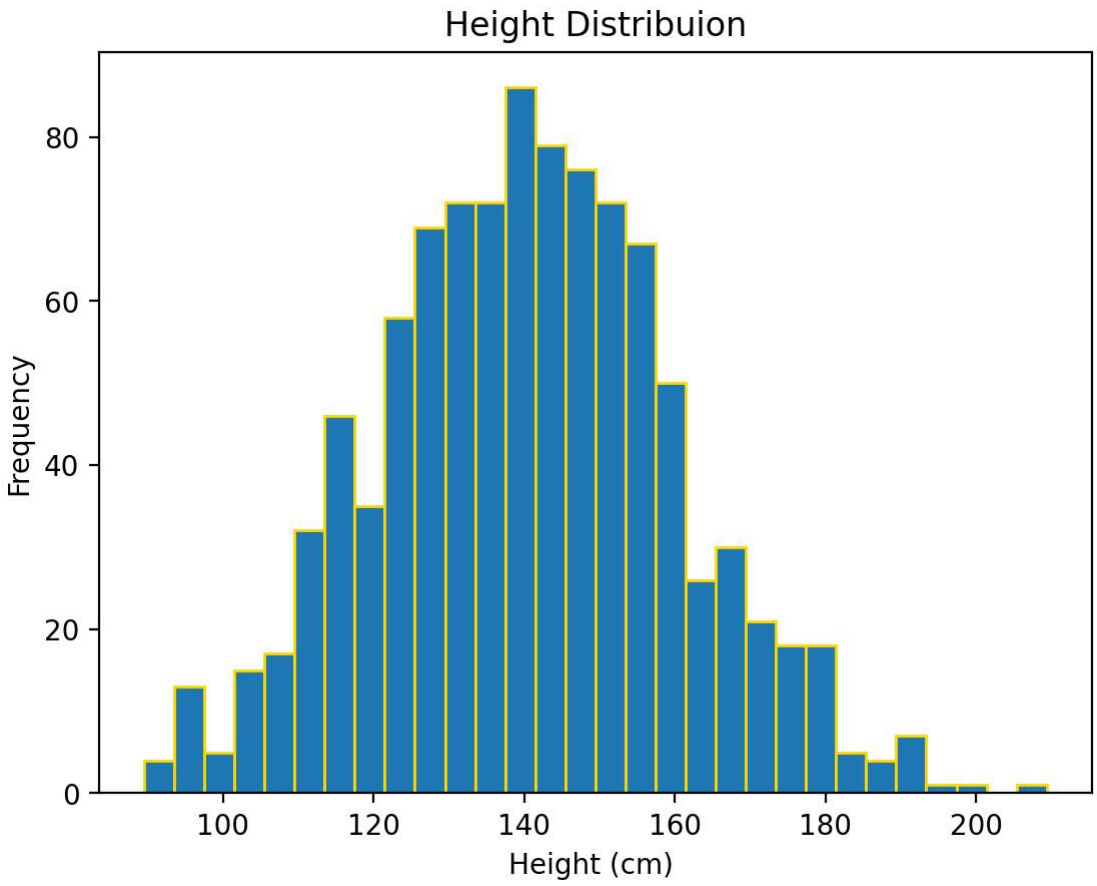
Esc :w-> save :q-> window close

PS C:users\KIIT\Desktop\ML class\vimm>Python hello.py

```
Hello World. Welcome to vim tutorial
```

In [10]:
```
#Q2
import numpy as np

import matplotlib.pyplot as plt

plt.figure(dpi=200)

height = np.random.normal(140, 20, 1000)
plt.ylabel("Frequency")
plt.hist(height, bins=30, ec="gold")

plt.title("Height Distribuion")
plt.xlabel("Height (cm) ")
```

Out[10]:
```
Text(0.5, 0, 'Height (cm) ')
```



Q3. Write down above code using vim on same working directory, and execute it using python interpreter, tell us difference between jupyter and notepad .

Ans:- Difference between jupyter and notepad are:- Notepad is a source code editor that supports several languages. Running in the MS Windows environment whereas the Jupyter Notebook is a web based application for creating computational documents. It supports several languages like Python (IPython), Julia, R etc. and is largely used for data analysis, data visualization and further interactive, exploratory computing.

Q4. Write Down difference between interpreter and compiler.

Ans:- Difference between interpreter and compiler are:-

Interpreter translates program one statement at a time.It usually take less amount of time to analyze the source code. No object code is generated,hence are memory efficient.Programming languages like Javascript,Python,Ruby use in interpreters.

whereas Comppiler scans the entire program and translates it as a whole into machine code.It usually take a large amount of time to analyze the source code.It generates object code which further requires linking,hence requires more memory. Programming languages like C,C++,Java use compilers.

Q5.What is Data Structure, why we need one in programs?
Ans:- A data structure is a specialized format for organizing, processing, retrieving and storing data. Data structures make it easy for users to access and work with the data they need in

Python Tasks - Day3

Q1. What are Data Structures? List some of data structures and their use in real world application?

Ans:- A data structure is a method of arranging data in order to make it more useful. It provides efficiency, reusability and abstraction. Some data structures are:

1)Linear Data Structure

a) Array:-

i)Storing the contacts on out phone.

ii) CPU scheduling in computers.

b) Stack:-

i)It is used in virtual machines like JVM.

ii) Forward-backward surfing in browser.

c) Queue:-

i) Operating system uses queues for job scheduling.

ii)Escalators,Printer spooler,car washes queue.

d) Linked List:-

i)Train coaches are connected to one another in doubly-linke
d list fashion.

In [12]:
```
# Q3. Create a list data type and store names of your friends in it (at le
# in list data type, try to figure out their working using help function i

# Ans:-

list=["Ram","Riya","Suman","Rani","Diya"]
print(list)
```

```
['Ram', 'Riya', 'Suman', 'Rani', 'Diya']
```

In [13]:
```
# Working using help function in python
```

In [14]:
```
help(list.append)
```

```
Help on built-in function append:

append(object, /) method of builtins.list instance
    Append object to the end of the list.
```

In [15]:
```
help(list.insert)
```

```
Help on built-in function insert:

insert(index, object, /) method of builtins.list instance
    Insert object before index.
```

In [16]:
```
help(list.extend)
```

```
Help on built-in function extend:

extend(iterable, /) method of builtins.list instance
    Extend list by appending elements from the iterable.
```

In [17]:
```
help(list.pop)
```

```
Help on built-in function pop:

pop(index=-1, /) method of builtins.list instance
    Remove and return item at index (default last).

    Raises IndexError if list is empty or index is out of range.
```

In [18]:
```
help(list.remove)
```

```
Help on built-in function remove:

remove(value, /) method of builtins.list instance
    Remove first occurrence of value.

    Raises ValueError if the value is not present.
```

In [19]:
```
help(list.sort)
```

```
Help on built-in function sort:

sort(*, key=None, reverse=False) method of builtins.list instance
    Sort the list in ascending order and return None.

    The sort is in-place (i.e. the list itself is modified) and stable
(i.e. the
    order of two equal elements is maintained).

    If a key function is given, apply it once to each list item and sort
them,
    ascending or descending, according to their function values.

    The reverse flag can be set to sort in descending order.
```

In [20]:
```
help(list.clear)
```

```
Help on built-in function clear:

clear() method of builtins.list instance
    Remove all items from list.
```

In [21]:
```
help(list.count)
```

```
Help on built-in function count:

count(value, /) method of builtins.list instance
    Return number of occurrences of value.
```

In [22]:
```
help(list.index)
```

```
Help on built-in function index:

index(value, start=0, stop=9223372036854775807, /) method of builtins.li
st instance
    Return first index of value.

    Raises ValueError if the value is not present.
```

In [24]:
```
help(list.copy)
```

```
Help on built-in function copy:

copy() method of builtins.list instance
    Return a shallow copy of the list.
```

Q4. What is difference between ordered data type and unordered data type?
Ans:- An ordered data type means the elements of the collection have a specific order. The order is independent of the value. Examples:- Lists,Strings and tuples. An unordered data type means that not only does the collection have order,but the order depends on the value of the element. Examples:- Sets and Dictionaries.

Q5.Write down types of each value given? (In python) Ans:-

a) 100 --> Integer

b) 105.5 --> Float

c) 192.56j --> Complex

d) 10+6j --> Complex

e) '10' --> String

f) 'Hello world' --> String

g)[10,20,50,100 --> List

h) {'name': 'sachin', 'age': 24, 'language': 'python'} --> Dictionary

In [1]:
```
#                                                    Python Tasks - Day4
# Q1. Explore split, strip, replace, center, title methods of string data

# Ans:-
str = '  Hi Everyone!How are you?    '

print(str.split()) #Return a list of the words in the string, using separa

print(str.strip()) #delete all the leading and trailing spaces

print(str.replace('Everyone','Rahul')) #replace the substring with a new s

print(str.center(100)) #center align the string, using a specified charact

print(str.title()) #returns a string where the first character in every wo
```

```
['Hi', 'Everyone!How', 'are', 'you?']
Hi Everyone!How are you?
  Hi Rahul!How are you?
                                        Hi Everyone!How are you?
  Hi Everyone!How Are You?
```

In [2]:
```
# Q2. Explore append, pop, remove, sort methods of list data type
# Ans:-
data = [25, 65, 90, 27, 17]

data.append(45) #add item from the last
print(data)

data.pop() #delete item from the last
print(data)

data.remove(65) #remove the matching item from the list
print(data)

data.sort() #arrange item in ascending or descending order
print(data)
```

```
[25, 65, 90, 27, 17, 45]
[25, 65, 90, 27, 17]
[25, 90, 27, 17]
[17, 25, 27, 90]
```

In [3]:
```
# Q3.  Create 5 real time lists to store some useful information in python
# Ans:-

cars=['BMW','Tata','Hyuindai']
Biscuits=['Oreo','Parle G','Sunfeast']
Cakes=['Choclate','Britannia']
Laptops=['HP','Dell','Lenovo','Asus']
Mobiles=['Samsung','Appple','Oppo','Vivo']
print(cars)
print(Biscuits)
print(Cakes)
print(Laptops)
print(Mobiles)
```

```
['BMW', 'Tata', 'Hyuindai']
['Oreo', 'Parle G', 'Sunfeast']
['Choclate', 'Britannia']
['HP', 'Dell', 'Lenovo', 'Asus']
['Samsung', 'Appple', 'Oppo', 'Vivo']
```

Q5. What is difference between mutable and immutable data types in python

Ans:-

A mutable data type is an object whose state can be modified after it is defined.Examples: Lists,Dictionaries,Sets etc whereas Immutable datatypes are objects that cannot be modified or altered after they have been created. Examples:- int,float,bool,string,tuple etc.

Q6.What are identifiers, list rules of identifiers in python.

Ans:-

An identifier is a user-defined name given to identities like class, functions, variables, modules, or any other object in Python.

```
Rules of identifiers are:

   1) The identifier is a combination of character digits and undersc
ore and the character includes letters in
      lowercase (a-z), letters in uppercase (A-Z), digits (0-9), and
an underscore (_).
```

Python Tasks – Day5

Q1. What is difference between shallow copy and deep copy?
Ans:- A shallow copy constructs a new compound object and then (to the extent possible) inserts references into it to the objects found in the original. A deep copy constructs a new compound object and then, recursively, inserts copies into it of the objects found in the original.

In [11]:
```
# Q2. You want to create a library management application, you need to sto

# Ans:-

books = [
    {'title': 'Wings on Fire', 'author': 'APJ Kalam', 'isbn': 'ABCD123'},
    {'title': 'Awesome Python', 'author': 'Guido Van Rossum', 'isbn': 'GHJ
    {'title': 'The Story of My life', 'author': 'Helen Keller', 'isbn': 'P
    # Add more books as needed"
]

# List to store information about students
students = [
    {'name': 'Rani', 'id': 'ID 1', 'year': 1},
    {'name': 'Raja', 'id': 'ID 2', 'year': 2},
    {'name': 'Rima', 'id': 'ID 3', 'year': 3},
    # Add more students as needed
]

print(books)
print(students)
```

```
[{'title': 'Wings on Fire', 'author': 'APJ Kalam', 'isbn': 'ABCD123'},
{'title': 'Awesome Python', 'author': 'Guido Van Rossum', 'isbn': 'GHJ12
3'}, {'title': 'The Story of My life', 'author': 'Helen Keller', 'isbn':
'POI675'}]
[{'name': 'Rani', 'id': 'ID 1', 'year': 1}, {'name': 'Raja', 'id': 'ID
2', 'year': 2}, {'name': 'Rima', 'id': 'ID 3', 'year': 3}]
```

Q3. This chapter introduced assignment statements, like spam = 10. What is the difference between an expression and a statement?

Ans:-

Expressions only contain identifiers, literals and operators, where operators include arithmetic and boolean operators, the function call operator () and similar, and can be reduced to some kind of "value", which can be any Python object. Statements on the other hand, are everything that can make up a line (or several lines) of Python code. Note that expressions are statements as well.

Q4. Define following

```
a) Atomic data types / Primary data types

b) Secondary data type / User Defined Data Type
```

Ans:- a) Atomic data types / Primary data types --> A primitive data type is either a data type that is built into a programming language, or one that could be characterized as a basic structure for building more sophisticated data types.

b) Secondary data type / User Defined Data Type --> A user-defined data type is a data type that is defined by the programmer rather than being built into the programming language. It allows programmers to create their own data types that are tailored to the specific needs of their programs.

Q5. What is UDF?

Ans:- UDF stands for User-Defined Function. An UDF is a function that is created by the user

In [ ]: