✅ 1. Menu-driven program for all built-in functions of Set

```
def set_operations():
    s = set()
    while True:
        print("\n1. Add element")
        print("2. Remove element")
        print("3. Clear set")
        print("4. Pop element")
        print("5. Display set")
        print("6. Exit")
        choice = input("Enter choice: ")

        if choice == '1':
            val = input("Enter value to add: ")
            s.add(val)
        elif choice == '2':
            val = input("Enter value to remove: ")
            s.discard(val)
        elif choice == '3':
            s.clear()
        elif choice == '4':
            if s:
                print("Popped:", s.pop())
        elif choice == '5':
            print("Set:", s)
        elif choice == '6':
```

```
            break

set_operations()
```

✅ 2. Menu-driven program for 5 built-in functions of Tuple

```
def tuple_operations():
    t = (1, 2, 3, 4, 5, 2)
    while True:
        print("\n1. Count")
        print("2. Index")
        print("3. Length")
        print("4. Max")
        print("5. Min")
        print("6. Exit")
        choice = input("Enter choice: ")

        if choice == '1':
            print("2 appears", t.count(2), "times")
        elif choice == '2':
            print("Index of 3:", t.index(3))
        elif choice == '3':
            print("Length:", len(t))
        elif choice == '4':
            print("Max:", max(t))
        elif choice == '5':
            print("Min:", min(t))
```

```
        elif choice == '6':
            break

tuple_operations()
```

✅ 3. Menu-driven program for 5 built-in functions of Dictionary

```
def dict_operations():
    d = {}
    while True:
        print("\n1. Add/Update")
        print("2. Remove key")
        print("3. Display all keys")
        print("4. Display all values")
        print("5. Check key exists")
        print("6. Exit")
        choice = input("Enter choice: ")

        if choice == '1':
            k = input("Key: ")
            v = input("Value: ")
            d[k] = v
        elif choice == '2':
            k = input("Key to remove: ")
            d.pop(k, None)
        elif choice == '3':
            print("Keys:", d.keys())
```

```
        elif choice == '4':
            print("Values:", d.values())
        elif choice == '5':
            k = input("Key to check: ")
            print("Exists" if k in d else "Not found")
        elif choice == '6':
            break

dict_operations()
```

✅ 4. Count number of occurrences of all vowels in a string

```
s = input("Enter a string: ").lower()
vowels = 'aeiou'
for v in vowels:
    print(f"{v}: {s.count(v)}")
```

✅ 5. Menu-driven program for 5 built-in functions of List

```
def list_operations():
    lst = []
    while True:
        print("\n1. Append")
        print("2. Insert")
        print("3. Remove")
        print("4. Sort")
        print("5. Reverse")
        print("6. Display")
```

```
        print("7. Exit")
        choice = input("Enter choice: ")

        if choice == '1':
            lst.append(input("Enter element: "))
        elif choice == '2':
            lst.insert(int(input("Index: ")), input("Value: "))
        elif choice == '3':
            lst.remove(input("Value to remove: "))
        elif choice == '4':
            lst.sort()
        elif choice == '5':
            lst.reverse()
        elif choice == '6':
            print("List:", lst)
        elif choice == '7':
            break

list_operations()
```

✅ 6. Class and user-defined method to check Palindrome

```
class Palindrome:
    def check(self, s):
        if s == s[::-1]:
            print("Palindrome")
        else:
```

```
            print("Not Palindrome")


p = Palindrome()

p.check(input("Enter string: "))
```

✅ 7. Class to create dictionary of squares from 1 to 30

```
class SquareDict:

    def create(self):

        print({i: i**2 for i in range(1, 31)})


s = SquareDict()

s.create()
```

✅ 8. Class and function to implement Stack

```
class Stack:

    def __init__(self):

        self.stack = []


    def push(self, val):

        self.stack.append(val)


    def pop(self):

        if self.stack:

            print("Popped:", self.stack.pop())


    def display(self):
```

```
        print("Stack:", self.stack)


s = Stack()
s.push(10)
s.push(20)
s.display()
s.pop()
s.display()
```

✅ 9. Class and function for Linear Search

```
class Search:
    def linear_search(self, lst, target):
        for i in range(len(lst)):
            if lst[i] == target:
                return i
        return -1


s = Search()
lst = [1, 3, 5, 7, 9]
target = int(input("Enter number to search: "))
res = s.linear_search(lst, target)
print("Found at index" if res != -1 else "Not found")
```

✅ 10. Shuffle and print a list using `random`

```
import random
lst = list(range(1, 11))
```

```
random.shuffle(lst)

print("Shuffled list:", lst)
```

✅ 11. All permutations using `itertools`

```
import itertools

for p in itertools.permutations([1,2,3,4,5,6,7,8,9]):
    print(p)
```

✅ 12. Compress and decompress using `zlib`

```
import zlib

s = input("Enter a string: ").encode()

compressed = zlib.compress(s)

print("Compressed:", compressed)

print("Decompressed:", zlib.decompress(compressed).decode())
```

✅ 13. Words without letter 'E'

```
def has_no_e(word):
    return 'e' not in word.lower()


words = ["hello", "sky", "world", "try", "run"]

count = 0

for w in words:
    if has_no_e(w):
        print(w)
        count += 1

print("Percentage:", (count/len(words))*100, "%")
```

✅ 14. Birthday age & next birthday countdown

```
from datetime import datetime, timedelta


class Birthday:
    def calculate(self, birthdate):
        today = datetime.now()
        dob = datetime.strptime(birthdate, "%Y-%m-%d")
        age = today.year - dob.year - ((today.month, today.day) < (dob.month, dob.day))


        next_birthday = dob.replace(year=today.year + 1 if today >= dob.replace(year=today.year) else today.year)
        diff = next_birthday - today


        print(f"Age: {age}")
        print(f"Time until next birthday: {diff.days} days, {diff.seconds // 3600} hours")


b = Birthday()
b.calculate(input("Enter birthday (YYYY-MM-DD): "))
```

✅ 15. Read file line-by-line and display

```
def read_lines():
    with open("file.txt", "r") as f:
        for line in f:
            print(line, end="")
```

```
read_lines()
```

✅ 16. Count lines not starting with capital 'T'

```
def count_lines():
    count = 0
    with open("file.txt", "r") as f:
        for line in f:
            if not line.startswith('T'):
                count += 1
    print("Lines not starting with 'T':", count)


count_lines()
```

✅ 17. Count and display total number of words

```
def count_words():
    with open("file.txt", "r") as f:
        words = f.read().split()
        print("Total words:", len(words))


count_words()
```

✅ 18. Write a Python program to print "Hello Python" and display the version of Python installed.

```
import sys

print("Hello Python")

print("Python version:", sys.version)
```

✅ 19. Write a program to check whether a number is even or odd.

```
num = int(input("Enter a number: "))
if num % 2 == 0:
    print("Even number")
else:
    print("Odd number")
```

✅ 20. Write a program to find the largest among three numbers.

```
a = int(input("Enter first number: "))
b = int(input("Enter second number: "))
c = int(input("Enter third number: "))
if a >= b and a >= c:
    print("Largest:", a)
elif b >= c:
    print("Largest:", b)
else:
    print("Largest:", c)
```

✅ 21. Write a program to calculate factorial using a loop.

```
n = int(input("Enter a number: "))
fact = 1
for i in range(1, n+1):
    fact *= i
print("Factorial:", fact)
```

✅ 22. Create a list and demonstrate append, insert, pop, reverse, and sort.

```
lst = [3, 1, 4]

lst.append(9)

lst.insert(1, 7)

lst.pop()

lst.reverse()

lst.sort()

print(lst)
```

✅ 23. Demonstrate built-in tuple methods: count(), index().

```
tpl = (1, 2, 2, 3)

print(tpl.count(2))

print(tpl.index(3))
```

✅ 24. Create a dictionary of fruits and prices. Add and delete an item.

```
fruits = {'apple': 50, 'banana': 20}

fruits['orange'] = 30

del fruits['banana']

print(fruits)
```

✅ 25. Create a set and demonstrate union, intersection, difference.

```
a = {1, 2, 3}

b = {2, 3, 4}

print(a.union(b))

print(a.intersection(b))

print(a.difference(b))
```

✅ 26. Replace all vowels with ‘*’ in a string.

```
s = input("Enter string: ")

vowels = "aeiouAEIOU"

new_str = ''.join(['*' if ch in vowels else ch for ch in s])

print(new_str)
```

✅ 27. Count number of words in a file.

```
def count_words():
    with open("sample.txt", "r") as f:
        data = f.read()
        print("Word count:", len(data.split()))
```

✅ 28. Program to handle divide by zero using try-except.

```
try:
    a = int(input("Enter numerator: "))
    b = int(input("Enter denominator: "))
    print("Result:", a / b)
except ZeroDivisionError:
    print("Cannot divide by zero!")
```

✅ 29. Class to check whether a number is palindrome.

```
class Palindrome:
    def check(self, num):
        return str(num) == str(num)[::-1]
```

```
obj = Palindrome()

print(obj.check(int(input("Enter number: "))))
```

✅ 30. Implement a stack using class.

```
class Stack:
    def __init__(self):
        self.stack = []

    def push(self, item):
        self.stack.append(item)

    def pop(self):
        if not self.stack:
            return "Empty"
        return self.stack.pop()

    def display(self):
        print(self.stack)

s = Stack()
s.push(5)
s.push(10)
s.display()
s.pop()
s.display()
```

✅ 31. Using pandas read a CSV and display first 5 rows.

```
import pandas as pd

df = pd.read_csv("data.csv")

print(df.head())
```

✅ 32. Create two NumPy arrays and add them element-wise.

```
import numpy as np

a = np.array([1, 2, 3])

b = np.array([4, 5, 6])

print(np.add(a, b))
```

✅ 33. Generate a bar chart using Matplotlib.

```
import matplotlib.pyplot as plt

x = ['Math', 'Science', 'English']

y = [90, 80, 75]

plt.bar(x, y)

plt.title("Marks")

plt.show()
```