Lab 1

Name: Aryan Dilipbhai Langhanoja
Date:
Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

##POST LAB##

AIM:- Write a program to demonstrate different number datatype in python

Datatype:- Data types specify the type of data that can be stored inside a variable.

Data Types	Classes	Description
Numeric	int, float, complex	holds numeric values
String	str	holds sequence of characters
Sequence	list, tuple, range	holds collection of items
Mapping	dict	holds data in key-value pair form
Boolean	bool	holds either true or false
Set	set, frozeenset	hold collection of unique items

Task1: Print interger, float, string and boolean with their types in python.

Python Code:

```
integer_variable = 30
float_variable = 1.7812
string_variable = "Aryan Langhanoja"
boolean_variable = True
print("Integer:", integer_variable, type(integer_variable))
print("Float:", float_variable, type(float_variable))
print("String:", string_variable, type(string_variable))
print("Boolean:", boolean_variable, type(boolean_variable))
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
 Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Integer: 30 <class 'int'>
Float: 1.7812 <class 'float'>
String: Aryan Langhanoja <class 'str'>
Boolean: True <class 'bool'>
```

Task2: Print list and tuple with their types in python.

Python Code:

```
list_variable = [1, 2, 3, 4, 5]
tuple_variable = (6, 7, 8, 9, 10)
print("List:", list_variable, type(list_variable))
print("Tuple:", tuple_variable, type(tuple_variable))
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
 Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
List: [1, 2, 3, 4, 5] <class 'list'>
Tuple: (6, 7, 8, 9, 10) <class 'tuple'>
```

Task3: Print dictionary and set with their types in python.

Python Code:

```
dictionary_variable = {
   "Name": "Aryan Langhanoja", "Age": 19, "City": "Ahmedabad"}
set_variable = {5, 6, 7, 8, 9, 10}
print("Dictionary:", dictionary_variable, type(dictionary_variable))
print("Set:", set_variable, type(set_variable))
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work> & D:/DLLs/Anaconda/python.exe
 Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Dictionary: {'Name': 'Aryan Langhanoja', 'Age': 19, 'City': 'Ahmedabad'} <class 'dict'>
Set: {5, 6, 7, 8, 9, 10} <class 'set'>
```

Task4: Python program to demonstrate data type of given number

Python Code:

```
a = 5
print("Type of a: ", type(a))
b = 5.0
print("Type of b: ", type(b))
c = 2 + 4j
print("Type of c: ", type(c))
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
 Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Type of a:  <class 'int'>
Type of b:  <class 'float'>
Type of c:  <class 'complex'>
```

Task5: Python program to demonstrate data type of given number

Python Code:

```
tuple1 = tuple([1, 2, 3, 4, 5])
print("First of tuple")
print(tuple1[0])
print("Last of tuple")
print(tuple1[-1])
print("Third of tuple")
print(tuple1[-3])
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
 Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
First of tuple
1
Last of tuple
5
Third of tuple
3
```

Lab 2

Name: Aryan Dilipbhai Langhanoja

Date: 03-11-2023

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditionals, loops and functions

AIM:- Write a program to perform different arithmetic operations on numbers in python

Arithmetic Operations:- Mathematical calculations that we can perform on numeric values such as integers and floating-point numbers.

Operator	Description	Syntax
+	Addition: adds two operands	x + y
–	Subtraction: subtracts two operands	x – y
*	Multiplication: multiplies two operands	x * y
/	Division (float): divides the first operand by the second	x / y
//	Division (floor): divides the first operand by the second	x // y
%	Modulus: returns the remainder when the first operand is divided by the second	x % y
**	Power (Exponent): Returns first raised to power second	x ** y

Task1: Print arithmetic operations for two values.

Python Code:

```
val1 = 5
val2 = 3
res = val1 + val2
print(res)
val1 = 5
val2 = 3
res = val1 - val2
```

```
print(res)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe '
8
2
```

Task2: Print arithmetic operations for two values.

Python Code:

```
val1 = 5
val2 = 3
res = val1 * val2
print(res)
val1 = 5
val2 = 2
res = val1 / val2
print(res)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
15
2.5
```

Task3: Print arithmetic operations for two values.

Python Code:

```
val1 = 5
val2 = 2
res = val1 % val2
print(res)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
1
```

Task4: Print arithmetic operations for two values.

Python Code:

```
val1 = 5
val2 = 3
res = val1 ** val2
```

```
print(res)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
125
```

Task5: Print arithmetic operations for two values.

Python Code:

```
val1 = 5
val2 = 2
res = val1 // val2
print(res)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
2
```

Lab 3

Name: Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditionals, loops and functions

AIM:- Write a program to create, concatenate and print a string and accessing sub string from a given string.

Introduction:- String Concatenation is the technique of combining two strings.

Task1: String Concatenation using + operator in python

Python Code:

```
var1 = "Aryan "
var2 = "Langhanoja"
var3 = var1 + var2
print(var3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Aryan Langhanoja
```

Task2: String Concatenation using join() method in python

Python Code:

```
var1 = "Aryan "
var2 = "Langhanoja"
print("".join([var1, var2]))
var3 = " ".join([var1, var2])
print(var3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Aryan Langhanoja
Aryan  Langhanoja
```

Task3: String Concatenation using % operator in python

Python Code:

```
var1 = "Aryan "
var2 = "Langhanoja "
print("% s % s" % (var1, var2))
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Aryan  Langhanoja
```

Task4: String Concatenation using format() function in python

Python Code:

```
var1 = "Aryan"
var2 = "Langhanoja"
print("{} {}".format(var1, var2))
var3 = "{} {}".format(var1, var2)
print(var3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Aryan Langhanoja
Aryan Langhanoja
```

Task5: String Concatenation using “,” in python

Python Code:

```
var1 = "Aryan"
var2 = "Dilipbhai"
var3 = "Langhanoja"
print(var1, var2, var3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Aryan Dilipbhai Langhanoja
```

Lab 4

Name: **Date:**

Enrollment No:

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditionals, loops and functions

AIM:- Write a program script to print the current date in following format "Sat August 26 22:12:59 IST 2023"

Introduction:- Pytz brings the Olson tz database into Python and thus supports almost all time zones. This module serves the date-time conversion functionalities and helps user serving international client's base. It enables time-zone calculations in our Python applications and also allows us to create timezone aware date time instances.

Task1: Write a program script to print the current date in following format "Sat August 26 22:12:59 IST 2023"

Python Code:

```
from datetime import datetime
import pytz
format_string = "%A, %d %B %Y %I:%M %p (%Z)"
desired_timezone = pytz.timezone('America/New_York')
curr_time = datetime.now(desired_timezone)
formatted_time = curr_time.strftime(format_string)
print(formatted_time)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Friday, 03 November 2023 08:46 PM (EDT)
```

Task2: It returns the list and set of commonly used timezones with pytz.common_timezones, pytz.common_timezones_set.

Python Code:

```
import pytz
print('Commonly used time-zones:',
   pytz.common_timezones, '\n')
print('Commonly used time-zones-set:',
   pytz.common_timezones_set, '\n')
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Commonly used time-zones-set: LazySet({'America/Chihuahua', 'America/Whitehorse', 'Africa/Djibouti', 'Africa/Maseru', 'Africa/Harare', 'Europe/Rome', 'Indian/Mayo
tte', 'America/Marigot', 'Europe/San_Marino', 'America/Kentucky/Monticello', 'Asia/Jayapura', 'Asia/Hong_Kong', 'America/Rankin_Inlet', 'Pacific/Efate', 'Europe/S
kopje', 'Europe/Moscow', 'Africa/Kampala', 'Asia/Manila', 'Asia/Kuala_Lumpur', 'Pacific/Pitcairn', 'America/Danmarkshavn', 'America/Mexico_City', 'Asia/Kathmandu'
, 'America/Guatemala', 'Asia/Ho_Chi_Minh', 'Pacific/Chatham', 'Europe/Busingen', 'America/Manaus', 'America/Tortola', 'America/Matamoros', 'Africa/El_Aaiun', 'Afr
ica/Dar_es_Salaam', 'Africa/Lubumbashi', 'Europe/Isle_of_Man', 'America/Araguaina', 'America/Port_of_Spain', 'America/Argentina/Rio_Gallegos', 'Asia/Vientiane', '
Pacific/Niue', 'Africa/Niamey', 'America/Phoenix', 'Asia/Aden', 'America/Halifax', 'America/Asuncion', 'Europe/Bratislava', 'Europe/Warsaw', 'America/La_Paz', 'As
ia/Macau', 'America/Bahia', 'Indian/Mauritius', 'US/Pacific', 'America/Panama', 'Antarctica/Davis', 'Africa/Libreville', 'America/Grenada', 'Europe/Berlin', 'Euro
pe/Malta', 'Europe/Vienna', 'Asia/Singapore', 'Canada/Newfoundland', 'Pacific/Apia', 'Europe/Guernsey', 'America/Denver', 'America/Argentina/San_Luis', 'Europe/Ul
yanovsk', 'Europe/Kirov', 'Australia/Lindeman', 'America/Indiana/Knox', 'Europe/Lisbon', 'America/Chicago', 'Africa/Lusaka', 'Africa/Juba', 'America/Detroit', 'Af
rica/Accra', 'Antarctica/DumontDUrville', 'Asia/Baku', 'Asia/Dhaka', 'America/Winnipeg', 'Pacific/Pago_Pago', 'Europe/Belgrade', 'Pacific/Kosrae', 'America/St_Bar
thelemy', 'Europe/Sofia', 'Pacific/Tarawa', 'Asia/Riyadh', 'America/El_Salvador', 'Atlantic/Madeira', 'Asia/Oral', 'US/Central', 'America/Lima', 'Europe/Tirane',
'America/Moncton', 'America/Argentina/Jujuy', 'America/Dawson', 'America/Cancun', 'Asia/Muscat', 'Canada/Eastern', 'Europe/Sarajevo', 'Africa/Lome', 'America/Thul
e', 'America/Mazatlan', 'Canada/Pacific', 'Indian/Maldives', 'Africa/Mbabane', 'Pacific/Honolulu', 'Atlantic/Azores', 'America/Creston', 'America/Hermosillo', 'Af
rica/Casablanca', 'Antarctica/McMurdo', 'US/Alaska', 'Australia/Melbourne', 'US/Arizona', 'Asia/Almaty', 'Africa/Bujumbura', 'GMT', 'Africa/Malabo', 'Europe/Podgo
rica', 'America/Barbados', 'Asia/Kuwait', 'America/Bahia_Banderas', 'Asia/Tomsk', 'America/Rio_Branco', 'Africa/Asmara', 'Africa/Monrovia', 'America/Indiana/Winam
ac', 'America/Cuiaba', 'Europe/Jersey', 'America/Anchorage', 'Canada/Central', 'Pacific/Kiritimati', 'Asia/Taipei', 'Europe/Amsterdam', 'Pacific/Majuro', 'Asia/Om
sk', 'Atlantic/Bermuda', 'Arctic/Longyearbyen', 'Pacific/Marquesas', 'America/Martinique', 'Africa/Abidjan', 'Asia/Nicosia', 'America/Montserrat', 'Indian/Reunion
', 'America/Indiana/Petersburg', 'Australia/Darwin', 'America/Cambridge_Bay', 'Europe/Astrakhan', 'America/Kralendijk', 'Asia/Yangon', 'America/Belem', 'Asia/Atyr
```

Task3: Libraries returns the list all the available timezones with pytz.all_timezones.

Python Code:

```
import pytz
print('the supported timezones by the pytz module:',
    pytz.all_timezones, '\n')
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
the supported timezones by the pytz module: ['Africa/Abidjan', 'Africa/Accra', 'Africa/Addis_Ababa', 'Africa/Algiers', 'Africa/Asmara', 'Africa/Asmera', 'Africa/B
amako', 'Africa/Bangui', 'Africa/Banjul', 'Africa/Bissau', 'Africa/Blantyre', 'Africa/Brazzaville', 'Africa/Bujumbura', 'Africa/Cairo', 'Africa/Casablanca', 'Afri
ca/Ceuta', 'Africa/Conakry', 'Africa/Dakar', 'Africa/Dar_es_Salaam', 'Africa/Djibouti', 'Africa/Douala', 'Africa/El_Aaiun', 'Africa/Freetown', 'Africa/Gaborone',
'Africa/Harare', 'Africa/Johannesburg', 'Africa/Juba', 'Africa/Kampala', 'Africa/Khartoum', 'Africa/Kigali', 'Africa/Kinshasa', 'Africa/Lagos', 'Africa/Libreville
', 'Africa/Lome', 'Africa/Luanda', 'Africa/Lubumbashi', 'Africa/Lusaka', 'Africa/Malabo', 'Africa/Maputo', 'Africa/Maseru', 'Africa/Mbabane', 'Africa/Mogadishu',
'Africa/Monrovia', 'Africa/Nairobi', 'Africa/Ndjamena', 'Africa/Niamey', 'Africa/Nouakchott', 'Africa/Ouagadougou', 'Africa/Porto-Novo', 'Africa/Sao_Tome', 'Afric
a/Timbuktu', 'Africa/Tripoli', 'Africa/Tunis', 'Africa/Windhoek', 'America/Adak', 'America/Anchorage', 'America/Anguilla', 'America/Antigua', 'America/Araguaina',
 'America/Argentina/Buenos_Aires', 'America/Argentina/Catamarca', 'America/Argentina/ComodRivadavia', 'America/Argentina/Cordoba', 'America/Argentina/Jujuy', 'Ame
rica/Argentina/La_Rioja', 'America/Argentina/Mendoza', 'America/Argentina/Rio_Gallegos', 'America/Argentina/Salta', 'America/Argentina/San_Juan', 'America/Argenti
na/San_Luis', 'America/Argentina/Tucuman', 'America/Argentina/Ushuaia', 'America/Aruba', 'America/Asuncion', 'America/Atikokan', 'America/Atka', 'America/Bahia',
'America/Bahia_Banderas', 'America/Barbados', 'America/Belem', 'America/Belize', 'America/Blanc-Sablon', 'America/Boa_Vista', 'America/Bogota', 'America/Boise', '
America/Buenos_Aires', 'America/Cambridge_Bay', 'America/Campo_Grande', 'America/Cancun', 'America/Caracas', 'America/Catamarca', 'America/Cayenne', 'America/Caym
an', 'America/Chicago', 'America/Chihuahua', 'America/Ciudad_Juarez', 'America/Coral_Harbour', 'America/Cordoba', 'America/Costa_Rica', 'America/Creston', 'Americ
a/Cuiaba', 'America/Curacao', 'America/Danmarkshavn', 'America/Dawson', 'America/Dawson_Creek', 'America/Denver', 'America/Detroit', 'America/Dominica', 'America/
Edmonton', 'America/Eirunepe', 'America/El_Salvador', 'America/Ensenada', 'America/Fort_Nelson', 'America/Fort_Wayne', 'America/Fortaleza', 'America/Glace_Bay', '
America/Godthab', 'America/Goose_Bay', 'America/Grand_Turk', 'America/Grenada', 'America/Guadeloupe', 'America/Guatemala', 'America/Guayaquil', 'America/Guyana',
'America/Halifax', 'America/Havana', 'America/Hermosillo', 'America/Indiana/Indianapolis', 'America/Indiana/Knox', 'America/Indiana/Marengo', 'America/Indiana/Pet
ersburg', 'America/Indiana/Tell_City', 'America/Indiana/Vevay', 'America/Indiana/Vincennes', 'America/Indiana/Winamac', 'America/Indianapolis', 'America/Inuvik',
'America/Iqaluit', 'America/Jamaica', 'America/Jujuy', 'America/Juneau', 'America/Kentucky/Louisville', 'America/Kentucky/Monticello', 'America/Knox_IN', 'America
```

Task4: To represent dictionary of country code as supported time zone as value.

Python Code:

```
import pytz
print('country_timezones =')
for key, val in pytz.country_timezones.items():
    print(key, '=', val, end=',')
print('\n')
print('Time-zones supported by Antarctica =', pytz.country_timezones['AQ'])
```

Output:

```
Time-zones supported by Antarctica = ['Antarctica/McMurdo', 'Antarctica/Casey', 'Antarctica/Davis', 'Antarctica/DumontDUrville', 'Antarctica/Mawson', 'Antarctica/
Palmer', 'Antarctica/Rothera', 'Antarctica/Syowa', 'Antarctica/Troll', 'Antarctica/Vostok']
```

Task5: String Concatenation using “,” in python

Python Code:

```
var1 = "Aryan"
var2 = "Dilipbhai"
var3 = "Langhanoja"
print(var1, var2, var3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Aryan Dilipbhai Langhanoja
```

Lab 5

Name: Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditionals, loops and functions

AIM:- Write a python program to create, append and remove lists in python

Introduction:- Lists are used to store multiple items in a single variable. Lists are one of 4 built-in data types in Python used to store collections of data, the other 3 are Tuple, Set, and Dictionary, all with different qualities and usage. Lists are created using square brackets

Task1: Write a program to create list in python

Python Code:

```
thislist = ["Aryan", "Dilipbhai", "Langhanoja"]
print(thislist)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
['Aryan', 'Dilipbhai', 'Langhanoja']
```

Task2: Write a python program th remove the word from a list using remove

Python Code:

```
thislist = ["Aryan", "Dilipbhai", "Langhanoja"]
thislist.remove("Dilipbhai")
print(thislist)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
['Aryan', 'Langhanoja']
```

Task3: Write a python program to add in the list using append

Python Code:

```
fruits = ["Aryan", "Langhanoja", "From"]
fruits.append("Ahmedabad")
print(fruits)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
['Aryan', 'Langhanoja', 'From', 'Ahmedabad']
```

Task4: Write a python program to find the length of a list

Python Code:

```
thislist = ["Aryan", "Dilipbhai", "Langhanoja"]
print(len(thislist))
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
3
```

Task5: Write a python program using the pop method to remove words by indexing

Python Code:

```
thislist = ["Aryan", "Dilipbhai", "Langhanoja"]
thislist.pop()
print(thislist)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
['Aryan', 'Dilipbhai']
```

Lab 6

Name: - Aryan Dilipbhai Langhanoja

Date:-

Enrollment No:- 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Develop programs to understand the control structure in python

Introduction:- A code is written to allow making choices and several pathways through the program to be followed depending on shifts in variable values. All programming languages contain a pre-included set of control structures that enable these control flows to execute, which makes it conceivable.

Task1: Python program to show how a sequential control structure

Python Code:

```
a = 25
b = 20
c = a - b
d = a + b
e = a * b
print("Subtraction is: ", c)
print("Addition is: ", d)
print("Multiplication is: ", e)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Subtraction is:  5
Addition is:  45
Multiplication is:  500
```

Task2: Write a python code for loop control statement

Python Code:

```
for letter in 'Aryan Langhanoja':
   if letter == 'r' or letter == 'n':
      continue
  print('Current Letter :', letter)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Current Letter : A
Current Letter : y
Current Letter : a
Current Letter :
Current Letter : L
Current Letter : a
Current Letter : g
Current Letter : h
Current Letter : a
Current Letter : o
Current Letter : j
Current Letter : a
```

Task3: Write a python code to bring control out of the loop

Python Code:

```
for letter in 'Aryan':
    if letter == 'r' or letter == 'a':
        break
print('Current Letter :', letter)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
Current Letter : r
```

Task4: Write a python code using for loop

Python Code:

```
l = [1, 2, 3, 4, 5, 6]
for i in range(len(l)):
    print(l[i], end = ", ")
print("\n")
for j in range(0,10):
    print(j, end = ", ")
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester - 3
1, 2, 3, 4, 5, 6,

0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
```

Task5: Write a python code using while loop

Python Code:

```
b = 9
a = 1
while a < b:
    print(a, end = " ")
    a = a + 1
print("completed")
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
1 2 3 4 5 6 7 8 completed
```

Lab 7

Name:- Aryan Dilipbhai Langhanoja

Date:-

Enrollment No:- 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Develop programs to understand the control structure in python

Introduction:- Tuples are used to store multiple items in a single variable. Tuple is one of 4 built-in data types in Python used to store collections of data, the other 3 are List, Set, and Dictionary, all with different qualities and usage. A tuple is a collection which is ordered and unchangeable. Tuples are written with round brackets.

Task1: Write a python program to create a simple tuple

Python Code:

```
thistuple = ("Aryan", "Dilipbhai", "Langhanoja")
print(thistuple)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
('Aryan', 'Dilipbhai', 'Langhanoja')
```

Task2: Write a python program to allow the tuple duplicates

Python Code:

```
thistuple = ("Aryan", "Dilipbhai", "Langhanoja", "From", "Ahmedabad")
print(thistuple)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
('Aryan', 'Dilipbhai', 'Langhanoja', 'From', 'Ahmedabad')
```

Task3 : Write a python program to show tuple items and data types

Python Code:

```
tuple1 = ("Aryan", "Dilipbhai", "Langhanoja")
tuple2 = (1, 5, 7, 9, 3)
```

```
tuple3 = (True, False, False)
print(tuple1)
print(tuple2)
print(tuple3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
('Aryan', 'Dilipbhai', 'Langhanoja')
(1, 5, 7, 9, 3)
(True, False, False)
```

Task4: Write a python program to join two transport using operator

Python Code:

```
tuple1 = ("A", "B" , "C")
tuple2 = (1, 2, 3)
tuple3 = tuple1 + tuple2
print(tuple3)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
('A', 'B', 'C', 1, 2, 3)
```

Task5: Write a python program to multiply two numbers using operator

Python Code:

```
fruits = ("Aryan", "Dilipbhai", "Langhanoja")
mytuple = fruits * 2
print(mytuple)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester -
('Aryan', 'Dilipbhai', 'Langhanoja', 'Aryan', 'Dilipbhai', 'Langhanoja')
```

Lab 8

Name:- Aryan Dilipbhai Langhanoja

Date:-

Enrollment No:- 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Write a program to demonstrate working with dictionaries in python

Introduction:- Dictionaries are used to store data values in key:value pairs. A dictionary is a collection which is ordered*, changeable and do not allow duplicates.

Task1: Write a python program to create a simple tuple

Python Code:

```
thisdict =  {
  "University": "Marwadi University",
  "College": "Faculty of Technology",
  "Year": 2023
}
print(thisdict)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester - 3/Programming With Python/Lab
{'University': 'Marwadi University', 'College': 'Faculty of Technology', 'Year': 2023}
```

Task2: Write a python program for duplicates not allowed

Python Code:

```
thisdict =  {
  "University": "Marwadi University",
  "College": "Faculty of Technology",
  "Year": 2022 ,
  "Year-2" : 2026
}
print(len(thisdict))
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
4
```

Task3: Write a python program to dict() constructor to make a dictionary

Python Code:

```
thisdict = dict(name = "Aryan Langhanoja", year = 2023, country = "India")
print(thisdict)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
{'name': 'Aryan Langhanoja', 'year': 2023, 'country': 'India'}
```

Task4: Write a python program to add elements in the dictionary

Python Code:

```
thisdict =  {
  "College": "Marwadi University",
  "Aame": "Aryan Langhanoja",
  "year": 2023
}
thisdict["Department"] = "ICT"
print(thisdict)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester - 3/Programming With
{'College': 'Marwadi University', 'Aame': 'Aryan Langhanoja', 'year': 2023, 'Department': 'ICT'}
```

Task5: Write a python program to remove items from a any element

Python Code:

```
thisdict =  {
  "College": "Marwadi University",
  "Name": "Aryan Langhanoja",
  "year": 2023
}
thisdict.pop("year")
print(thisdict)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
{'College': 'Marwadi University', 'Name': 'Aryan Langhanoja'}
```

Lab 9

Name:- Aryan Dilipbhai Langhanoja

Date:-

Enrollment No:- 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Write a python program to convert temperature to and from Celsius to Fahrenheit.

Introduction:- The centigrade scale, which is also called the Celsius scale, was developed by Swedish astronomer Andres Celsius. In the centigrade scale, water freezes at 0 degrees and boils at 100 degrees.

Task1: Write a python program to convert temperature into different measures.

Python Code:

```
celsius = 45
fahrenheit = (celsius * 1.8) + 32
print('%.2f Celsius is Equivalent to: %.2f Fahrenheit'
    % (celsius, fahrenheit))
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
45.00 Celsius is Equivalent to: 113.00 Fahrenheit
```

Task2: Write a python program to convert temperature into different measures.

Python Code:

```
fahrenheit = 273
celsius = (fahrenheit-32)/1.8
print('%.2f Fahrenheit is equivalent to: %.2f Celsius'
    % (fahrenheit ,celsius))
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester
273.00 Fahrenheit is equivalent to: 133.89 Celsius
```

Lab 10

Name:- Aryan Dilipbhai Langhanoja

Date:

Enrollment No:- 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Write a python program to define a module and import a specific function in that module to another program.

Introduction:- As our program grows bigger, it may contain many lines of code. Instead of putting everything in a single file, we can use modules to separate codes in separate files as per their functionality. This makes our code organized and easier to maintain. Module is a file that contains code to perform a specific task. A module may contain variables, functions, classes etc.

Task1: Write a python program to import math function

Python Code:

```
import math
print("The value of pi is", math.pi)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
The value of pi is 3.141592653589793
```

Task2: Write a python program to import function by renaming it .

Python Code:

```
import math as m
print(m.pi)
```

Output:

```
PS C:\Users\abc> & D:/DLLs/Anaconda/python.exe
3.141592653589793
```

Task3: Write a python program to convert temperature into different measures.

Python Code:

```
def greeting(name):
    print("Hello, " + name)

import mymodule
mymodule.greeting("Aryan Langhanoja")
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\
efull Data/Semester - 3/Programming With Py
Hello, Aryan Langhanoja
```

Task4: Write a python program to use variables in the module.

Python Code:

```
def person1(name):
    return "Hello " + name
import mymodule
a = mymodule.person1("Aryan")
print(a)
```

Output:

```
PS D:\Aryan Data\Usefull Data\S
efull Data/Semester - 3/Progra
Hello Aryan
```

Task5: Write a python program for built in function in the module

Python Code:

```
import platform
x = dir(platform)
print(x)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Usefull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
['_Processor', '_WIN32_CLIENT_RELEASES', '_WIN32_SERVER_RELEASES', '__builtins__', '__cached__', '__copyright__', '__doc__', '__file__', '__loader__', '__name__', '__package__', '__spec__', '__version__', '_comparable_version', '_component_re', '_default_architecture', '_follow_symlinks', '_get_machine_win32', '_ironpython26_sys_version_parser', '_ironpython_sys_version_parser', '_java_getprop', '_libc_search', '_mac_ver_xml', '_node', '_norm_version', '_os_release_cache', '_os_release_candidates', '_os_release_line', '_os_release_unescape', '_parse_os_release', '_platform', '_platform_cache', '_pypy_sys_version_parser', '_sys_version', '_sys_version_cache', '_sys_version_parser', '_syscmd_file', '_syscmd_ver', '_uname_cache', '_unknown_as_blank', '_ver_output', '_ver_stages', 'architecture', 'collections', 'freedesktop_os_release', 'functools', 'itertools', 'java_ver', 'libc_ver', 'mac_ver', 'machine', 'node', 'os', 'platform', 'processor', 'python_branch', 'python_build', 'python_compiler', 'python_implementation', 'python_revision', 'python_version', 'python_version_tuple', 're', 'release', 'sys', 'system', 'system_alias', 'uname', 'uname_result', 'version', 'win32_edition', 'win32_is_iot', 'win32_ver']
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
```

Lab 11

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Write a python program to call data member and function using classes and object.

Introduction:- A class is a user-defined blueprint or prototype from which objects are created. Classes provide a means of bundling data and functionality together. Creating a new class creates a new type of object, allowing new instances of that type to be made. Each class instance can have attributes attached to it for maintaining its state. Class instances can also have methods (defined by their class) for modifying their state.

Task1: Write a python program to create object

Python Code:

```
class name:
    attr1 = "marwadi"
    attr2 = "university"
    def fun(self):
        print("I'm study in", self.attr1)
        print("I'm study in", self.attr2)
Rodger = name()
print(Rodger.attr1)
Rodger.fun()
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
marwadi
I'm study in marwadi
I'm study in university
```

Task2: Write a python program using int() method

Python Code:

```
class Person:
    def __init__(self, name):
        self.name = name
    def say_hi(self):
        print('Hello, My Name Is', self.name)
p = Person('Aryan')
p.say_hi()
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Hello, My Name Is Aryan
```

Task3: Write a python program using class with input.

Python Code:

```
class Student:
    'A student class'
    stuCount = 0
    def __init__(self):
        self.name = input('Enter student name:')
        self.rollno = input('Enter student rollno:')
        Student.stuCount += 1
    def displayStudent(self):
        print("Name:", self.name, "Rollno:", self.rollno)
stu1 = Student()
stu2 = Student()
stu3 = Student()
stu1.displayStudent()
stu2.displayStudent()
stu3.displayStudent()
print('total no. of students:', Student.stuCount)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Enter student name:Aryan
Enter student rollno:30
Enter student name:Krish
Enter student rollno:22
Enter student name:Jay
Enter student rollno:40
Name: Aryan Rollno: 30
Name: Krish Rollno: 22
Name: Jay Rollno: 40
total no. of students: 3
```

Task4: Write a python program using self parameter.

Python Code:

```
class harsh:
    def __init__(somename, name, company):
        somename.name = name
        somename.company = company
    def show(somename):
        print("Hello my name is " + somename.name +
              " and I work in "+somename.company+".")
obj = harsh("Aryan", "ICT Marwadi Univerisy")
obj.show()
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Hello my name is Aryan and I work in ICT Marwadi Univerisy.
```

Task5: Write a python program using instance variables using a constructor

Python Code:

```
class Dog:
    animal = 'dog'
    def __init__(self, breed):
        self.breed = breed
    def setColor(self, color):
        self.color = color
    def getColor(self):
        return self.color
Rodger = Dog("Pug")
Rodger.setColor("Brown")
print(Rodger.getColor())
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Brown
```

Lab 12

Name: Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Practical based on Numpy ndarray.

Introduction:- Array in Numpy is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers. In Numpy, number of dimensions of the array is called rank of the array.A tuple of integers giving the size of the array along each dimension is known as shape of the array. An array class in Numpy is called as ndarray. Elements in Numpy arrays are accessed by using square brackets and can be initialized by using nested Python Lists.

Task1: Write a python program to demonstrate basic array

Python Code:

```
import numpy as np
arr = np.array([[1, 2, 3],
                [4, 2, 5]])
print("Array is of type: ", type(arr))
print("No. of dimensions: ", arr.ndim)
print("Shape of array: ", arr.shape)
print("Size of array: ", arr.size)
print("Array stores elements of type: ", arr.dtype)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Array is of type:  <class 'numpy.ndarray'>
No. of dimensions:  2
Shape of array:  (2, 3)
Size of array:  6
Array stores elements of type:  int32
```

Tas2: Write a python program for array indexing.

Python Code:

```
import numpy as np
arr = np.array([[-1, 2, 0, 4],
```

```
            [4, -0.5, 6, 0],
            [2.6, 0, 7, 8],
            [3, -7, 4, 2.0]])
temp = arr[:2, ::2]
print("Array with first 2 rows and alternate"
    "columns(0 and 2):\n", temp)
temp = arr[[0, 1, 2, 3], [3, 2, 1, 0]]
print("Elements at indices (0, 3), (1, 2), (2, 1),"
    "(3, 0):\n", temp)
cond = arr > 0
temp = arr[cond]
print("Elements greater than 0:\n", temp)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Array with first 2 rows and alternatecolumns(0 and 2):
 [[-1.  0.]
 [ 4.  6.]]
Elements at indices (0, 3), (1, 2), (2, 1),(3, 0):
 [4. 6. 0. 3.]
Elements greater than 0:
 [2.  4.  4.  6.  2.6 7.  8.  3.  4.  2. ]
```

Task3: Write a python program to convert temperature into different measures.

Python Code:

```
import numpy as np
a = np.array([1, 2, 5, 3])
print("Adding 1 to every element:", a+1)
print("Subtracting 3 from each element:", a-3)
print("Multiplying each element by 10:", a*10)
print("Squaring each element:", a**2)
a *= 2
print("Doubled each element of original array:", a)
a = np.array([[1, 2, 3], [3, 4, 5], [9, 6, 0]])
print("Original array:\n", a)
print("Transpose of array:\n", a.T)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Adding 1 to every element: [2 3 6 4]
Subtracting 3 from each element: [-2 -1  2  0]
Multiplying each element by 10: [10 20 50 30]
Squaring each element: [ 1  4 25  9]
Doubled each element of original array: [ 2  4 10  6]
Original array:
 [[1 2 3]
 [3 4 5]
 [9 6 0]]
Transpose of array:
 [[1 3 9]
 [2 4 6]
 [3 5 0]]
```

Task4: Write a python program for unary operator.

Python Code:

```
import numpy as np
arr = np.array([[1, 5, 6],
                [4, 7, 2],
                [3, 1, 9]])
print("Largest element is:", arr.max())
print("Row-wise maximum elements:",
      arr.max(axis=1))
print("Column-wise minimum elements:",
      arr.min(axis=0))
print("Sum of all array elements:",
      arr.sum())
print("Cumulative sum along each row:\n",
      arr.cumsum(axis=1))
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Largest element is: 9
Row-wise maximum elements: [6 7 9]
Column-wise minimum elements: [1 1 2]
Sum of all array elements: 38
Cumulative sum along each row:
 [[ 1  6 12]
 [ 4 11 13]
 [ 3  4 13]]
```

Task5: Write a python program for binary operator.

Python Code:

```
import numpy as np
a = np.array([[1, 2],
              [3, 4]])
b = np.array([[4, 3],
              [2, 1]])
print("Array sum:\n", a + b)
print("Array multiplication:\n", a*b)
print("Matrix multiplication:\n", a.dot(b))
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Array sum:
 [[5 5]
 [5 5]]
Array multiplication:
 [[4 6]
 [6 4]]
Matrix multiplication:
 [[ 8  5]
 [20 13]]
```

Lab 13

Name: Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Practical based on Pandas Data Structure

Introduction:- Pandas is an open-source library that uses for working with relational or labeled data both easily and intuitively. It provides various data structures and operations for manipulating numerical data and time series. It offers a tool for cleaning and processes your data. It is the most popular Python library that is used for data analysis.

Task1: Write a python program for series holding the char data type

Python Code:

```
import pandas as pd
list = ['A', 'r', 'y', 'a', 'n']
res = pd.Series(list)
print(res)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
0    A
1    r
2    y
3    a
4    n
dtype: object
```

Tas2: Write a python program for series holding the int data type.

Python Code:

```
import pandas as pd
list = [1, 2, 3, 4, 5]
res = pd.Series(list)
print(res)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
0    1
1    2
2    3
3    4
4    5
dtype: int64
```

Task3: Write a python program for holding a dictionary.

Python Code:

```
import pandas as pd
dic = {'Id': 1013, 'Name': 'Aryan',
       'State': 'Maniput', 'Age': 24}
res = pd.Series(dic)
print(res)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
Id         1013
Name      Aryan
State    ujarat
Age          18
dtype: object
```

Task4: Write a python program to create dataframe.

Python Code:

```
import pandas as pd
df = pd.DataFrame({'X': [78, 85, 96, 80, 86], 'Y': [
            84, 94, 89, 83, 86], 'Z': [86, 97, 96, 72, 83]})
print(df)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work>
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
    X   Y   Z
0  78  84  86
1  85  94  97
2  96  89  96
3  80  83  72
4  86  86  83
```

Task5: Write a python program using key / value object as series

Python Code:

```
import platform
x = dir(platform)
print(x)
```

Output:

```
PS D:\Aryan Data\Usefull Data\Semester - 3\Programming With Python\Lab Work> & D:/DLLs/Anaconda/python.exe "d:/Aryan Data/Us
efull Data/Semester - 3/Programming With Python/Lab Work/Lab_Manual.py"
['_Processor', '_WIN32_CLIENT_RELEASES', '_WIN32_SERVER_RELEASES', '__builtins__', '__cached__', '__copyright__', '__doc__',
 '__file__', '__loader__', '__name__', '__package__', '__spec__', '__version__', '_comparable_version', '_component_re', '_d
efault_architecture', '_follow_symlinks', '_get_machine_win32', '_ironpython26_sys_version_parser', '_ironpython_sys_version
_parser', '_java_getprop', '_libc_search', '_mac_ver_xml', '_node', '_norm_version', '_os_release_cache', '_os_release_candi
dates', '_os_release_line', '_os_release_unescape', '_parse_os_release', '_platform', '_platform_cache', '_pypy_sys_version_
parser', '_sys_version', '_sys_version_cache', '_sys_version_parser', '_syscmd_file', '_syscmd_ver', '_uname_cache', '_unkno
wn_as_blank', '_ver_output', '_ver_stages', 'architecture', 'collections', 'freedesktop_os_release', 'functools', 'itertools
', 'java_ver', 'libc_ver', 'mac_ver', 'machine', 'node', 'os', 'platform', 'processor', 'python_branch', 'python_build', 'py
thon_compiler', 'python_implementation', 'python_revision', 'python_version', 'python_version_tuple', 're', 'release', 'sys'
, 'system', 'system_alias', 'uname', 'uname_result', 'version', 'win32_edition', 'win32_is_iot', 'win32_ver']
```

Lab 14

Name: Aryan Diliphai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Simulate continues time elementary signals

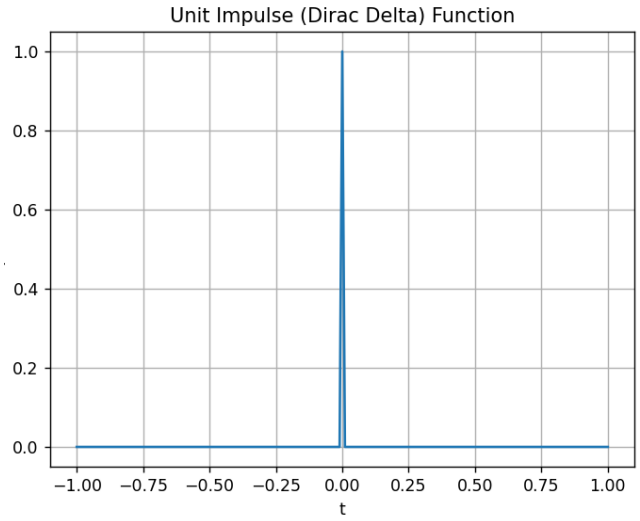
Introduction:- To simulate continuous-time elementary signals in Python, you can use libraries like NumPy and Matplotlib for generating and plotting the signals.

Task1: Write a python program to represent unit impulse function.

Python Code:

```
import numpy as np
import matplotlib.pyplot as plt
t = np.linspace(-1, 1, 201)
impulse = np.zeros_like(t)
impulse[t == 0] = 1
plt.plot(t, impulse)
plt.title('Unit Impulse (Dirac Delta) Function')
plt.xlabel('t')
plt.ylabel('Amplitude')
plt.grid(True)
plt.show()
```

Output:

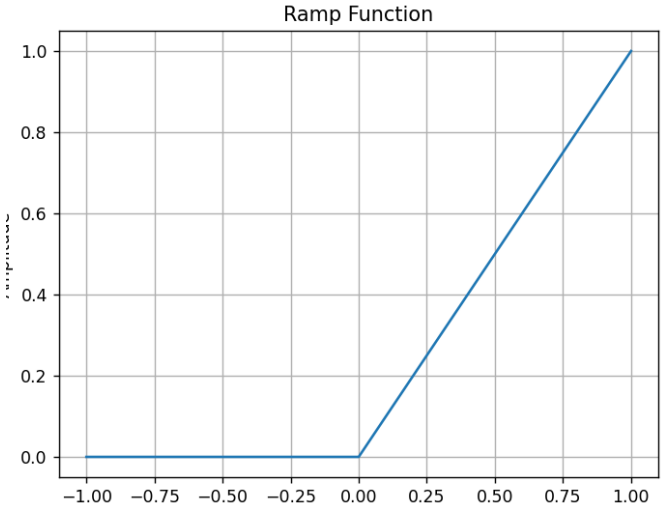


Tas2: Write a python program to represent ramp function.

Python Code:

```
ramp = np.maximum(0, t)
plt.plot(t, ramp)
plt.title('Ramp Function')
plt.xlabel('t')
plt.ylabel('Amplitude')
plt.grid(True)
plt.show()
```

Output:

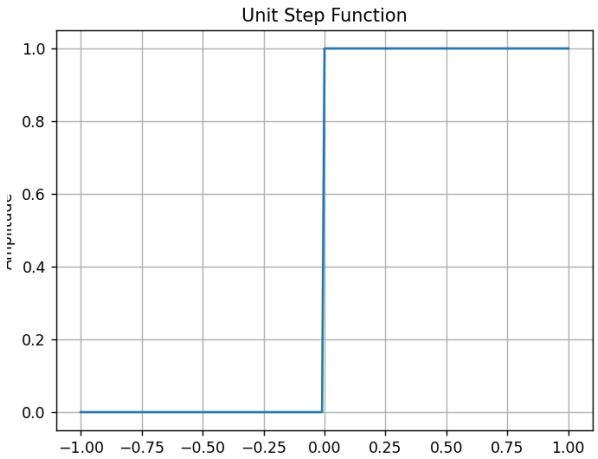


Task3: Write a python program to represent unit step function.

Python Code:

```
unit_step = np.heaviside(t, 1)
plt.plot(t, unit_step)
plt.title('Unit Step Function')
plt.xlabel('t')
plt.ylabel('Amplitude')
plt.grid(True)
plt.show()
```

Output:

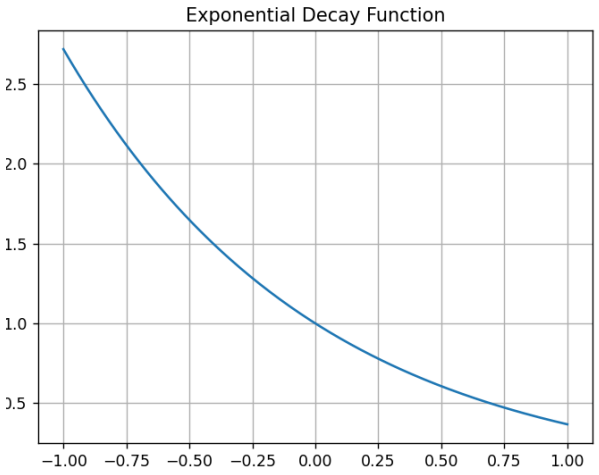


Task4: Write a python program to represent exponential decay function.

Python Code:

```
decay = np.exp(-t)
plt.plot(t, decay)
plt.title('Exponential Decay Function')
plt.xlabel('t')
plt.ylabel('Amplitude')
plt.grid(True)
plt.show()
```

Output:

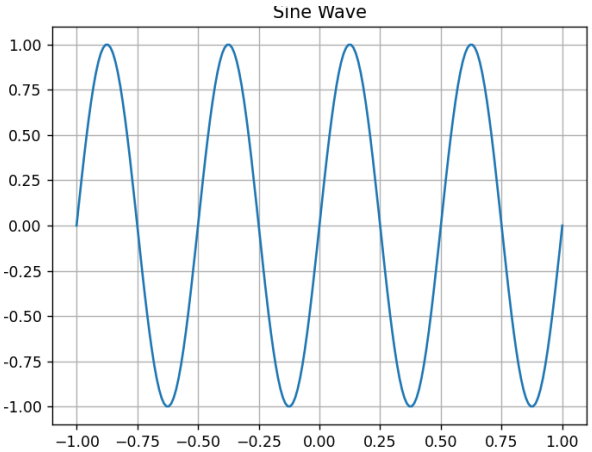


Task5: Write a python program to represent sine wave function.

Python Code:

```
frequency = 2
sine_wave = np.sin(2 * np.pi * frequency * t)
plt.plot(t, sine_wave)
plt.title('Sine Wave')
plt.xlabel('t')
plt.ylabel('Amplitude')
plt.grid(True)
plt.show()
```

Output:



Lab 15

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Simulate basic operations on continuous time elementary signals

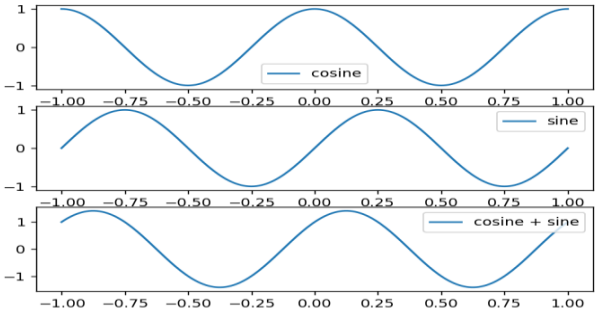
Introduction:- Continuous-time elementary signals simulated in Python. We'll cover addition, scaling, time shifting, differentiation, and integration using some common elementary signals. We'll use the numpy library for numerical calculations and matplotlib for plotting.

Task1: Write a python program to add of two signals(cosine+sine)

Python Code:

```
signal1 = np.cos(2 * np.pi * t)
signal2 = np.sin(2 * np.pi * t)
result_signal = signal1 + signal2
plt.figure()
plt.subplot(3, 1, 1)
plt.plot(t, signal1, label='cosine')
plt.legend()
plt.subplot(3, 1, 2)
plt.plot(t, signal2, label='sine')
plt.legend()
plt.subplot(3, 1, 3)
plt.plot(t, result_signal, label='cosine + sine')
plt.legend()
plt.show()
```

Output:

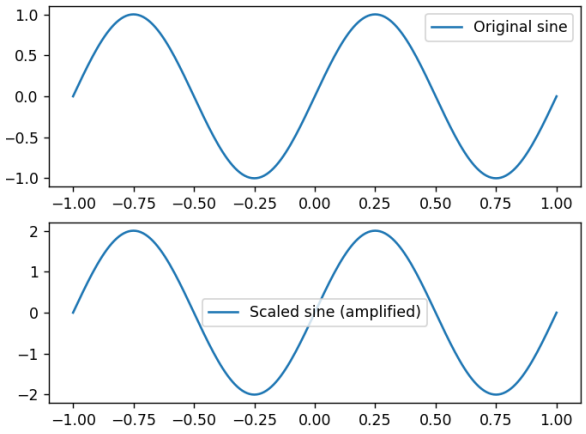


Tas2: Write a python program for scaling a signal.

Python Code:

```
original_signal = np.sin(2 * np.pi * t)
scaled_signal = 2 * original_signal
plt.figure()
plt.subplot(2, 1, 1)
plt.plot(t, original_signal, label='Original sine')
plt.legend()
plt.subplot(2, 1, 2)
plt.plot(t, scaled_signal, label='Scaled sine (amplified)')
plt.legend()
plt.show()
```

Output:

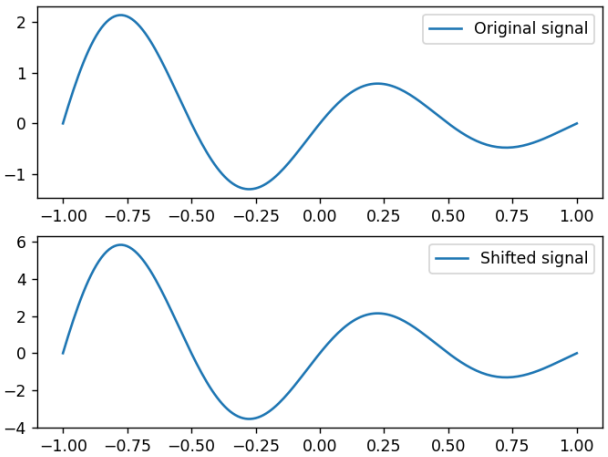


Task3: Write a python program for time shifting.

Python Code:

```
shifted_signal = np.exp(-(t - 1)) * np.sin(2 * np.pi * (t - 1))
plt.figure()
plt.subplot(2, 1, 1)
plt.plot(t, original_signal, label='Original signal')
plt.legend()
plt.subplot(2, 1, 2)
plt.plot(t, shifted_signal, label='Shifted signal')
plt.legend()
plt.show()
```

Output:

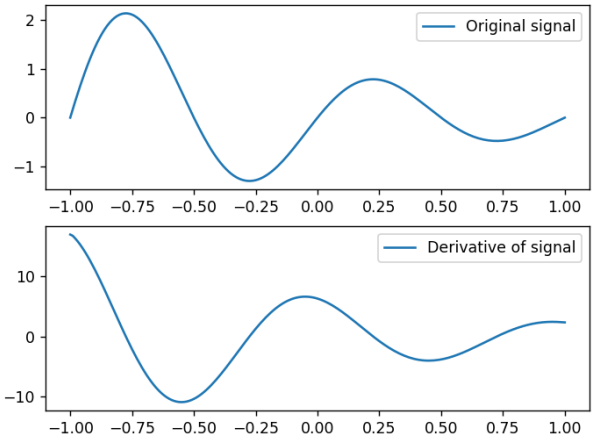


Task4: Write a python program for differentiation of a signal.

Python Code:

```
original_signal = np.exp(-t) * np.sin(2 * np.pi * t)
derivative_signal = np.gradient(original_signal, t)
plt.figure()
plt.subplot(2, 1, 1)
plt.plot(t, original_signal, label='Original signal')
plt.legend()
plt.subplot(2, 1, 2)
plt.plot(t, derivative_signal, label='Derivative of signal')
plt.legend()
plt.show()
```

Output:

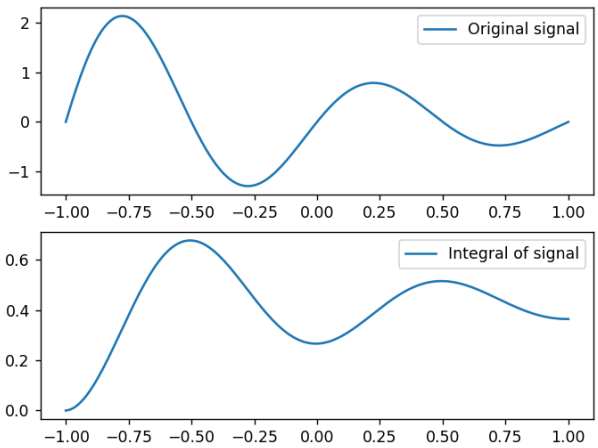


Task5: Write a python program for integration of a signal

Python Code:

```
original_signal = np.exp(-t) * np.sin(2 * np.pi * t)
integral_signal = np.cumsum(original_signal) * (t[1] - t[0])
plt.figure()
plt.subplot(2, 1, 1)
plt.plot(t, original_signal, label='Original signal')
plt.legend()
plt.subplot(2, 1, 2)
plt.plot(t, integral_signal, label='Integral of signal')
plt.legend()
plt.show()
```

Output:



Lab 16

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Simulate sampling on contiguous time signals and generate a frequency spectrum of signals before and after sampling

Introduction:- To simulate sampling of continuous-time signals and generate their frequency spectra before and after sampling, you can use Python with libraries such as numpy and matplotlib.

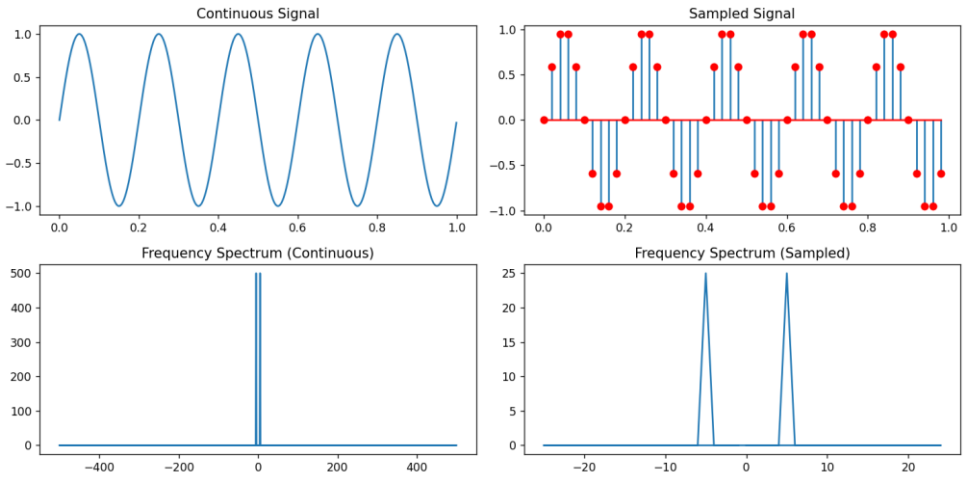
Task1: Write a python program for continuous time sinusoidal signal.

Python Code:

```
import numpy as np
import matplotlib.pyplot as plt
Fs = 1000
T = 1.0
t_continuous = np.linspace(0, T, int(T * Fs), endpoint=False)
freq1 = 5
signal_continuous1 = np.sin(2 * np.pi * freq1 * t_continuous)
Fs_new1 = 50  # New sampling frequency (Hz)
t_sampled1 = np.arange(0, T, 1 / Fs_new1)
signal_sampled1 = np.sin(2 * np.pi * freq1 * t_sampled1)
# Compute frequency spectra
fft_continuous1 = np.fft.fft(signal_continuous1)
fft_sampled1 = np.fft.fft(signal_sampled1)
# Frequency axis for plotting
frequencies1 = np.fft.fftfreq(len(t_continuous), 1 / Fs)
frequencies_sampled1 = np.fft.fftfreq(len(t_sampled1), 1 / Fs_new1)
plt.figure(figsize=(12, 6))
plt.subplot(2, 2, 1)
plt.plot(t_continuous, signal_continuous1)
plt.title('Continuous Signal')
plt.subplot(2, 2, 2)
plt.stem(t_sampled1, signal_sampled1, markerfmt='ro', basefmt='r-')
plt.title('Sampled Signal')
plt.subplot(2, 2, 3)
```

```
plt.plot(frequencies1, abs(fft_continuous1))
plt.title('Frequency Spectrum (Continuous)')
plt.subplot(2, 2, 4)
plt.plot(frequencies_sampled1, abs(fft_sampled1))
plt.title('Frequency Spectrum (Sampled)')
plt.tight_layout()
plt.show()
```

Output:



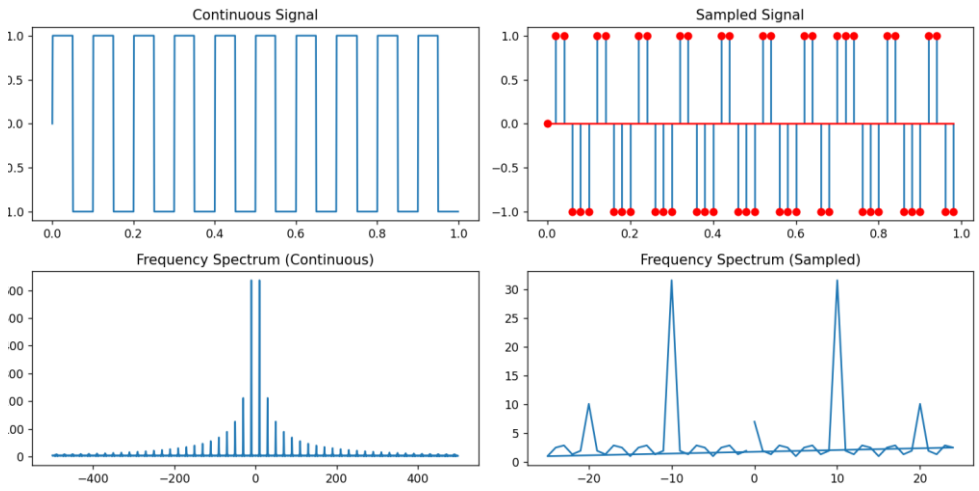
Tas2: Write a python program for continues time square wave signal.

Python Code:

```
import numpy as np
import matplotlib.pyplot as plt
# Define time parameters
Fs = 1000  # Sampling frequency (Hz)
T = 1.0    # Total time (seconds)
t_continuous = np.linspace(0, T, int(T * Fs), endpoint=False)
#Continuous-time square wave signal
freq2 = 10  # Frequency of the square wave (Hz)
signal_continuous2 = np.sign(np.sin(2 * np.pi * freq2 * t_continuous))
# Sample the signal
Fs_new2 = 50  # New sampling frequency (Hz)
t_sampled2 = np.arange(0, T, 1 / Fs_new2)
signal_sampled2 = np.sign(np.sin(2 * np.pi * freq2 * t_sampled2))
# Compute frequency spectra
fft_continuous2 = np.fft.fft(signal_continuous2)
fft_sampled2 = np.fft.fft(signal_sampled2)
```

```
# Frequency axis for plotting
frequencies2 = np.fft.fftfreq(len(t_continuous), 1 / Fs)
frequencies_sampled2 = np.fft.fftfreq(len(t_sampled2), 1 / Fs_new2)
# Plot the results
plt.figure(figsize=(12, 6))
plt.subplot(2, 2, 1)
plt.plot(t_continuous, signal_continuous2)
plt.title('Continuous Signal')
plt.subplot(2, 2, 2)
plt.stem(t_sampled2, signal_sampled2, markerfmt='ro', basefmt='r-')
plt.title('Sampled Signal')
plt.subplot(2, 2, 3)
plt.plot(frequencies2, abs(fft_continuous2))
plt.title('Frequency Spectrum (Continuous)')
plt.subplot(2, 2, 4)
plt.plot(frequencies_sampled2, abs(fft_sampled2))
plt.title('Frequency Spectrum (Sampled)')
plt.tight_layout()
plt.show()
```

Output:



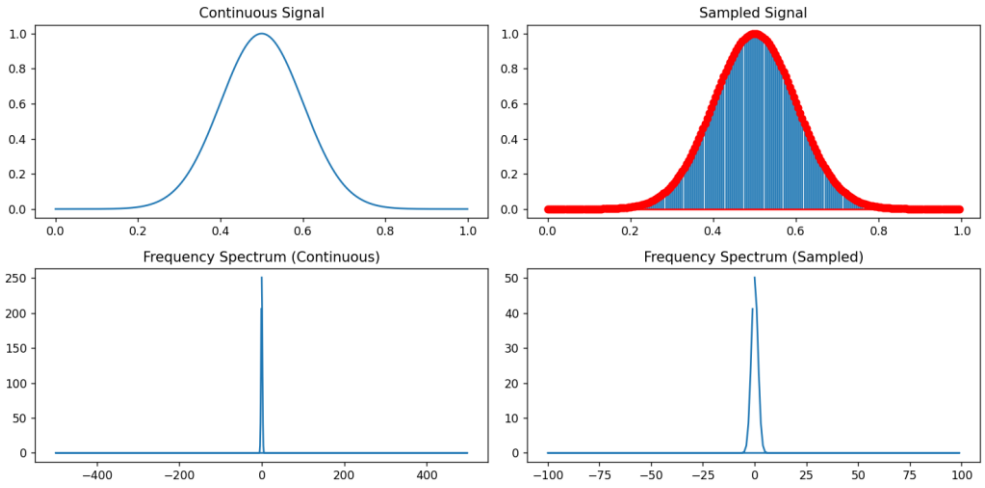
Task3: Write a python program for continues time gaussain signal .

Python Code:

```
import numpy as np
import matplotlib.pyplot as plt
# Define time parameters
Fs = 1000  # Sampling frequency (Hz)
T = 1.0    # Total time (seconds)
t_continuous = np.linspace(0, T, int(T * Fs), endpoint=False)
```

```
#  Continuous-time Gaussian signal
mu = 0.5  # Mean of the Gaussian
sigma = 0.1  # Standard deviation of the Gaussian
signal_continuous3 = np.exp(-(t_continuous - mu)**2 / (2 * sigma**2))
# Sample the signal
Fs_new3 = 200  # New sampling frequency (Hz)
t_sampled3 = np.arange(0, T, 1 / Fs_new3)
signal_sampled3 = np.exp(-(t_sampled3 - mu)**2 / (2 * sigma**2))
# Compute frequency spectra
fft_continuous3 = np.fft.fft(signal_continuous3)
fft_sampled3 = np.fft.fft(signal_sampled3)
# Frequency axis for plotting
frequencies3 = np.fft.fftfreq(len(t_continuous), 1 / Fs)
frequencies_sampled3 = np.fft.fftfreq(len(t_sampled3), 1 / Fs_new3)
# Plot the results
plt.figure(figsize=(12, 6))
plt.subplot(2, 2, 1)
plt.plot(t_continuous, signal_continuous3)
plt.title('Continuous Signal')
plt.subplot(2, 2, 2)
plt.stem(t_sampled3, signal_sampled3, markerfmt='ro', basefmt='r-')
plt.title('Sampled Signal')
plt.subplot(2, 2, 3)
plt.plot(frequencies3, abs(fft_continuous3))
plt.title('Frequency Spectrum (Continuous)')
plt.subplot(2, 2, 4)
plt.plot(frequencies_sampled3, abs(fft_sampled3))
plt.title('Frequency Spectrum (Sampled)')
plt.tight_layout()
plt.show()
```

Output:

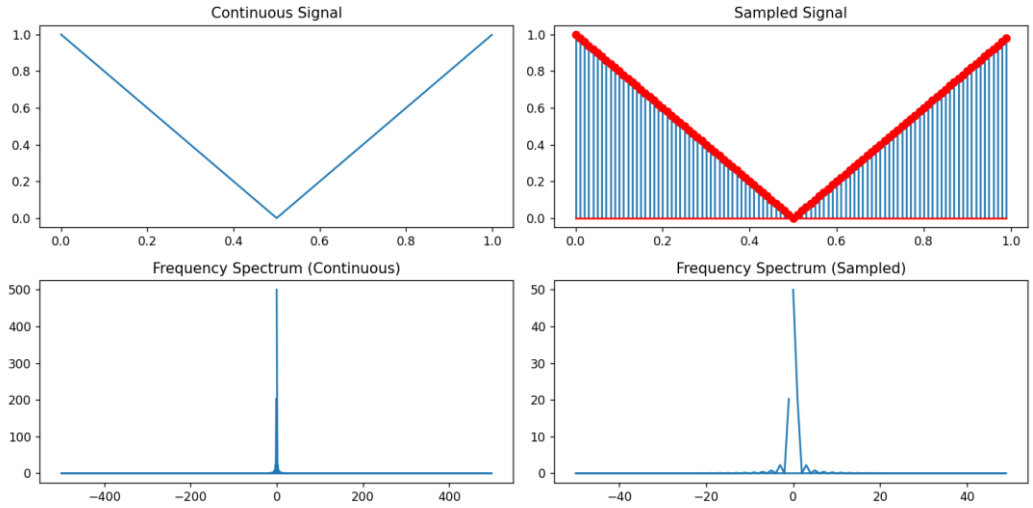


Task4: Write a python program for continuous time triangular signal .

Python Code:

```
import numpy as np
import matplotlib.pyplot as plt
# Define time parameters
Fs = 1000  # Sampling frequency (Hz)
T = 1.0    # Total time (seconds)
t_continuous = np.linspace(0, T, int(T * Fs), endpoint=False)

# Example 4: Continuous-time triangular signal
signal_continuous4 = np.abs(t_continuous - T / 2) / (T / 2)

# Sample the signal
Fs_new4 = 100  # New sampling frequency (Hz)
t_sampled4 = np.arange(0, T, 1 / Fs_new4)
signal_sampled4 = np.abs(t_sampled4 - T / 2) / (T / 2)

# Compute frequency spectra
fft_continuous4 = np.fft.fft(signal_continuous4)
fft_sampled4 = np.fft.fft(signal_sampled4)

# Frequency axis for plotting
frequencies4 = np.fft.fftfreq(len(t_continuous), 1 / Fs)
frequencies_sampled4 = np.fft.fftfreq(len(t_sampled4), 1 / Fs_new4)

# Plot the results
plt.figure(figsize=(12, 6))
plt.subplot(2, 2, 1)
plt.plot(t_continuous, signal_continuous4)
plt.title('Continuous Signal')
plt.subplot(2, 2, 2)
plt.stem(t_sampled4, signal_sampled4, markerfmt='ro', basefmt='r-')
plt.title('Sampled Signal')
plt.subplot(2, 2, 3)
plt.plot(frequencies4, abs(fft_continuous4))
plt.title('Frequency Spectrum (Continuous)')
plt.subplot(2, 2, 4)
plt.plot(frequencies_sampled4, abs(fft_sampled4))
plt.title('Frequency Spectrum (Sampled)')
plt.tight_layout()
plt.show()
```

Output:



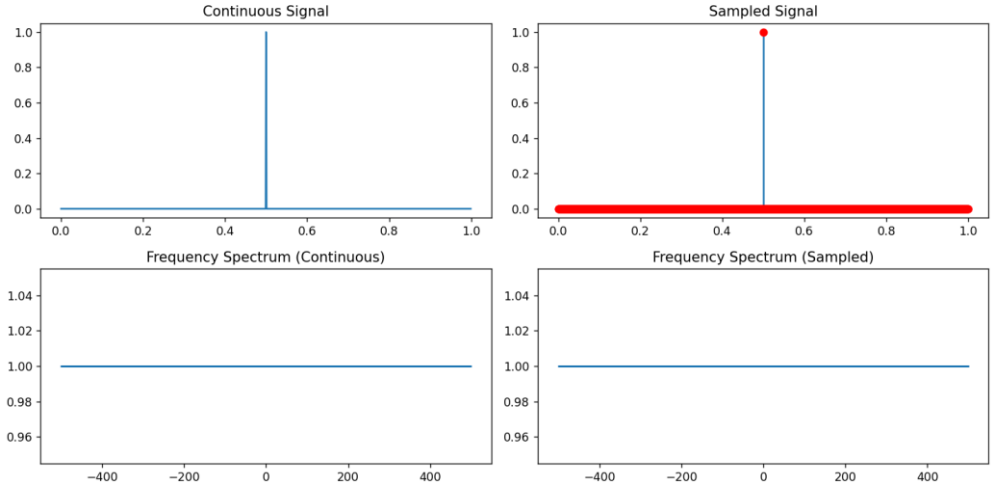
Task5: Write a python program for continuous time impulse signal.

Python Code:

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

# Define time parameters
Fs = 1000  # Sampling frequency (Hz)
T = 1.0    # Total time (seconds)
t_continuous = np.linspace(0, T, int(T * Fs), endpoint=False)
#Continuous-time impulse signal
signal_continuous5 = np.zeros_like(t_continuous)
signal_continuous5[len(t_continuous) // 2] = 1  # Impulse at the midpoint

# Sample the signal
Fs_new5 = 1000  # New sampling frequency (Hz)
t_sampled5 = np.arange(0, T, 1 / Fs_new5)
signal_sampled5 = np.zeros_like(t_sampled5)
signal_sampled5[len(t_sampled5) // 2] = 1  # Impulse at the midpoint

# Compute frequency spectra
fft_continuous5 = np.fft.fft(signal_continuous5)
fft_sampled5 = np.fft.fft(signal_sampled5)

# Frequency axis for plotting
frequencies5 = np.fft.fftfreq(len(t_continuous), 1 / Fs)
frequencies_sampled5 = np.fft.fftfreq(len(t_sampled5), 1 / Fs_new5)

# Plot the results
plt.figure(figsize=(12, 6))
```

```
plt.subplot(2, 2, 1)
plt.plot(t_continuous, signal_continuous5)
plt.title('Continuous Signal')
plt.subplot(2, 2, 2)
plt.stem(t_sampled5, signal_sampled5, markerfmt='ro', basefmt='r-')
plt.title('Sampled Signal')
plt.subplot(2, 2, 3)
plt.plot(frequencies5, abs(fft_continuous5))
plt.title('Frequency Spectrum (Continuous)')
plt.subplot(2, 2, 4)
plt.plot(frequencies_sampled5, abs(fft_sampled5))
plt.title('Frequency Spectrum (Sampled)')
plt.tight_layout()
plt.show()
```

Output:



Lab 17

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- To understand image processing in python with Matplotlib.

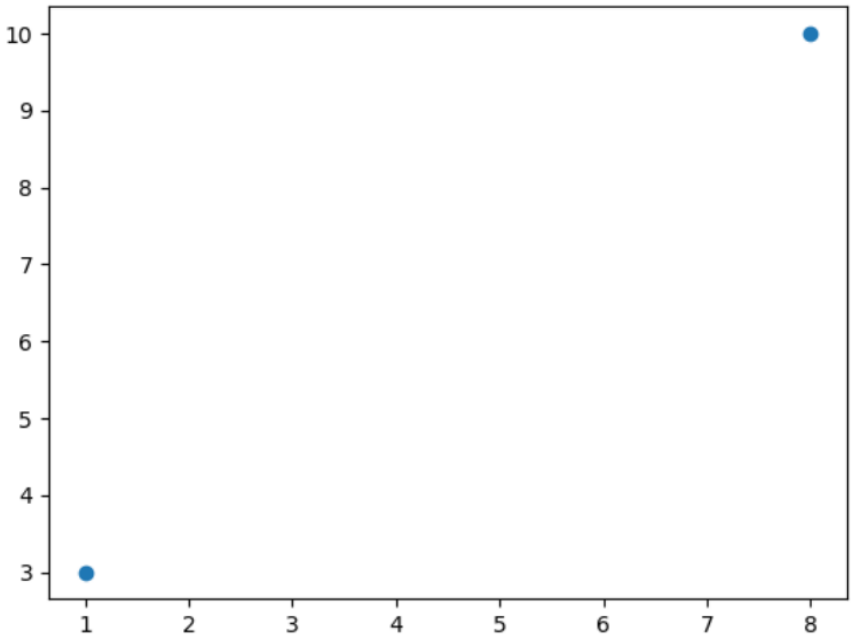
Introduction:- Matplotlib is a low level graph plotting library in python that serves as a visualization utility.Matplotlib was created by John D. Hunter. Matplotlib is open source and we can use it freely. Matplotlib is mostly written in python, a few segments are written in C, Objective-C and Javascript for Platform compatibility.

Task1: Write a python program to draw two points at different position.

Python Code:

```
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

xpoints = np.array([1, 8])
ypoints = np.array([3, 10])

plt.plot(xpoints, ypoints, 'o')
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

Output:

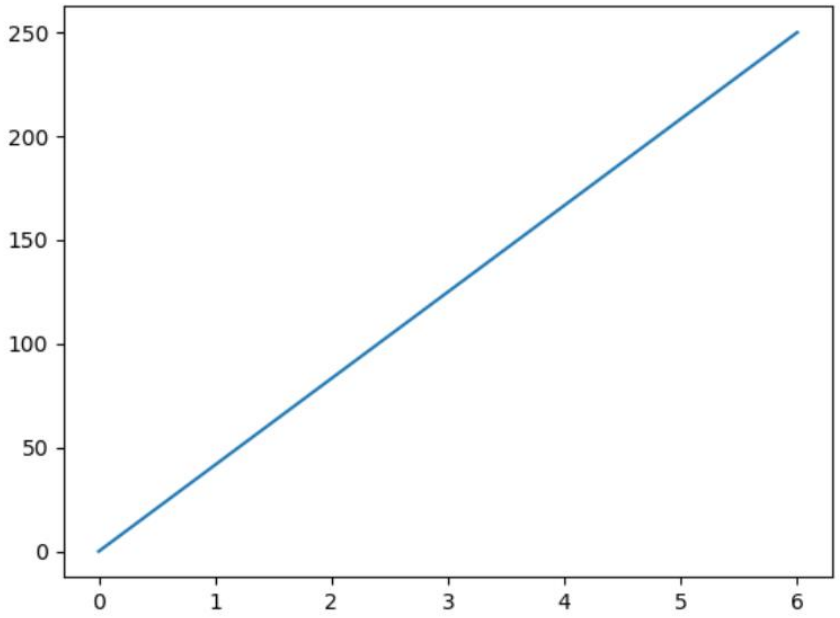


Tas2: Write a python program to draw a line diagram from position (0,0).

Python Code:

```
import sys
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
import numpy as np
xpoints = np.array([0, 6])
ypoints = np.array([0, 250])
plt.plot(xpoints, ypoints)
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

Output:

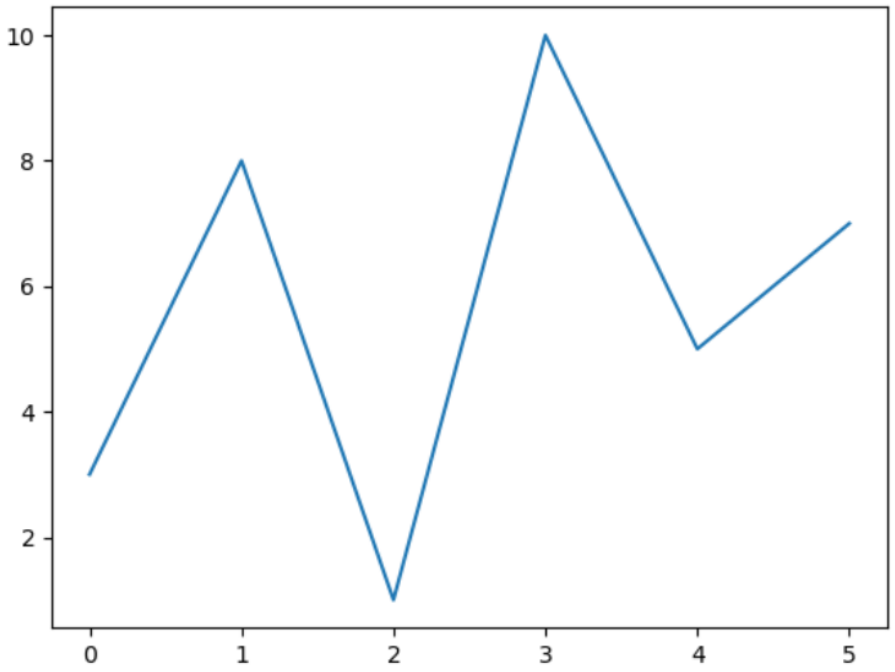


Task3: Write a python program for without using x-points, using defualt points.

Python Code:

```
import sys
import matplotlib
matplotlib.use('Agg')

import matplotlib.pyplot as plt
import numpy as np

ypoints = np.array([3, 8, 1, 10, 5, 7])

plt.plot(ypoints)
plt.show()

#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

Output:

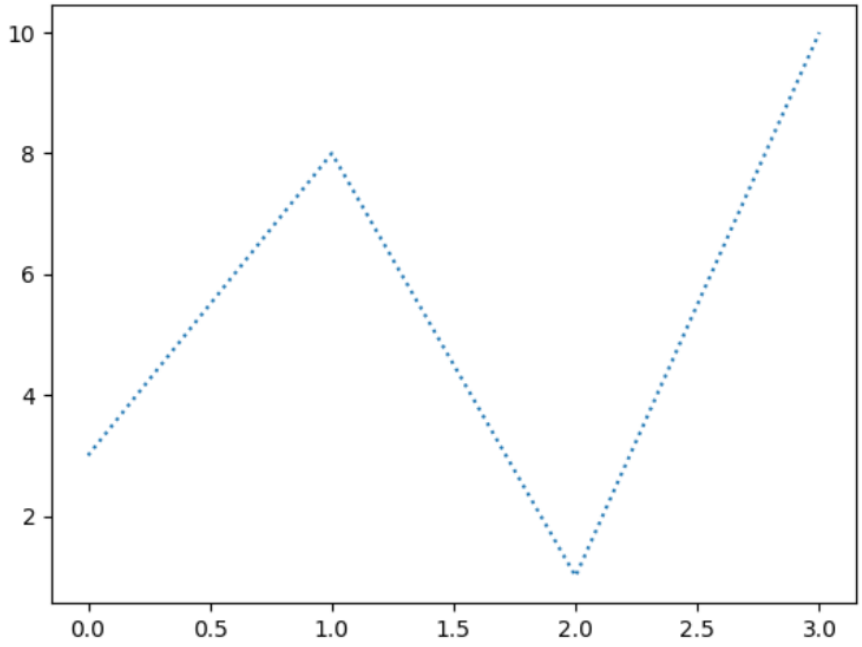


Task4: Write a python program to draw a different line style.

Python Code:

```
import sys
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
import numpy as np
ypoints = np.array([3, 8, 1, 10])
plt.plot(ypoints, linestyle = 'dotted')
plt.show()
#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

Output:

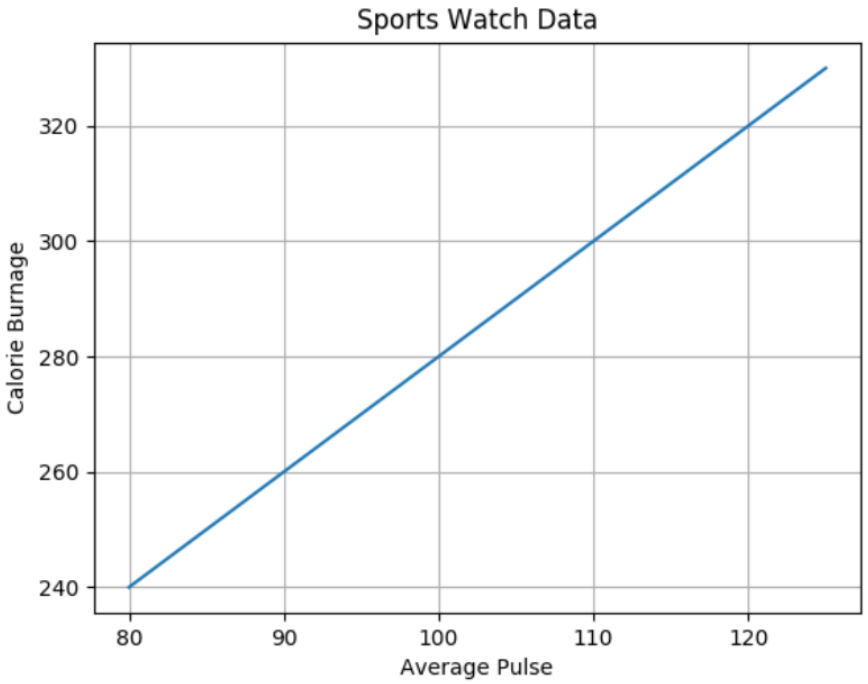


Task5: Write a python program with using the grid function.

Python Code:

```
import sys
import matplotlib
matplotlib.use('Agg')
import numpy as np
import matplotlib.pyplot as plt
x = np.array([80, 85, 90, 95, 100, 105, 110, 115, 120, 125])
y = np.array([240, 250, 260, 270, 280, 290, 300, 310, 320, 330])
plt.title("Sports Watch Data")
plt.xlabel("Average Pulse")
plt.ylabel("Calorie Burnage")
plt.plot(x, y)
plt.grid()
plt.show()
#Two  lines to make our compiler able to draw:
plt.savefig(sys.stdout.buffer)
sys.stdout.flush()
```

Output:


Sports Watch Data
Calorie Burnage
320
300
280
260
240
80
90
100
110
120
Average Pulse

Lab 18

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Simulate correlation and convolution operation discrete time sequences.

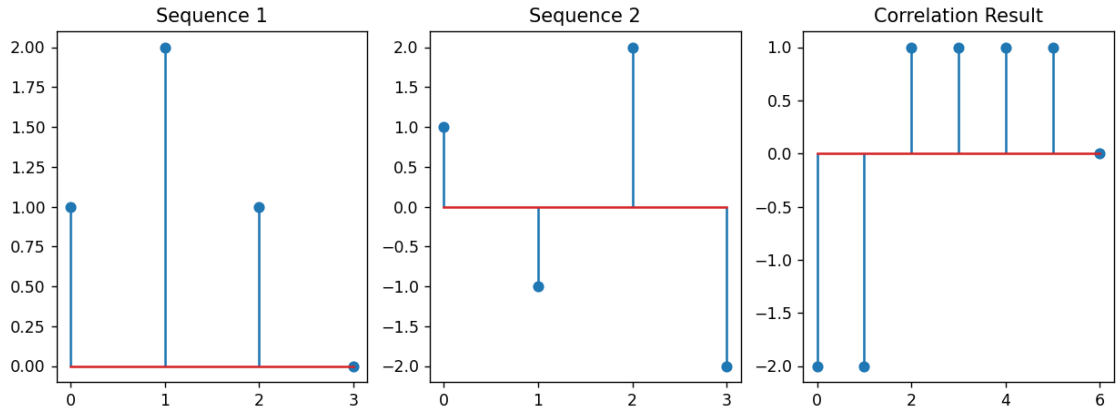
Introduction:- Correlation and convolution are fundamental operations in signal processing and mathematics, particularly in the context of discrete-time sequences. These operations are used to analyze, process, and manipulate signals in various applications, including image processing, audio processing, and communication systems.

Task1: Write a python program for simple correlation.

Python Code:

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

# Define two discrete-time sequences
sequence1 = np.array([1, 2, 1, 0])
sequence2 = np.array([1, -1, 2, -2])

#  Correlation
correlation_result = np.correlate(sequence1, sequence2, mode='full')

plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.stem(sequence1, use_line_collection=True)
plt.title('Sequence 1')
plt.subplot(1, 3, 2)
plt.stem(sequence2, use_line_collection=True)
plt.title('Sequence 2')
plt.subplot(1, 3, 3)
plt.stem(correlation_result, use_line_collection=True)
plt.title('Correlation Result')
plt.show()
```

Output:

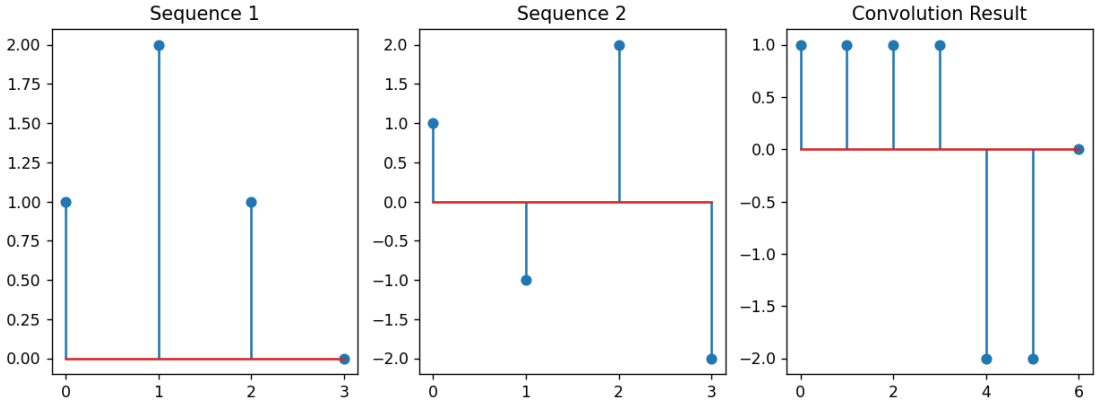


Tas2: Write a python program for simple convolution.

Python Code:

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

# Define two discrete-time sequences
sequence1 = np.array([1, 2, 1, 0])
sequence2 = np.array([1, -1, 2, -2])
#Convolution
convolution_result = np.convolve(sequence1, sequence2, mode='full')

plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.stem(sequence1, use_line_collection=True)
plt.title('Sequence 1')
plt.subplot(1, 3, 2)
plt.stem(sequence2, use_line_collection=True)
plt.title('Sequence 2')
plt.subplot(1, 3, 3)
plt.stem(convolution_result, use_line_collection=True)
plt.title('Convolution Result')
plt.show()
```

Output:

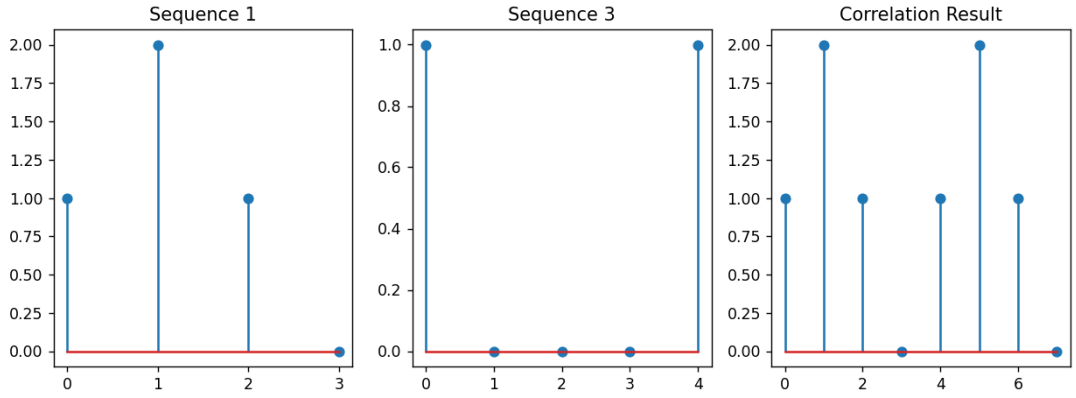


Task3: Write a python program for correlation with different sequences.

Python Code:

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

# Define two discrete-time sequences
sequence1 = np.array([1, 2, 1, 0])
sequence2 = np.array([1, -1, 2, -2])
#Correlation with different sequences
sequence3 = np.array([1, 0, 0, 0, 1])
correlation_result2 = np.correlate(sequence1, sequence3, mode='full')

plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.stem(sequence1, use_line_collection=True)
plt.title('Sequence 1')
plt.subplot(1, 3, 2)
plt.stem(sequence3, use_line_collection=True)
plt.title('Sequence 3')
plt.subplot(1, 3, 3)
plt.stem(correlation_result2, use_line_collection=True)
plt.title('Correlation Result')
plt.show()
```

Output:

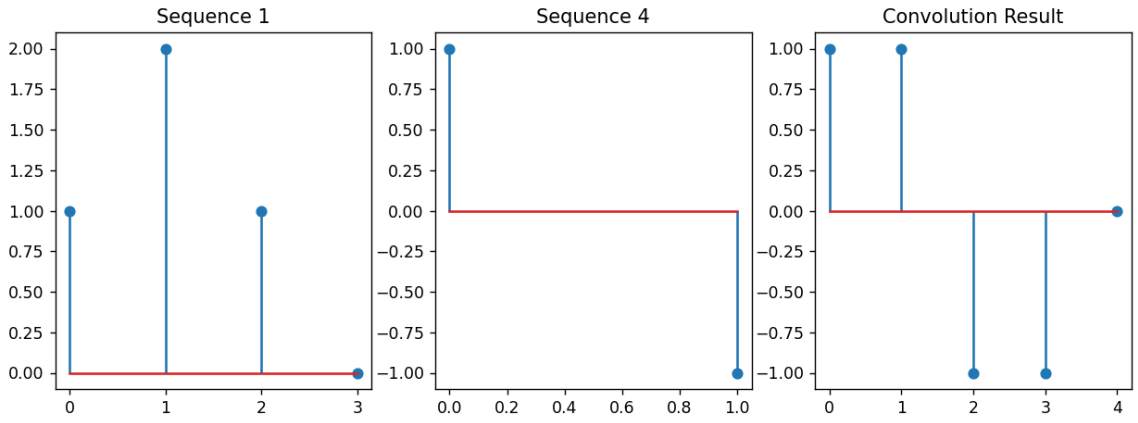


Task4: Write a python program for convolution with different sequences.

Python Code:

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

# Define two discrete-time sequences
sequence1 = np.array([1, 2, 1, 0])
sequence2 = np.array([1, -1, 2, -2])
#Convolution with different sequences
sequence4 = np.array([1, -1])
convolution_result2 = np.convolve(sequence1, sequence4, mode='full')

plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.stem(sequence1, use_line_collection=True)
plt.title('Sequence 1')
plt.subplot(1, 3, 2)
plt.stem(sequence4, use_line_collection=True)
plt.title('Sequence 4')
plt.subplot(1, 3, 3)
plt.stem(convolution_result2, use_line_collection=True)
plt.title('Convolution Result')
plt.show()
```

Output:

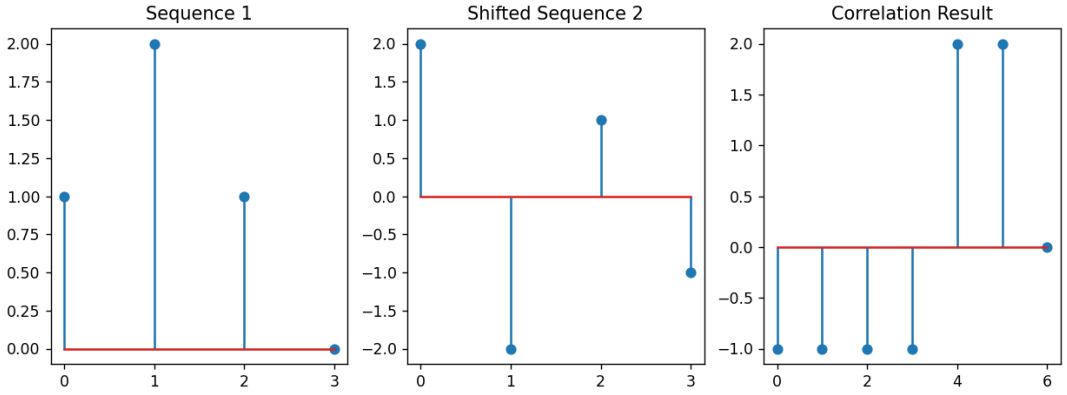


Task5: Write a python program for correlation with a shift sequence.

Python Code:

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

# Define two discrete-time sequences
sequence1 = np.array([1, 2, 1, 0])
sequence2 = np.array([1, -1, 2, -2])
#Correlation with a shifted sequence
shifted_sequence = np.roll(sequence2, shift=2)  # Shift sequence2 by 2 positions
correlation_result3 = np.correlate(sequence1, shifted_sequence, mode='full')

plt.figure(figsize=(12, 4))
plt.subplot(1, 3, 1)
plt.stem(sequence1, use_line_collection=True)
plt.title('Sequence 1')
plt.subplot(1, 3, 2)
plt.stem(shifted_sequence, use_line_collection=True)
plt.title('Shifted Sequence 2')
plt.subplot(1, 3, 3)
plt.stem(correlation_result3, use_line_collection=True)
plt.title('Correlation Result')
plt.show()
```

Output:



Lab 19

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Write script named copyfile.py. This script should prompt the user for the names of two text file. The contents of the first the second file.

Introduction:- As our program grows bigger, it may contain many lines of code. Instead of putting everything in a single file, we can use modules to separate codes in separate files as per their functionality. This makes our code organized and easier to maintain. Module is a file that contains code to perform a specific task. A module may contain variables, functions, classes etc.

Task1: Write a python program using file handling to read and append.

Python Code:

```
# open both files
with open('first.txt','r') as firstfile, open('second.txt','a') as secondfile:

    # read content from first file
    for line in firstfile:

        # append content to second file
        secondfile.write(line)
```

Output:

Task2: Write a python program using to read and write.

Python Code:

```
# open both files
with open('first.txt','r') as firstfile, open('second.txt','w') as secondfile:

    # read content from first file
    for line in firstfile:

        # write content to second file
        secondfile.write(line)
```

Output:

Task3: Write a python program to convert temperature into different measures.

Python Code:

```
# import module
import shutil
# use copyfile()
shutil.copyfile('first.txt','second.txt')
```

Output:

Lab 20

Name: - Aryan Dilipbhai Langhanoja

Date:

Enrollment No: 92200133030

CO1: To write, test, and debug simple Python programs

CO2: To implement Python programs with conditional, loops and functions

AIM:- Write a GUI program to create Tic-tac-toe in python.

Introduction:- Tic-tac-toe (American English), noughts and crosses (British English), or Xs and Os is a paper-and-pencil game for two players, X and O, who take turns marking the spaces in a 3×3 grid. The player who succeeds in placing three of their marks in a horizontal, vertical, or diagonal row is the winner.tkinter Python library is used to create the GUI. Two options are available to play the game, along with the system or with another player.

Task1: Write a python program to create Tic-tac-toe game.

Python Code:

```
import tkinter as tk
from tkinter import messagebox
# Initialize variables
current_player = "X"
board = [""] * 9
# Create the main game window
window = tk.Tk()
window.title("Tic-Tac-Toe")
# Function to handle button clicks
def button_click(index):
    global current_player
    if board[index] == "":
        board[index] = current_player
        buttons[index].config(text=current_player)
        if check_winner(current_player):
            messagebox.showinfo("Tic-Tac-Toe", f"Player {current_player} wins!")
            reset_game()
        elif "" not in board:
            messagebox.showinfo("Tic-Tac-Toe", "It's a draw!")
            reset_game()
        else:
            current_player = "O" if current_player == "X" else "X
```

```
# Function to check for a winner
def check_winner(player):
    winning_combinations = [
        [0, 1, 2], [3, 4, 5], [6, 7, 8],  # Rows
        [0, 3, 6], [1, 4, 7], [2, 5, 8],  # Columns
        [0, 4, 8], [2, 4, 6]              # Diagonals
    ]
    for combo in winning_combinations:
        if board[combo[0]] == board[combo[1]] == board[combo[2]] == player:
            return True
    return False
# Function to reset the game
def reset_game():
    global board, current_player
    board = [""] * 9
    current_player = "X"
    for button in buttons:
        button.config(text="", state="active")
# Create buttons for the game board
buttons = []
for i in range(9):
    row = i // 3
    col = i % 3
    button = tk.Button(window, text="", width=10, height=3,
                       command=lambda i=i: button_click(i))
    button.grid(row=row, column=col)
    buttons.append(button)
# Start the game
window.mainloop()
```

Output: