PYTHON RECORD

16-08-2025

 # 1. Ask for first name

```
first_name = input("Enter your first name: ")
print("Hello", first_name)
```

OUTPUT:

```
Enter your first name: John
Hello John
```

 # 2. Ask for first name and surname

```
first_name = input("Enter your first name: ")
surname = input("Enter your surname: ")
print("Hello", first_name, surname)
```

OUTPUT:

```
Enter your first name: John
Enter your surname: Smith
Hello John Smith
```

 # 3. Ask for two numbers and add them

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
total = num1 + num2
print("The total is", total)
```

OUTPUT:

```
Enter first number: 4
Enter second number: 6
The total is 10.0
```

 # 4. Ask for three numbers, add first two, multiply by third

```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
num3 = float(input("Enter third number: "))
answer = (num1 + num2) * num3
print("The answer is", answer)
```

OUTPUT:

```
Enter first number: 2
```

Enter second number: 3
Enter third number: 4
The answer is 20.0

 # 5. Divide total bill among diners

```
bill = float(input("Enter the total bill amount: "))
diners = int(input("Enter number of diners: "))
each = bill / diners
print("Each person must pay", round(each, 2))
```

OUTPUT:

Enter the total bill amount: 120
Enter number of diners: 4
Each person must pay 30.0

 # 6. Convert days to hours, minutes, and seconds

```
days = int(input("Enter number of days: "))
hours = days * 24
minutes = hours * 60
seconds = minutes * 60
print(days, "days =", hours, "hours,", minutes, "minutes,", seconds, "seconds")
```

OUTPUT:

Enter number of days: 2
2 days = 48 hours, 2880 minutes, 172800 seconds

 # 7. How many times smaller number goes into larger number

```
large = int(input("Enter a number over 100: "))
small = int(input("Enter a number under 10: "))
times = large // small
print(small, "goes into", large, times, "times")
```

OUTPUT:

Enter a number over 100: 120
Enter a number under 10: 5
5 goes into 120 24 times

 # 8. Convert kilograms to pounds

```
kg = float(input("Enter weight in kilograms: "))
pounds = kg * 2.204
```

```
print(kg, "kilograms is", round(pounds, 3), "pounds")
```
OUTPUT:
```
Enter weight in kilograms: 50
50.0 kilograms is 110.2 pounds
```

 # 9. Pizza slices
```
slices_start = int(input("How many slices of pizza did you start with? "))
slices_eaten = int(input("How many slices have you eaten? "))
slices_left = slices_start - slices_eaten
print("You have", slices_left, "slices left.")
```
OUTPUT:
```
How many slices of pizza did you start with? 8
How many slices have you eaten? 3
You have 5 slices left.
```

 # 10. Display numbers in order
```
num1 = float(input("Enter first number: "))
num2 = float(input("Enter second number: "))
if num1 > num2:
   print(num2, num1)
else:
   print(num1, num2)
```
OUTPUT:
```
Enter first number: 20
Enter second number: 10
10.0 20.0
```

 # 11. Area of a circle
```
import math
radius = float(input("Enter the radius of the circle: "))
area = math.pi * radius ** 2
print("The area of the circle is", round(area, 2))
```
OUTPUT:
```
Enter the radius of the circle: 5
The area of the circle is 78.54
```

 # 12. Celsius to Fahrenheit

```
celsius = float(input("Enter temperature in Celsius: "))
fahrenheit = (celsius * 9/5) + 32
print(celsius, "°C is", fahrenheit, "°F")
```

OUTPUT:

```
Enter temperature in Celsius: 25
25.0 °C is 77.0 °F
```

 # 13. Check if a number is positive or not; if positive, check even/odd

```
num = int(input("Enter a number: "))
if num > 0:
    print("The number is positive.")
    if num % 2 == 0:
        print("It is even.")
    else:
        print("It is odd.")
else:
    print("The number is not positive.")
```

OUTPUT:

```
Enter a number: 7
The number is positive.
It is odd.
```

```
# 14. Simple calculator
print("\n--- Simple Calculator ---")
print("1. Add\n2. Subtract\n3. Multiply\n4. Divide")
choice = input("Choose an operation (1/2/3/4): ")
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
if choice == "1":
    print("Result:", a + b)
elif choice == "2":
    print("Result:", a - b)
elif choice == "3":
    print("Result:", a * b)
elif choice == "4":
```

```
    if b != 0:
        print("Result:", a / b)
    else:
        print("Error: Division by zero!")
else:
    print("Invalid choice.")
```

OUTPUT:

```
--- Simple Calculator ---
1. Add
2. Subtract
3. Multiply
4. Divide
Choose an operation (1/2/3/4): 3
Enter first number: 5
Enter second number: 6
Result: 30.0
```

19-08-2025

#1. Find A = 1 / 2 * h * (a + b)

```
h = float(input("Enter height: "))
a = float(input("Enter base a: "))
b = float(input("Enter base b: "))
A = 1/2 * h * (a + b)
print("Area is:", A)
```

OUTPUT:

```
Enter height: 4
Enter base a: 5
Enter base b: 7
Area is: 24.0
```

#2.Convert Celsius ↔ Fahrenheit

```
# Celsius to Fahrenheit
c = float(input("Enter temperature in Celsius: "))
f = (9/5) * c + 32
print("Temperature in Fahrenheit:", f)
```

```
# Fahrenheit to Celsius
f = float(input("Enter temperature in Fahrenheit: "))
c = (f - 32) * 5/9
print("Temperature in Celsius:", c)
```

OUTPUT:

```
Enter temperature in Celsius: 25
Temperature in Fahrenheit: 77.0
Enter temperature in Fahrenheit: 77
Temperature in Celsius: 25.0
```

#3. Calculator Operation

```
a = float(input("Enter first number: "))
b = float(input("Enter second number: "))
print("Select operation: + - * /")
op = input("Enter operator: ")
if op == '+':
    print("Result =", a + b)
elif op == '-':
    print("Result =", a - b)
elif op == '*':
    print("Result =", a * b)
elif op == '/':
    print("Result =", a / b)
else:
    print("Invalid operator")
```

OUTPUT:

```
Enter first number: 10
Enter second number: 5
Select operation: + - * /
Enter operator: *
Result = 50.0
```

#4. Output of the following code

```
num1 = '10'
num2 = '20'
sum = num1 + num2
print(sum)
```

OUTPUT:
1020

#5. Identify the Error
num1 = '10'
num2 = 20.65
sum = num1 + num2
print(sum)

OUTPUT:
TypeError: can only concatenate str (not "float") to str

#6. Invoice of a Product
cost = 5000
cgst = cost * 0.09
sgst = cost * 0.09
total = cost + cgst + sgst
print("Cost of Production:", cost)
print("Add: CGST @ 9%:", cgst)
print("Add: SGST @ 9%:", sgst)
print("Total Cost of Product:", total)
OUTPUT:
Cost of Production: 5000
Add: CGST @ 9%: 450.0
Add: SGST @ 9%: 450.0
Total Cost of Product: 5900.0

OPERATORS:
#1. Convert grams to kilograms and grams
grams = int(input("Enter weight in grams: "))
kg = grams // 1000
g = grams % 1000
print(kg, "kilograms and", g, "grams")
OUTPUT:
Enter weight in grams: 2575
2 kilograms and 575 grams

#2. Identity Operator Example

```
x1 = 5
y1 = 5
x2 = 'Hello'
y2 = 'Hello'
x3 = [1,2,3]
y3 = [1,2,3]
print(x1 is y1)
print(x2 is y2)
print(x3 is y3)
```

OUTPUT:

True
True
False

CONDITIONAL STATEMENTS

#1. Student Mark Analysis

```
mark = int(input("Enter mark: "))
if mark >= 90:
    print("Grade: A+")
elif mark >= 75:
    print("Grade: A")
elif mark >= 60:
    print("Grade: B")
elif mark >= 40:
    print("Grade: C")
else:
    print("Fail")
```

OUTPUT:

Enter mark: 85
Grade: A

#2.Check data type, number/positive/even/odd

```
value = input("Enter something: ")
if value.isdigit():
    num = int(value)
    if num > 0:
        print("Positive")
```

```
        if num % 2 == 0:
            print("Even")
        else:
            print("Odd")
    else:
        print("Negative")
else:
    print("Not a number, data type:", type(value))
```

OUTPUT:

```
Enter something: 8
Positive
Even
```

#3. Simple Interest

```
p = float(input("Enter principal amount: "))
t = float(input("Enter number of years: "))
if p > 100000:
    r = 5
else:
    r = 3.5
si = (p * r * t) / 100
print("Simple Interest:", si)
```

OUTPUT:

```
Enter principal amount: 120000
Enter number of years: 2
Simple Interest: 12000.0
```

LOOPING STATEMENTS

#1.Add 10 consecutive numbers starting from 1

```
i = 1
total = 0
while i <= 10:
    total += i
    i += 1
```

```
print("Sum of 10 consecutive numbers:", total)
```

OUTPUT:

Sum of 10 consecutive numbers: 55

22-08-2025

#1.Program using operators:

```
x = ["Rose", "Lotus"]
print('Is value Present?', "Rose" in x)
print('Is value not Present?', "Riya" not in x)
a = ["Rose", "Lotus"]
b = ["Rose", "Lotus"]
c = a
print(a is c)
print(a is not c)
print(a is b)
print(a is not b)
print(a == b)
print(a != b)
```

OUTPUT:

Is value Present? True
Is value not Present? True
True
False
False
True
True
False

 # 3. Program Using Conditional Statements

```
a = int(input("Enter a : "))
b = int(input("Enter b : "))
c = int(input("Enter c : "))

if a > b and a > c:
```

```
    print(a, "is bigger from given numbers")
elif b > c and b > a:
    print(b, "is bigger from given numbers")
else:
    print(c, "is bigger from given numbers")
```

OUTPUT:

```
 Enter a : 10
 Enter b : 20
 Enter c : 30
30 is bigger from given numbers
```

#4.Program Using Loops

```
def print_pattern(height):
    for i in range(height):
        for j in range(height):
            if j == 0 or i == height - 1:
                print("*", end="")
            else:
                print(" ", end="")
        print()

height = 5
print_pattern(height)
```

OUTPUT:

```
 *
 *
 *
 *
 *****
```

#5.Program to Compute Weight in Pounds

```
weight = float(input("Enter your weight : "))
weight = weight * 2.20462
print(f"Weight in Pounds : {weight}")
```

OUTPUT:

Enter your weight : 75
Weight in Pounds : 165.3465

#6. Program to Compute GCD of Three Numbers

```
import math
e1, e2, e3 = input("Enter 3 Numbers = ").split(',')
e1 = int(e1)
e2 = int(e2)
e3 = int(e3)
result = math.gcd(e1, e2, e3)
print(f"GCD of {e1}, {e2}, {e3} = {result}")
```

OUTPUT:

Enter 3 Numbers = 10,20,30
GCD of 10, 20, 30 = 10

#7. Program for Factorial of a Number

```
number = int(input("Enter the integer = "))
factorial = 1
if number < 0:
   print("Factorial does not exist")
elif number == 0:
   print("Factorial of 0 is 1")
else:
   for i in range(1, number + 1):
      factorial = factorial * i
   print(f"factorial of {number} = {factorial}")
```

OUTPUT:

Enter the integer = 25
factorial of 25 = 15511210043330985984000000

#8. Program to Find Maximum Number in a List

```
list1 = [100, 200, 300, 400, 500]
max_number = list1[0]
for i in list1:
```

```
    if max_number < i:
        max_number = i
print(list1)
print(f"Max Number in List : {max_number}")
```

OUTPUT:

```
[100, 200, 300, 400, 500]
 Max Number in List : 500
```

#9. Program to Create and Print Lists

```
list1 = [1, 2, 3, 4, 5]
print("Printing List 1")
print("--------------")
for i in range(0, 5):
    print(f"Element present at index {i} = {list1[i]}")
print()
list1 = ["SaiTeja", "Charan", "Achyuth", "Anirudh", "Puneeth"]
print("Printing List 2")
print("--------------")
for i in range(0, 5):
    print(f"Element present at index {i} = {list1[i]}")
print()
list1 = ["Sai", 1, 2.3, "Saiteja", 2 * 3]
print("Printing List 3")
print("--------------")
for i in range(0, 5):
    print(f"Element present at index {i} = {list1[i]}")
```

OUTPUT:

```
Printing List 1
--------------
Element present at index 0 = 1
Element present at index 1 = 2
Element present at index 2 = 3
Element present at index 3 = 4
Element present at index 4 = 5
```

Printing List 2

Element present at index 0 = SaiTeja
Element present at index 1 = Charan
Element present at index 2 = Achyuth
Element present at index 3 = Anirudh
Element present at index 4 = Puneeth

Printing List 3

Element present at index 0 = Sai
Element present at index 1 = 1
Element present at index 2 = 2.3
Element present at index 3 = Saiteja
Element present at index 4 = 6

#10. Program for Strings

```
fname = "Sai Teja "
lname = "Karanam"
print("String Concatenation :", fname + lname)
print("fname * 4 :", fname * 4)
print("s is present in fname :", 's' in fname)
print("S is present in fname :", 'S' in fname)
print("Slicing of fname :", fname[2:7])
print("s is not present in lname:", 's' not in lname)
```

OUTPUT:

String Concatenation : Sai Teja Karanam
fname * 4 : Sai Teja Sai Teja Sai Teja Sai Teja
s is present in fname : False
S is present in fname : True
Slicing of fname : i Tej
s is not present in lname: True

26-08-2025

#1.Budget Allocation Program

```
total_budget = float(input("Total budget: "))
marketing_ratio = float(input("Marketing Ratio: "))
development_ratio = float(input("Development Ratio: "))
operations_ratio = float(input("Operations Ratio: "))
total_ratio = marketing_ratio + development_ratio + operations_ratio
marketing_budget = total_budget * (marketing_ratio / total_ratio)
development_budget = total_budget * (development_ratio / total_ratio)
operations_budget = total_budget * (operations_ratio / total_ratio)
print(f"Marketing budget: {marketing_budget:.2f}")
print(f"Development budget: {development_budget:.2f}")
print(f"Operations budget: {operations_budget:.2f}")
```

OUTPUT:

```
Total budget: 10000
Marketing Ratio: 3
Development Ratio: 5
Operations Ratio: 2
Marketing budget: 3000.00
Development budget: 5000.00
Operations budget: 2000.00
```

#2. Social Media Reach Program

```
followers = int(input("Followers: "))
followers_per_person = int(input("Followers per Person: "))
growth_percent = float(input("Growth Percentage: "))
estimated_reach = followers * followers_per_person * (1 + growth_percent / 100)
print(f"Total Estimated Reach = {int(estimated_reach)}")
```

OUTPUT:

```
Followers: 500
Followers per Person: 200
```

Growth Percentage: 10
Total Estimated Reach = 10000

#3. Total Stars Counted Program

```
nights = int(input("Nights: "))
total_stars = 0
for i in range(1, nights + 1):
    stars = int(input(f"Stars counted on night {i}: "))
    total_stars += stars
print(f"Total stars counted: {total_stars}")
```

OUTPUT:
Nights: 4
Stars counted each night: 20, 35, 25, 40
Total stars counted: 120

#4.Rectangle Pattern Printing Program

```
height = int(input("Height: "))
width = int(input("Width: "))
for i in range(height):
    print("#" * width)
print()
for i in range(width):
    print("#" * height)
```

OUTPUT:
Output

Height: 5
Width: 3
###
###
###
###
###

###
###
###

#5.Temperature Analysis Program

```
temperatures = [32, 34, 29, 31, 28, 33, 35, 36, 30, 28, 31, 32, 34, 35, 29, 30]
avg_temp = sum(temperatures) / len(temperatures)
days_above_avg = sum(1 for t in temperatures if t > avg_temp)
temp_difference = max(temperatures) - min(temperatures)
print(f"Average Temperature = {avg_temp:.1f}")
print(f"Days Above Average = {days_above_avg}")
print(f"Temp Difference = {temp_difference}")
```

OUTPUT:

Temperatures = [32, 34, 29, 31, 28, 33, 35, 36, 30, 28, 31, 32, 34, 35, 29, 30]
Average Temperature = 31.5
Days Above Average = 8
Temp Difference = 8

08-09-2025

List Operations and Methods

```
my_list = ['p','r','o','g','r','a','m','m','e']
print(my_list[2:5])
print(my_list[5:])
print(my_list[:])

list1 = [10, 15, 11, 65, 30]
list1.insert(1,80)
print('INSERTED ELEMENT IN THE LIST IS :', list1)
list1.sort()
print('SORTED ELEMENT IN THE LIST IS :', list1)
list1.reverse()
removed = list1.pop(3)
print('REVERSED ELEMENT IN THE LIST IS :', list1)
```

```
print('REMOVED ELEMENT IN THE LIST IS :', removed)
print('LARGEST ELEMENT IN THE LIST IS :',max(list1))
print('SMALLEST ELEMENT IN THE LIST IS :', min(list1))
```

OUTPUT:

```
['o', 'g', 'r']
['a', 'm', 'm', 'e']
['p', 'r', 'o', 'g', 'r', 'a', 'm', 'm', 'e']
INSERTED ELEMENT IN THE LIST IS : [10, 80, 15, 11, 65, 30]
SORTED ELEMENT IN THE LIST IS : [10, 11, 15, 30, 65, 80]
REVERSED ELEMENT IN THE LIST IS : [80, 65, 30, 15, 11, 10]
REMOVED ELEMENT IN THE LIST IS : 15
LARGEST ELEMENT IN THE LIST IS : 80
SMALLEST ELEMENT IN THE LIST IS : 10
```

LIST COMPREHENSION

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
newlist = [x for x in fruits if "a" in x]
print(newlist)
```

OUTPUT:

```
['apple', 'banana', 'mango']
```

Price with Tax Program

```
prices = [1.09, 23.56, 57.84, 4.56, 6.78]
TAX_RATE = .08
def get_price_with_tax(price):
    return price * (1 + TAX_RATE)
final_prices = list(map(get_price_with_tax, prices))
print(final_prices)
```

OUTPUT:

```
[1.1772, 25.4448, 62.4672, 4.9248, 7.3224]
```

Frequency of Elements in a List

```
l = [13,4,5,3,4]
```

```
l1 = []
l2 = []
print("Element \t \t frequency")
for i in l:
    if i not in l2:
        x = l.count(i)
        l1.append(x)
        l2.append(i)
for i in range(len(l1)):
    print(l2[i], "\t \t \t", l1[i])
```

OUTPUT:

Element	frequency
13	1
4	2
5	1
3	1

Tuple Operations and Methods

```
tuple1 = (1, 'a', 2, 'b')
tuple2 = (2, 'c', 4, 'd')
print(type(tuple1))
print(len(tuple1))
print(tuple1 + tuple2)
print(tuple2 * 3)
```

OUTPUT:

```
<class 'tuple'>
4
(1, 'a', 2, 'b', 2, 'c', 4, 'd')
(2, 'c', 4, 'd', 2, 'c', 4, 'd', 2, 'c', 4, 'd')
```

Create Tuple Dynamically

```
t = tuple()
n = int(input("How many values you want to enter: "))
```

```
for i in range(n):
    a = input("Enter Number: ")
    t = t + (a,)
print("Entered Numbers are: ")
print(t)
```

OUTPUT:

```
How many values you want to enter: 3
Enter Number: 5
Enter Number: 6
Enter Number: 7
Entered Numbers are:
('5', '6', '7')
```

Dictionary Operations

```
country_capitals =
{
    "United States": "Washington D.C.",
    "England": "London",
    "Germany": "Berlin",
    "Canada": "Ottawa"
}
print(len(country_capitals))
print(country_capitals)
country_capitals["Italy"] = "Rome"
print(country_capitals)
del country_capitals["Germany"]
print(country_capitals)
country_capitals.clear()
print(country_capitals)
```

OUTPUT:

```
4
{'United States': 'Washington D.C.', 'England': 'London', 'Germany': 'Berlin', 'Canada':
'Ottawa'}
{'United States': 'Washington D.C.', 'England': 'London', 'Germany': 'Berlin', 'Canada':
'Ottawa', 'Italy': 'Rome'}
```

{'United States': 'Washington D.C.', 'England': 'London', 'Canada': 'Ottawa', 'Italy': 'Rome'}
{}

String Operations and Methods

```
a = "Hello, World!"
print(a[2:5])
print(a[:5])
print(a[2:])
print(a[-5:-2])
print(a.upper())
print(a.lower())
print(a.strip())
print(a.replace("H", "J"))
print(a.split(","))
print(len(a))
a = "Hello"
b = "World"
c = b + a
print(c)
c = b + "\t " + a
print(c)
```

OUTPUT:

```
llo
Hello
llo, World!
orl
HELLO, WORLD!
hello, world!
Hello, World!
Jello, World!
['Hello', ' World!']
13
WorldHello
World	 Hello
```

String Advanced Methods

```
str1 = "Hello"
str2 = " Hello"
print(str1 is str2)
print(str1 is not str2)
print(str1*3)
print(str1+str2)
print(str1[4])
print(str1[2:4])
print('w' in str1)
print('wo' not in str2)
print(r'Hello\n world')
print("The string str1 : %s" % (str1))
```

OUTPUT:

```
False
True
HelloHelloHello
Hello Hello
o
ll
False
False
Hello\n world
The string str1 : Hello
```

String split() and find() Example

```
str1 = "Java is a programming language"
str2 = str1.split()
print(str1)
print(str2)
str2 = str1.split(sep=',')
print(str1)
print(str2)
str1 = "python is a programming language"
str2 = str1.find("is")
```

```
str3 = str1.find("java")
str4 = str1.find("n",5)
str5 = str1.find("i",5,25)
print(str2, str3, str4, str5)
```

OUTPUT:

```
Java is a programming language
['Java', 'is', 'a', 'programming', 'language']
Java is a programming language
['Java is a programming language']
7 -1 11 8
```

Consonants in Sentence

```
sentence = "The rocket, who was named Ted, came back from Mars because he missed his friends."
def is_consonant(letter):
    vowels = "aeiou"
    return letter.isalpha() and letter.lower() not in vowels
print([char for char in sentence if is_consonant(char)])
```

OUTPUT:

```
['T', 'h', 'r', 'c', 'k', 't', 'w', 'h', 'w', 's', 'n', 'm', 'd', 'T', 'd', 'c', 'm', 'b', 'c', 'k', 'f', 'r', 'm', 'M', 'r', 's', 'b', 'c', 's', 'h', 'm', 's', 's', 'd', 'h', 's', 'f', 'r', 'n', 'd', 's']
```

CASE STUDY 1 – STUDENT MANAGEMENT SYSTEM

PROGRAM (Python):

```
# Using a dictionary to store student data, where keys are student IDs
# and values are dictionaries containing name, age, and a list of courses.

students = {
    "S001": {"name": "Alice",   "age": 20, "courses": ["Math", "Physics"]},
    "S002": {"name": "Bob",     "age": 21, "courses": ["Chemistry", "Biology"]},
    "S003": {"name": "Charlie", "age": 19, "courses": ["Math", "Computer Science"]}
}
```

```
def add_student(student_id, name, age, courses):
    """Adds a new student to the system."""
    students[student_id] = {"name": name, "age": age, "courses": courses}
    print(f"Student {name} added.")

def enroll_course(student_id, course_name):
    """Enrolls a student in a new course."""
    if student_id in students:
        students[student_id]["courses"].append(course_name)
        print(f"Student {students[student_id]['name']} enrolled in {course_name}.")
    else:
        print("Student not found.")

def get_student_info(student_id):
    """Retrieves and prints a student's information."""
    if student_id in students:
        info = students[student_id]
        print(f"ID: {student_id}, Name: {info['name']}, Age: {info['age']}, "
              f"Courses: {', '.join(info['courses'])}")
    else:
        print("Student not found.")

# Example usage:
add_student("S004", "David", 22, ["History"])
enroll_course("S001", "Chemistry")
get_student_info("S001")
```

EXAMPLE INPUT & OUTPUT (console):

```
Student David added.
Student Alice enrolled in Chemistry.
ID: S001, Name: Alice, Age: 20, Courses: Math, Physics, Chemistry
```

CASE STUDY 2 – INVENTORY MANAGEMENT SYSTEM

PROGRAM (Python):

```
# Using a dictionary where keys are product names
# and values are tuples containing (price, quantity).

inventory = {
    "Laptop":   (1200, 10),
    "Mouse":    (25,   50),
    "Keyboard": (75,   20)
}

def add_product(product_name, price, quantity):
    """Adds a new product to the inventory."""
    inventory[product_name] = (price, quantity)
    print(f"Product '{product_name}' added.")

def update_quantity(product_name, new_quantity):
    """Updates the quantity of an existing product."""
    if product_name in inventory:
        price, _ = inventory[product_name]  # Get current price
        inventory[product_name] = (price, new_quantity)
        print(f"Quantity of '{product_name}' updated to {new_quantity}.")
    else:
        print("Product not found.")

def list_products():
    """Lists all products and their details."""
    print("Current Inventory:")
    for product, details in inventory.items():
        price, quantity = details
        print(f"- {product}: Price=${price}, Quantity={quantity}")

# Example usage:
add_product("Monitor", 300, 15)
update_quantity("Laptop", 8)
```

```
list_products()
```

EXAMPLE INPUT & OUTPUT (console):

```
Product 'Monitor' added.
Quantity of 'Laptop' updated to 8.
Current Inventory:
- Laptop: Price=$1200, Quantity=8
- Mouse: Price=$25, Quantity=50
- Keyboard: Price=$75, Quantity=20
- Monitor: Price=$300, Quantity=15
```

CASE STUDY 3 – UNIQUE USER TAGS MANAGEMENT

PROGRAM (Python):

```
# Using a dictionary where keys are user IDs and values are sets of tags.

user_tags = {
    "U001": {"admin", "developer", "tester"},
    "U002": {"developer", "frontend"},
    "U003": {"tester"}
}

def add_tag_to_user(user_id, tag):
    """Adds a tag to a user's set of tags."""
    if user_id in user_tags:
        user_tags[user_id].add(tag)
        print(f"Tag '{tag}' added to user '{user_id}'.")
    else:
        print("User not found.")

def check_user_has_tag(user_id, tag):
    """Checks if a user has a specific tag."""
    if user_id in user_tags:
        if tag in user_tags[user_id]:
```

```
            print(f"User '{user_id}' HAS tag '{tag}'.")
            return True
        else:
            print(f"User '{user_id}' DOES NOT have tag '{tag}'.")
            return False
    else:
        print("User not found.")
        return False

# Example usage:
add_tag_to_user("U001", "backend")
check_user_has_tag("U002", "developer")
check_user_has_tag("U003", "admin")
```

EXAMPLE INPUT & OUTPUT (console):

```
Tag 'backend' added to user 'U001'.
User 'U002' HAS tag 'developer'.
User 'U003' DOES NOT have tag 'admin'.
```

STRING BASICS – CREATION, CONVERSION, INDEXING

```
# Creating a string
s = 'Good Morning'
print(s)
# Conversion: int to str
a = 99   # int
print(str(a))   # int to str
# Output:
# Good Morning
# 99
```

INDEXING (POSITIVE INDEX)

```
s = 'GoodMorning'
print(s[0])            # First character
print(s[2])            # Third character
print(s[4])
```

```
print(s[len(s) - 1])     # Last character
print(len(s))            # Length of string
# Example Output:
# G
# o
# M
# g
# 11

# QUESTION:
# What happens if we access s[11]?

# Because the valid indices are 0 to len(s)-1, i.e., 0 to 10,
# s[11] raises:
# IndexError: string index out of range

NEGATIVE INDEXING
s = 'GoodMorning'
print(s[-1])   # Last character
print(s[-2])   # Second last character

# Output:
# g
# n

SLICING
s = 'Good Morning'
print(s[1:4])     # from index 1 to 3
print(s[1:])      # from index 1 to end
print(s[:4])      # from start to index 3
print(s[:])       # whole string
print(s[0:8:3])   # from 0 to 7, step 3
print(s[::-1])    # reverse string

# Output:
# ood
# ood Morning
```

```
# Good
# Good Morning
# GdM
# gninroM dooG

EDITING AND DELETING – STRING IMMUTABILITY
s = 'GoodMorning'
s[0] = "M"
# This will cause an error:
# TypeError: 'str' object does not support item assignment
# Reason: Strings in Python are IMMUTABLE (cannot change in-place).
# Correct way: create a new string
s = 'GoodMorning'
a = "M" + s[1:]
print(a)

# Output:
# MoodMorning


# Deleting parts of a string like this DOES NOT work:
s = 'GoodMorning'
# del s[0]       # ❌ Error: 'str' object doesn't support item deletion
# del s[:3:2]    # ❌ Error

# Only deleting the whole variable is allowed:
s = 'GoodMorning'
del s
# print(s)   # NameError: name 's' is not defined

CONCATENATION & MULTIPLICATION
# Concatenation
s = "Good" + "-" + "Morning"
print(s)

# Output:
# Good-Morning
```

```
# Multiplication
s = "Good"
print(s * 3)

# Output:
# GoodGoodGood

RELATIONAL OPERATIONS ON STRINGS
print("Hello" == "World")   # False
print("Hello" != "World")   # True
print("Mumbai" > "Pune")    # Comparison based on Unicode / lexicographic
print("Goa" < "Indore")

# Output (depends on lexicographic order):
# False
# True
# False
# True

LOGICAL OPERATIONS WITH STRINGS
print("Hello" and "World")   # both True → returns last truthy: "World"
print("" and "World")       # first False → ""
print("" or "World")        # False or True → "World"
print(not "Hello")          # non-empty → True → not True = False
print(not "")               # empty string → False → not False = True

# Output:
# World
#
# World
# False
# True

LOOPS WITH STRINGS

s = 'GoodMorning'
```

```
for ch in s:
    print(ch)

# Output (each char on new line):
# G
# o
# o
# d
# M
# o
# r
# n
# i
# n
# g
```

MEMBERSHIP OPERATORS

```
s = 'GoodMorning'
if 'M' in s:
    print('true')

# Output:
# true
```

COMMON STRING FUNCTIONS

```
s = 'GoodMorning'
print(len(s))              # length
print(min(s))              # smallest character (by Unicode)
print(max(s))              # largest character
print(sorted(s))           # sorted characters in a list
print(sorted(s, reverse=True))

# Example Output:
# 11
# G
# r
# ['G', 'M', 'd', 'g', 'i', 'n', 'n', 'o', 'o', 'o', 'r']
```

```
# ['r', 'o', 'o', 'o', 'n', 'n', 'i', 'g', 'd', 'M', 'G']
TITLE / CAPITALIZE
s = "its raining outside"
print(s.capitalize())   # first char uppercase
print(s.title())        # each word capitalized

# Output:
# Its raining outside
# Its Raining Outside

UPPER / LOWER / SWAPCASE
s = "Its Raining Outside"
print(s.upper())
print(s.lower())
print(s.swapcase())

# Output:
# ITS RAINING OUTSIDE
# its raining outside
# iTS rAINING oUTSIDE

COUNT
s = "Its Raining Outside"
print(s.count("i"))      # number of 'i'
print(s.count("ing"))    # occurrence of substring "ing"

# Output:
# 3
# 1
FIND / INDEX
s = "Its Raining Outside"
print(s.find("ing"))     # index of first occurrence
print(s.index("ing"))    # same as find, but error if not found
print(s.find("down"))    # -1
# print(s.index("down")) # ValueError

# Output:
```

```
# 8
# 8
# -1
# (index("down") would raise: ValueError: substring not found)

EXERCISE 1 – REVERSE A STRING
# Method 1: Slicing
s = input("Enter a string: ")
rev = s[::-1]
print("Reversed string:", rev)

# Sample I/O:
# Enter a string: Hello
# Reversed string: olleH

EXERCISE 2 – REMOVE SPECIAL SYMBOLS / PUNCTUATION
import string
s = input("Enter a string: ")
cleaned = ""
for ch in s:
    if ch.isalnum() or ch.isspace():   # keep letters, digits, spaces
        cleaned += ch
print("Cleaned string:", cleaned)

# Example:
# Input: Hello!!! How are you??? 123...
# Output: Cleaned string: Hello How are you 123

EXERCISE 3 – FIND WORDS WITH BOTH LETTERS AND NUMBERS
s = input("Enter a sentence: ")
words = s.split()
print("Words with both letters and digits:")
for w in words:
    has_alpha = any(ch.isalpha() for ch in w)
    has_digit = any(ch.isdigit() for ch in w)
    if has_alpha and has_digit:
        print(w)
```

```
# Example:
# Input: abcd 123 ab12 cd34x y7z 99
# Output:
# ab12
# cd34x
# y7z
```

EXERCISE 4 – FIND ALL OCCURRENCES OF SUBSTRING (CASE-INSENSITIVE)

```
s = input("Enter main string: ")
sub = input("Enter substring to search: ")
s_lower = s.lower()
sub_lower = sub.lower()
start = 0
positions = []
while True:
    idx = s_lower.find(sub_lower, start)
    if idx == -1:
        break
    positions.append(idx)
    start = idx + 1
print("Occurrences at indices:", positions)

# Example:
# Enter main string: Banana BANANA baNaNa
# Enter substring to search: ana
# Occurrences at indices: [1, 3, 8, 10, 15, 17]
```

EXERCISE 5 – COUNT LETTERS, DIGITS, SPECIAL SYMBOLS

```
s = input("Enter a string: ")
letters = digits = specials = 0
for ch in s:
    if ch.isalpha():
        letters += 1
    elif ch.isdigit():
        digits += 1
```

```
    else:
        specials += 1
print("Letters:", letters)
print("Digits:", digits)
print("Special symbols:", specials)

# Example:
# Input: Abc123$#!
# Output:
# Letters: 3
# Digits: 3
# Special symbols: 3
```

WORD FREQUENCY USING STRING SPLIT

```
str1 = 'apple mango apple orange orange apple guava mango mango'
# Split into list of words
words = str1.split()
unique_words = []
for w in words:
    if w not in unique_words:
        unique_words.append(w)
for w in unique_words:
    print('Frequency of', w, 'is :', words.count(w))

# Output:
# Frequency of apple is : 3
# Frequency of mango is : 3
# Frequency of orange is : 2
# Frequency of guava is : 1
```

PASSWORD VALIDATION – COMPLEX CONDITIONS

```
# Conditions:
# - Minimum 8 characters
# - At least 1 lowercase letter
# - At least 1 uppercase letter
# - At least 1 digit
```

```
# - At least 1 symbol from [_ or @ or $]
# - All characters must be from allowed sets (no other symbols)
s = input("Enter password: ")
l = u = p = d = 0
if len(s) >= 8:
    for ch in s:
        if ch.islower():
            l += 1        # lowercase
        elif ch.isupper():
            u += 1        # uppercase
        elif ch.isdigit():
            d += 1        # digit
        elif ch in ['@', '$', '_']:
            p += 1        # allowed special symbol
        else:
            # invalid symbol not in allowed set - count as special
            p += 0
if (l >= 1 and u >= 1 and p >= 1 and d >= 1 and l + u + d + p == len(s)):
    print("Valid Password")
else:
    if u == 0:
        print("upper_case is missing")
    if l == 0:
        print("lower_case is missing")
    if d == 0:
        print("digit is missing")
    if p == 0:
        print("symbol is missing")
    print("Invalid Password")

# Example:
# Input: R@m@_f0rtu9e$
# Output: Valid Password
#
# Input: Rama_fortune$
# Output:
# digit is missing
```

```
# Invalid Password
#
# Input: Rama#fortu9e
# Output:
# symbol is missing
# Invalid Password
```

REMOVING UNWANTED CHARACTERS & WHITESPACE

```
text = "   Hello World!   "
cleaned = text.strip()
print(cleaned)

# Output:
# Hello World!
```

SEARCHING & REPLACING STRINGS

```
text = "Hello world!"
updated = text.replace("world", "everyone")
print(updated)

# Output:
# Hello everyone!

# Searching:
# text.find("world")  -> index or -1
# text.rfind("world") -> search from right
```

SPLITTING & JOINING STRINGS (BASIC & REGEX)

```
import re
text = "apple, orange; banana, grape"
fruits = re.split(r'[;, ]+', text)
print(fruits)

# Output:
# ['apple', 'orange', 'banana', 'grape']
```

```
fruits = ["apple", "orange", "banana"]
joined = ", ".join(fruits)
print(joined)

# Output:
# apple, orange, banana
```

13-10-2025

Random float between 0 and 100

```
import random
num = random.random()
num = num * 100
print(num)
```

OUTPUT:
37.52938493749182

Random integer between 0 and 9

```
import random
num = random.randint(0, 9)
print(num)
import random
num = random.randint(0, 9)  # includes both 0 and 9
print(num)
```

OUTPUT:
6

Divide two random integers

```
import random
num1 = random.randint(0, 1000)
num2 = random.randint(1, 1000)
newrand = num1 / num2
print(newrand)
```

OUTPUT:
2.0

Random number from 0 to 95, step 5

```
import random
num = random.randrange(0, 100, 5)
print(num)
```

OUTPUT:
45

Randomly choose a colour

```
import random
colour = random.choice(["red", "black", "green"])
print(colour)
```

OUTPUT:
black

HEADS OR TAILS QUESTION + PROGRAM

```
import random
value = random.choice(["h", "t"])
choice = input("Make your choice (h or t): ").lower()
if value == choice:
    print("You win")
else:
    print("Bad luck")
if value == "h":
    print("The computer selected heads")
else:
    print("The computer selected tails")
```

OUTPUT:

Make your choice (h or t): h
Bad luck

The computer selected tails

COLOUR GUESSING GAME QUESTION + PROGRAM

```
import random
colour = random.choice(["black", "white", "blue", "red", "yellow"])
correct = False
while correct == False:
    guess = input("Pick one colour: black, white, blue, red, yellow: ").lower()
     if guess == colour:
        print("Well done YOU WON THE GAME!")
        correct = True
    else:
        if colour == "black":
            print("You are such a BLACK-luck guy.")
        elif colour == "white":
            print("Too naive to choose WHITE.")
        elif colour == "blue":
            print("You are probably feeling BLUE right now.")
        elif colour == "red":
            print("Is your face RED right now?")
        elif colour == "yellow":
            print("I bet your future is as bright as YELLOW.")
```

OUTPUT:

```
Pick one colour: black, white, blue, red, yellow: red
You are probably feeling BLUE right now.
Pick one colour: black, white, blue, red, yellow: yellow
You are probably feeling BLUE right now.
Pick one colour: black, white, blue, red, yellow: blue
Well done YOU WON THE GAME!
```

COLLECTION MODULE PROGRAMS

```
from collections import Counter
counts = Counter(['a', 'b', 'a', 'c', 'b'])
print(counts)
```

OUTPUT:
Counter({'a': 2, 'b': 2, 'c': 1})

```
from collections import deque
d = deque([1, 2, 3])
print(d)
d.appendleft(0)
print(d)
d.pop()
print(d)
```

OUTPUT:
deque([1, 2, 3])

```
from collections import namedtuple
Point = namedtuple('Point', ['x', 'y'])
p = Point(10, 20)

print(p.x)
print(p.y)
```

OUTPUT:
10
20

```
from collections import defaultdict
s = [
   ('yellow', 1),
   ('blue', 2),
   ('yellow', 3),
   ('blue', 4),
   ('red', 1)
]
d = defaultdict(list)
for k, v in s:
   d[k].append(v)
```

```
print(d)
```

OUTPUT:

```
defaultdict(<class 'list'>, {
   'yellow': [1, 3],
   'blue': [2, 4],
   'red': [1]
})
```

FILE HANDLING:

14-10-2025

1.Reading a file(r mode)

```
file_object = open("my_data.txt", "r")
print("Content of 'my_data.txt':")
file_content = file_object.read()
print(file_content)
file_object.close()
```

OUTPUT:

```
Hello world #if the file name contains
Python file handling
```

Using with open()

```
with open("my_data.txt", "r") as file_object:
   print("Content of 'my_data.txt':")
```

```
    file_content = file_object.read()
    print(file_content)
```

OUTPUT:
Content of 'my_data.txt':
Hello world
Python file handling

2.Writing to a file(w mode)

```
# Open the file in write mode ('w').
file_object = open("my_data.txt", "w")
print("Enter lines to write to the file. Type 'exit' to stop.")
while True:
    line = input("Enter line: ")
    if line.lower() == "exit":
        break
    file_object.write(line + "\n")  # Add newline after each line
file_object.close()
print("Data written to 'my_data.txt' successfully.")
```

If user enters:
Enter line: Apple
Enter line: Banana
Enter line: exit

OUTPUT:
Enter lines to write to the file. Type 'exit' to stop.
Data written to 'my_data.txt' successfully.

This file will contain:
Apple
Banana

3.Appending to a file(a mode)

```
# Open the file in append mode ('a').
file_object = open("my_data.txt", "a")

print("Enter lines to append to the file. Type 'exit' to stop.")
```

```
while True:
    line = input("Enter line to append: ")
    if line.lower() == "exit":
        break
    file_object.write(line + "\n")

file_object.close()
print("Data appended to 'my_data.txt' successfully.")
```

If user enters:
Enter line to append: Mango
Enter line to append: Orange
Enter line to append: exit

OUTPUT:
Enter lines to append to the file. Type 'exit' to stop.
Data appended to 'my_data.txt' successfully.

And file becomes:
Apple
Banana
Mango
Orange

4.Write given sentences to Demo1.txt

```
def main():
    obj1 = open("Demo1.txt", "w")  # Opens file in write mode
    obj1.write("Hello, How are you?\n")
    obj1.write("Welcome to The chapter File Handling.\n")
    obj1.write("Enjoy the session.\n")
    obj1.close()

main()  # Call to main function
```

File contains:
Hello, How are you?

Welcome to The chapter File Handling.
Enjoy the session.

5.Writing numbers to a file

```
def main():
    obj1 = open("WriteNumbers.txt", "w")  # Open file in write mode
    for x in range(1, 21):  # 1 to 20 inclusive
        obj1.write(str(x))   # Convert number to string
        obj1.write(" ")      # Add space between numbers
    obj1.close()

main()
```

File contains:
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

6.Generate 50 random numbers(500-1000)

```
from random import randint  # Import randint

fp1 = open("WriteNumRandom.txt", "w")  # Open file in write mode
for _ in range(50):  # 50 iterations
    x = randint(500, 1000)  # Generate one random number
    fp1.write(str(x) + " ")  # Convert to string and write with space
fp1.close()  # Close the file
```

OUTPUT:
734 680 912 501 998 600 743 954 820 503 ... (total 50 numbers)

7.Reading text from a file:

Assume ReadDemo1.txt contains:
Python programming
Language
Welcome to
programming language

```
a.read()
fp = open("ReadDemo1.txt", "r")  # Open in read mode
```

```
text = fp.read()                # Read whole file as one string
print(text)
fp.close()
```

Given file contains:
Python programming
Language
Welcome to
programming language

Output:
Python programming
Language
Welcome to
programming language

```
b.readline()
fp = open("ReadDemo1.txt", "r")
for line in fp:
    print(line, end="")  # end="" to avoid extra blank lines
fp.close()
```

OUTPUT:
Python programming

Language

Welcome to

programming language

8.Read continents of numbers.txt

```
fp1 = open("numbers.txt", "r")
num = fp1.read()      # Entire contents as a single string
print(num)
print(type(num))      # <class 'str'>
```

```
fp1.close()
```

If file contains:
1 2 3 4 5

OUTPUT:
1 2 3 4 5
<class 'str'>

9.Sum numbers from numbers.txt
If contents of numbers.txt:

5
2
4
6
8
10

```
fp1 = open("numbers.txt", "r")

num = int(fp1.readline())   # first line: how many numbers
print(num)
total = 0
print("The", num, "numbers present in the file are as follows:")
for i in range(num):
    num1 = int(fp1.readline())  # read next number
    print(num1)
    total += num1
print("Sum of all the numbers (except first):")
print(total)
fp1.close()
```

OUTPUT:
5
The 5 numbers present in the file are as follows:
2
4

6
8
10
Sum of all the numbers (except first):
30

10.Write and read from same file:

```
# Writing part
f = open("new.txt", "w")
while True:
    st = input("Enter next line: ")
    if st == "END":
        break
    f.write(st + "\n")
f.close()
# Reading part
f = open("new.txt", "r")
while True:
    st = f.readline()
    if not st:   # end of file
        break
    if st[0].isupper():
        print(st, end="")  # end="" to avoid extra blank line
f.close()
```

User enters:
Enter next line:Hello world
Enter next line:welcome to
Enter next line:Python programming
Enter next line:END

OUTPUT:
Hello world
Python programming

17-10-2025

1) PALINDROME CHECK + DIGIT OCCURRENCE COUNT

PROGRAM:

```
from collections import Counter

value = input("Enter a value : ")
if value == value[::-1]:
    print("Palindrome")
else:
    print("Not Palindrome")

counted_dict = Counter(value)
for key in sorted(counted_dict.keys()):
    print(f'{key} appears {counted_dict[key]} times')
```

INPUT & OUTPUT:

```
Enter a value : 1221
Palindrome
1 appears 2 times
2 appears 2 times
```

2) STRING SIMILARITY BETWEEN TWO STRINGS

PROGRAM:

```
from difflib import SequenceMatcher
str1 = input("Enter String 1 : ")
str2 = input("Enter String 2 : ")
sim = SequenceMatcher(None, str1, str2).ratio()
print("Similarity between strings \"" + str1 + "\" and \"" + str2 + "\" is : ", sim)
```

INPUT & OUTPUT:

```
Enter String 1 : apple
Enter String 2 : apply
Similarity between strings "apple" and "apply" is :  0.8
```

3) DISPLAY FIRST N LINES + COUNT WORD OCCURRENCES

PROGRAM:

```
import os.path
import sys
fname = input("Enter the filename : ")
if not os.path.isfile(fname):
    print("File", fname, "doesn't exists")
    sys.exit(0)
infile = open(fname, "r")
lineList = infile.readlines()
lineCount = int(input("Enter the number of lines you want to display : "))
for i in range(lineCount):
    print(i+1, ":", lineList[i], end="")
word = input("Enter a word : ")
cnt = 0
for line in lineList:
    cnt += line.count(word)
print("The word", word, "appears", cnt, "times in the file")
```

INPUT & OUTPUT:

```
Enter the filename : Wel.txt
Enter the number of lines you want to display : 1
1 : welcome to cse ffffgggggggg
Enter a word : cse
The word cse appears 1 times in the file
```

4) FILE READ METHODS (read, readline, readlines)

PROGRAM (read):

```
with open("example.txt", "r") as file:
      content = file.read()
    print(content)
```

INPUT & OUTPUT:

```
Line one
Line two
Line three
```

PROGRAM (readline):

```
with open("example.txt", "r") as file:
    line1 = file.readline()
    print(line1)
```

OUTPUT:

```
Line one
```

PROGRAM (readlines):

```
with open("example.txt", "r") as file:
    lines = file.readlines()
    for line in lines:
        print(line.strip())
```

OUTPUT:

```
Line one
Line two
Line three
```

PROGRAM (loop):

```
with open("example.txt", "r") as file:
    for line in file:
        print(line.strip())
```

OUTPUT:

```
Line one
Line two
Line three
```

5) FILE FUNCTIONS (read, readable, seek, tell)

PROGRAM (read 3 chars):

```
f = open("Wel.txt", "r")
print(f.read(3))
```

OUTPUT:

Wel

PROGRAM (read full file):

```
f = open("Wel.txt", "r")
print(f.read(-1))
```

OUTPUT:
Welcome to cse

PROGRAM (readable):

```
f = open("Wel.txt", "r")
print(f.readable())
```

OUTPUT:
True

PROGRAM (seek):

```
f = open("Wel.txt", "r")
f.seek(2)
print(f.readline())
```

OUTPUT:
lcome to cse

PROGRAM (seekable):

```
f = open("Wel.txt", "r")
print(f.seekable())
```

OUTPUT:
True

PROGRAM (tell):

```
f = open("Wel.txt", "r")
print(f.tell())
```

OUTPUT:
0

PROGRAM:
6.Write sentences to Demo1.txt

```
def main():
    obj1 = open("Demo1.txt", "w")
    obj1.write("Hello, How are You?\n")
    obj1.write("Welcome to The chapter File Handling.\n")
    obj1.write("Enjoy the session.\n")
    obj1.close()

main()
```

OUTPUT (in file):
Hello, How are You?
Welcome to The chapter File Handling.
Enjoy the session.

7) WRITE NUMBERS 1–20 TO FILE

PROGRAM:

```
def main():
    obj1 = open("WriteNumbers.txt", "w")
    for x in range(1, 21):
        obj1.write(str(x) + " ")
    obj1.close()

main()
```

OUTPUT (in file):
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

8) WRITE 50 RANDOM NUMBERS TO FILE

PROGRAM:

```
from random import randint

fp1 = open("WriteNumRandom.txt", "w")
for _ in range(50):
    x = randint(500, 1000)
    fp1.write(str(x) + " ")
fp1.close()
```

OUTPUT (in file):
723 611 955 843 502 999 ... (50 random numbers)

9) READ ReadDemo1.txt USING read()
PROGRAM:

```
fp = open("ReadDemo1.txt", "r")
text = fp.read()
print(text)
```

OUTPUT:
Python programming
Language
Welcome to
programming language

10) SUM OF NUMBERS FROM numbers.txt
PROGRAM:

```
fp1 = open("numbers.txt", "r")
num = int(fp1.readline())
print(num)
total = 0
print("The", num, "numbers present in the file are as follows:")
for i in range(num):
    num1 = int(fp1.readline())
    print(num1)
    total += num1
```

```
print("Sum of all the numbers (except first):")
print(total)
```

INPUT (numbers.txt):

```
5
2
4
6
8
10
```

OUTPUT:

```
5
The 5 numbers present in the file are as follows:
2
4
6
8
10
Sum of all the numbers (except first):
30
```

11) WRITE & DISPLAY ONLY UPPERCASE LINES

PROGRAM:

```
f = open("new.txt", "w")
while True:
    st = input("Enter next line:")
    if st == "END":
        break
    f.write(st + '\n')
f.close()

f = open("new.txt", "r")
while True:
    st = f.readline()
    if not st:
```

```
        break
    if st[0].isupper():
        print(st)
f.close()
```

INPUT & OUTPUT:

```
Enter next line:Hello world
Enter next line:welcome to
Enter next line:Python programming
Enter next line:END

Hello world
Python programming
```

12) SUBJECT MENU PROGRAM (1/2/3)

PROGRAM:

```
count = 0
while count < 5:
    print('1) Create a new file')
    print('2) Display the file')
    print('3) Add a new item to the file ')
    selection_input = int(input('Make a selection 1, 2 or 3 : '))
      if selection_input == 1:
        file = open('Subject.txt', 'w')
        subject_name = input('Enter school subject name : ')
        file.write(subject_name + '\n')
        file.close()
     elif selection_input == 2:
        file = open('Subject.txt', 'r')
        print(file.read())
        file.close()
     elif selection_input == 3:
        file = open('Subject.txt', 'a')
        subject_name = input('Enter school subject name to add into file : ')
        file.write(subject_name + '\n')
        file.close()
```

```
    else:
        print("Invalid choice!")
    count += 1
file = open('Subject.txt', 'r')
print(file.read())
file.close()
```

INPUT & OUTPUT (sample):
1) Create a new file
2) Display the file
3) Add a new item to the file
Make a selection 1, 2 or 3 : 1
Enter school subject name : Maths

1) Create a new file
2) Display the file
3) Add a new item to the file
Make a selection 1, 2 or 3 : 3
Enter school subject name to add into file : Physics

1) Display the file
2) Display the file
3) Add a new item
Make a selection 1, 2 or 3 : 2
Maths
Physics

Final Subject.txt contains:
Maths
Physics
Chemistry

24-10-2025

PROGRAM 1 – ZeroDivisionError

```
n= 10
try:
```

```
    res = n / 0
except ZeroDivisionError:
    print("Can't be divided by zero!")

# OUTPUT:
# Can't be divided by zero!
```

PROGRAM 2 – SyntaxError (using eval)

```
# Note: A normal SyntaxError happens BEFORE the program runs,
# so we cannot catch it around normal code.
# But we can demonstrate catching SyntaxError when using eval().

code = "print('Hello world'"   # Missing closing parenthesis

try:
    eval(code)
except SyntaxError:
    print("Verify the line: try section properly closed or not")

# OUTPUT:
# Verify the line: try section properly closed or not
```

PROGRAM 3 – ValueError and ZeroDivisionError

```
try:
    x = int("str")   # This will cause ValueError
    inv = 1 / x      # Inverse calculation
except ValueError:
    print("Not Valid! Give only integer input")
except ZeroDivisionError:
    print("Zero has no inverse!")

# OUTPUT:
# Not Valid! Give only integer input
```

PROGRAM 4 – Exception inside except (problematic version)

```
# Try this to understand error possibility:
# Give input: a
try:
    n = 0
    res = 100 / n
except ZeroDivisionError:
    print("You can't divide by zero!")
    n = int(input("enter value "))    # If you enter 'a' → ValueError (uncaught)
    res = 100 / n
    print(res)
except ValueError:
    print("Enter a valid number!")
else:
    print("Result is", res)
finally:
    print("Execution complete.")

# Example run (input = a) causes:
# You can't divide by zero!
# enter value a
# ValueError: invalid literal for int() with base 10: 'a'
# (Program crashes before reaching finally in some setups)
```

PROGRAM 4 – SOLUTION (Exception within exception handled)

```
try:
    n = 0
    res = 100 / n
except ZeroDivisionError:
    print("You can't divide by zero!")
    try:
        n = int(input("enter value "))
        res = 100 / n
        print(res)
    except ValueError:
        print("Enter only numbers except 0!")
        n = int(input("enter value "))
```

```
        res = 100 / n
        print(res)
else:
    print("Result is", res)
finally:
    print("Execution complete.")

# TYPE THE FOLLOWING AS OUTPUT (example):

# You can't divide by zero!
# enter value a
# Enter only numbers except 0 !
# enter value 7
# 14.285714285714286
# Execution complete.
```

PROGRAM 5.1 – Catching Multiple Exceptions (tuple)

```
a = ["10", "twenty", 30]  # Mixed list of integers and strings
try:
    total = int(a[0]) + int(a[1])  # 'twenty' cannot be converted to int
    print(total)
except (ValueError, TypeError) as e:
    print("Error", e)
except IndexError:
    print("Index out of range.")

# OUTPUT:
# Error invalid literal for int() with base 10: 'twenty'
```

PROGRAM 5.1 – ALTERNATE VERSION (expanded)

```
a = ["10", "twenty", 30]  # Mixed list of integers and strings
try:
    total = int(a[0]) + int(a[1])
    print(total)
except TypeError:
    print("ERROR DUE TO type")
except ValueError:
```

```
    print("ERROR DUE TO value error")
except IndexError:
    print("Index out of range.")

# OUTPUT:
# ERROR DUE TO value error
```

PROGRAM 6 – General Exception Handling Pattern

```
try:
    numerator = int(input("Enter a numerator: "))
    denominator = int(input("Enter a denominator: "))
    result = numerator / denominator
except ValueError:
    # Handles invalid input (e.g., non-integer)
    print("Invalid input. Please enter an integer.")
except ZeroDivisionError:
    # Handles division by zero
    print("Error: Cannot divide by zero.")
except Exception as e:
    # Catches any other unexpected exceptions
    print(f"An unexpected error occurred: {e}")
else:
    # Executed if no exception occurred in the try block
    print(f"The result of the division is: {result}")
finally:
    # Always executed, regardless of exceptions
    print("Program execution completed.")

# Example run 1:
# Enter a numerator: 10
# Enter a denominator: 2
# The result of the division is: 5.0
# Program execution completed.

# Example run 2:
# Enter a numerator: 10
# Enter a denominator: 0
```

```
# Error: Cannot divide by zero.
# Program execution completed.
```

PROGRAM 7 – Exception Using File Handling

```
# Run this program with a NON-EXISTING file name like "example.txt"
try:
    f = open("example.txt", "r")
    # Code to read from the file
except FileNotFoundError:
    print("File not found!")
else:
    # Executes only if no exception in try
    print(f.read())
    f.close()
finally:
    print("Finished file operation.")

# Example when file does NOT exist:
# File not found!
# Finished file operation.
```

PROGRAM 8 – Exception Handling in a Function

```
def divide(x, y):
    try:
        result = x / y
    except ZeroDivisionError:
        print("division by zero!")
    else:
        print("result is", result)
    finally:
        print("executing finally clause")

divide(12, 0)
divide("12", "2")  # No TypeError is caught here, just propagates

# OUTPUT:
# division by zero!
```

```
# executing finally clause
# Traceback (most recent call last):
#   ...
# TypeError: unsupported operand type(s) for /: 'str' and 'str'
```

PROGRAM 9.1 – No Exception Handling in Loop

```
def divide_num(num_list):
    for num in num_list:
        print(10 / num)
num_list = [5, 6, 0, 7]
divide_num(num_list)

# OUTPUT:
# 2.0
# 1.6666666666666667
# Traceback (most recent call last):
#   File "...", line 6, in <module>
#     divide_num(num_list)
#   File "...", line 3, in divide_num
#     print(10/num)
# ZeroDivisionError: division by zero
```

PROGRAM 9.2 – Exception Handling for Each Elemen

```
def divide_num(num_list):
    for num in num_list:
        try:
            print(10 / num)
        except ZeroDivisionError as error:
            print(error)
            print('Zero is not a valid argument here')
        else:
            print('in else block')

num_list = [5, 6, 0, 7]
divide_num(num_list)

# OUTPUT:
```

```
# 2.0
# in else block
# 1.6666666666666667
# in else block
# division by zero
# Zero is not a valid argument here
# 1.4285714285714286
# in else block
```

PROGRAM 10 – Multiple Exceptions with File + Division

```
def divide_num(num):
    try:
        f = open("testfile", "a")  # Open file in append mode
        r = 10 / num
        f.write("Result 10/%d is %f\n" % (num, r))
    except ZeroDivisionError as error:
        print(error)
        print('Zero is not a valid argument here')
    except FileNotFoundError as error:
        print(error)
        print('Passed file does not exist')
    finally:
        print('closing file')
        # f may not exist if open() failed, so check first:
        try:
            f.close()
        except NameError:
            pass

divide_num(0)

# OUTPUT:
# division by zero
# Zero is not a valid argument here
# closing file
```