1. Python Program for factorial of a number

```
# Python 3 program to find
# factorial of given number
def factorial(n):

    # single line to find factorial
    return 1 if (n==1 or n==0) else n * factorial(n - 1)
```

2. Python program to print all Prime numbers in an Interval

```
def print_primes_in_interval(start, end):
    primes = []
    for num in range(start, end + 1):
        if num > 1:
            for i in range(2, num):
                if (num % i) == 0:
                    break
            else:
                primes.append(num)
    print("Prime numbers between", start, "and", end, "are:", primes)

# Example usage:
start = 10
end = 50
print_primes_in_interval(start, end)
```

3. Python Program for n-th Fibonacci number

```
def fibonacci(n):
    if n <= 1:
        return n
    else:
        return fibonacci(n-1) + fibonacci(n-2)

# Example usage:
n = 10
print(f"The {n}-th Fibonacci number is:", fibonacci(n))
```

4. Python Program for Sum of squares of first n natural numbers

```
def sum_of_squares(n):
    return sum(i**2 for i in range(1, n+1))

# Example usage:
n = 5
print("Sum of squares of the first", n, "natural numbers is:", sum_of_squares(n))
```

5. Python Program to find largest element in an array

```
def find_largest(arr):
    return max(arr)

# Example usage:
array = [10, 5, 20, 8, 15]
print("The largest element in the array is:", find_largest(array))
```

6. Python program to find largest number in a list

```
def find_largest(arr):
    return max(arr)

# Example usage:
numbers = [10, 5, 20, 8, 15]
print("The largest number in the list is:", find_largest(numbers))
```

7. Python program to print all even numbers in a range

```
def print_even_numbers(start, end):
    even_numbers = [num for num in range(start, end + 1) if num % 2 == 0]
    print("Even numbers between", start, "and", end, "are:", even_numbers)


# Example usage:
start = 1
end = 20
print_even_numbers(start, end)
```

8. Remove multiple elements from a list in Python

```
def remove_elements(lst, indices):
    return [elem for index, elem in enumerate(lst) if index not in indices]


# Example usage:
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
indices_to_remove = [1, 3, 5]  # Indices of elements to remove
new_list = remove_elements(my_list, indices_to_remove)
print("Original list:", my_list)
print("List after removal:", new_list)
```

9. Break a list into chunks of size N in Python

```
def chunk_list(lst, chunk_size):
    for i in range(0, len(lst), chunk_size):
        yield lst[i:i + chunk_size]


# Example usage:
my_list = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
chunk_size = 3
chunks = list(chunk_list(my_list, chunk_size))
```

```
print("Original list:", my_list)
print(f"List broken into chunks of size {chunk_size}:", chunks)
```

10. Python program to multiply two matrices

```
def multiply_matrices(matrix1, matrix2):
    return [[sum(a * b for a, b in zip(row1, col2)) for col2 in zip(*matrix2)] for row1 in matrix1]

# Example usage:
matrix1 = [[1, 2, 3],
           [4, 5, 6],
           [7, 8, 9]]

matrix2 = [[9, 8, 7],
           [6, 5, 4],
           [3, 2, 1]]

result = multiply_matrices(matrix1, matrix2)

print("Matrix 1:")
for row in matrix1:
    print(row)

print("\nMatrix 2:")
for row in matrix2:
    print(row)

print("\nResultant Matrix:")
for row in result:
    print(row)
```

11. Python program to check if a string is palindrome or not

```
def is_palindrome(s):
    s = ''.join(char.lower() for char in s if char.isalnum())
    return s == s[::-1]


# Example usage:
string = "A man, a plan, a canal, Panama"
print("Is the string a palindrome?", is_palindrome(string))
```

12. Ways to remove i'th character from string in Python

```
def remove_character(s, i):
    return s[:i] + s[i+1:]


# Example usage:
string = "hello"
index = 2
print("String after removing the character at index", index, ":", remove_character(string, index))
```

13. Python program to accept the strings which contains all vowels

```
def contains_all_vowels(s):
    return all(char in s.lower() for char in 'aeiou')


# Example usage:
string = "A quick brown fox jumps over the lazy dog"
if contains_all_vowels(string):
    print("The string contains all vowels.")
else:
    print("The string does not contain all vowels.")
```

14. Remove all duplicates from a given string in Python

```
def remove_duplicates(s):
    return ''.join(set(s))


# Example usage:
string = "hello"
print("String after removing duplicates:", remove_duplicates(string))
```

15. Python program to split and join a string

```
def split_and_join(s):
    return '-'.join(s.split())


# Example usage:
string = "This is a sample string"
print("Original string:", string)
result = split_and_join(string)
print("String after split and join:", result)
```

16. Python – Replace duplicate Occurrence in String

```
def replace_duplicate_occurrences(s):
    return ''.join('$' if s.count(char) > 1 else char for char in s)


# Example usage:
string = "hello"
print("Original string:", string)
result = replace_duplicate_occurrences(string)
print("String after replacing duplicate occurrences:", result)
```

17. Python program to find the sum of all items in a dictionary

```
def sum_dictionary_values(dictionary):
    return sum(dictionary.values())


# Example usage:
my_dict = {'a': 10, 'b': 20, 'c': 30}
print("Sum of all items in the dictionary:", sum_dictionary_values(my_dict))
```

18. Python – Sort Dictionary key and values List

```
def sort_dict_by_keys(d):
    return {k: d[k] for k in sorted(d)}


# Example usage:
my_dict = {'b': 3, 'a': 2, 'c': 1}
sorted_dict = sort_dict_by_keys(my_dict)
print("Sorted dictionary by keys:", sorted_dict)
```

19. Python Dictionary to find mirror characters in a string

```
def find_mirror_characters(s):
    mirror_dict = {'b': 'd', 'd': 'b', 'p': 'q', 'q': 'p'}
    return [(char, mirror_dict[char]) for char in s if char in mirror_dict]


# Example usage:
string = "bedqp"
mirror_characters = find_mirror_characters(string)
print("Mirror characters in the string:", mirror_characters)
```

20. Python – Adding Tuple to List and vice – versa

```
# Example usage:
my_tuple = (1, 2, 3)
```

```
my_list = [4, 5, 6]

my_list.append(my_tuple)

print("List after adding tuple:", my_list)
```

2)

```
# Example usage:

my_list = [4, 5, 6]

my_tuple = (1, 2, 3)

updated_tuple = my_tuple + tuple(my_list)

print("Tuple after adding list:", updated_tuple)
```

21. Python – Convert Nested Tuple to Custom Key Dictionary

```
def nested_tuple_to_dict(nested_tuple, keys):

    return [{keys[i]: item[i] for i in range(len(keys))} for item in nested_tuple]


# Example usage:

nested_tuple = (('John', 25), ('Jane', 30), ('Jim', 35))

keys = ['name', 'age']

custom_dict = nested_tuple_to_dict(nested_tuple, keys)

print("Custom key dictionary:", custom_dict)
```

22. Python Program to print an Inverted Star Pattern

```
def inverted_star_pattern(rows):

    for i in range(rows, 0, -1):

        print("*" * i)


# Example usage:

rows = 5

inverted_star_pattern(rows)
```

23. Python Program to print double sided stair-case pattern

```
def double_sided_staircase(rows):
    for i in range(1, rows * 2):
        if i <= rows:
            print("*" * i)
        else:
            print("*" * (rows * 2 - i))

# Example usage:
rows = 5
double_sided_staircase(rows)
```

24. Python program to convert time from 12 hour to 24 hour format

```
def convert_12_to_24(time_12h):
    return '{:02d}:{:02d}'.format((int(time_12h[:2]) % 12) + (12 if 'PM' in time_12h else 0),
int(time_12h[3:5]))

# Example usage:
time_12h = "04:30 PM"
print("Time in 12-hour format:", time_12h)
print("Time in 24-hour format:", convert_12_to_24(time_12h))
```

25. Python program to find difference between current time and given time

```
from datetime import datetime

def time_difference(given_time):
    given_time_obj = datetime.strptime(given_time, '%H:%M')
    current_time_obj = datetime.now()
    difference = given_time_obj - current_time_obj
    return difference.total_seconds() // 60
```

```
# Example usage:
given_time = "15:30"
print("Given time:", given_time)
print("Difference from current time (in minutes):", time_difference(given_time))
```

30. Python Program for Print Number series without using any loop

```
def print_series(n):
    print(*range(1, n + 1), sep='\n')


# Example usage:
n = 5
print_series(n)
```

31. Write a program to create bank account class with two attributes. Description: Write a class with 2 attributes(owner and balance). In this assignment you need to maintain a bank account where 2 operations need to be done repeatedly. First one is —deposit|| and the other operation is —Withdraw||. If the user selects the withdrawal operation, then you need to check whether the owner has sufficient bank balance or not.

```
class BankAccount:
    def __init__(self, owner, balance=0):
        self.owner = owner
        self.balance = balance


    def deposit(self, amount):
        self.balance += amount
        print(f"Deposit of ${amount} accepted. Current balance: ${self.balance}")


    def withdraw(self, amount):
        if amount <= self.balance:
            self.balance -= amount
        print(f"Withdrawal of ${amount} accepted. Current balance: ${self.balance}")
    else:
        print("Insufficient funds!")

# Example usage:
account = BankAccount("John Doe", 1000)
print(f"Account owner: {account.owner}, Balance: ${account.balance}")
account.deposit(500)
account.withdraw(200)
account.withdraw(1500)
```

32. Python program to add two integers with handling expectations Write a Python program input and add two integers only and handle the exceptions. Problem Solution: In this program, we are reading two integers number from the user using int(input()) an handling the following exceptions, ValueError – Occurs when input value is not an integer. ZeroDivisionError – Occurs when divisor is zero. Exception – Any other error

```
def add_two_integers():
    try:
        num1 = int(input("Enter the first integer: "))
        num2 = int(input("Enter the second integer: "))

        result = num1 + num2
        print("Sum:", result)

    except ValueError:
        print("Error: Please enter integers only.")

    except ZeroDivisionError:
        print("Error: Division by zero is not allowed.")

    except Exception as e:
        print("Error:", e)
```

```
# Example usage:

add_two_integers()
```

33. Write a program to add two numbers by taking these values as inputs and display the sum as the output

```
def add_two_numbers():

   num1 = float(input("Enter the first number: "))

   num2 = float(input("Enter the second number: "))

   sum = num1 + num2

   print("Sum:", sum)


# Example usage:

add_two_numbers()
```

34. Write a program to take input of age and depending on the age output a pop-up message showing if the person is eligible to vote or not.

```
def check_voting_eligibility(age):

   if age >= 18:

      print("You are eligible to vote!")

   else:

      print("You are not eligible to vote.")


# Example usage:

age = int(input("Enter your age: "))

check_voting_eligibility(age)
```

35. Menu Driven program to create a simple calculator.

```
def add(x, y):
   return x + y

def subtract(x, y):
   return x - y

def multiply(x, y):
   return x * y

def divide(x, y):
   if y == 0:
       return "Cannot divide by zero"
   else:
       return x / y

print("Select operation:")
print("1. Add")
print("2. Subtract")
print("3. Multiply")
print("4. Divide")

choice = input("Enter choice (1/2/3/4): ")

if choice in ('1', '2', '3', '4'):
   num1 = float(input("Enter first number: "))
   num2 = float(input("Enter second number: "))

   if choice == '1':
       print("Result:", add(num1, num2))
   elif choice == '2':
       print("Result:", subtract(num1, num2))
   elif choice == '3':
       print("Result:", multiply(num1, num2))
   elif choice == '4':
       print("Result:", divide(num1, num2))
else:
   print("Invalid input")
```