

# Python Programming Lab

Manav Rachna International Institute of Research and  
Studies

School of Computer Applications

Submitted By	
Student Name	SNEHA TANWAR
Roll No	25/SCA/MCAN/068
Programme	Masters of Computer Applications
Semester	1 <sup>st</sup> Semester
Section	B
Department	School of Computer Applications
Batch	2025-27
Submitted To	
Faculty Name	Dr. Anupama Luthra



Manav Rachna Campus Rd, Gadakhor Basti Village, Sector 43, Faridabad, Haryana 121004

**1. Write a Python program to calculate number of days between two dates.**

```
from datetime import date  
  
d1 = date(2014, 7, 2) d2  
= date(2014, 7, 11)  
delta = d2 - d1  
print(f"{delta.days} days")
```

A screenshot of a code execution interface. On the left, there is a dark grey panel labeled "Output". Inside, the text "9 days" is displayed above the message "==== Code Execution Successful ===". In the top right corner of the "Output" panel, there is a small "Clear" button.

**2. Write a Python program that accepts an integer (n) and computes the value of n+nn+nnn**

```
n = input("Enter an integer: ")  
  
nn = n + n  
nnn = n + n + n  
  
result = int(n) + int(nn) + int(nnn)  
  
print("The result is:", result)
```

**Output** Clear

```
Enter an integer: 6
The result is: 738

==== Code Execution Successful ====
```

3. Ask the user for a number. Depending on whether the number is even or odd, print out an appropriate message to the user.

```
num = int(input("Enter a number: "))

if num % 2 == 0:
    print("The number is even.") else:
    print("The number is odd.")
```

**Output** Clear

```
Enter a number: 24
The number is even.

==== Code Execution Successful ====
```

4. Write a Python program which accepts a sequence of comma-separated numbers from user and generate a list and a tuple with those numbers.

```
numbers = input("Enter numbers separated by commas: ")

num_list = numbers.split(",")
num_tuple = tuple(num_list)

print("List:", num_list)
```

```
print("Tuple:", num_tuple)
```

**Output** Clear

```
Enter numbers separated by commas: 2,3,4,5
List: ['2', '3', '4', '5']
Tuple: ('2', '3', '4', '5')

==== Code Execution Successful ====
```

5. Write a Python program to calculate the sum of three given numbers, if the values are equal then return thrice of their sum.

```
a      = int(input("Enter first number: ")) b = int(input("Enter
second number: "))
c = int(input("Enter third number: "))

total = a + b + c

if a == b == c:
    total *= 3

print("The result is:", total)
```

**Output** Clear

```
Enter first number: 20
Enter second number: 47
Enter third number: 78
The result is: 145

==== Code Execution Successful ====
```

6. Write a Python program to test whether a passed letter is a vowel or not

```
letter = input("Enter a letter: ").lower()

if letter in 'aeiou':
    print("The letter is a vowel.") else:
    print("The letter is not a vowel.")
```

**Output**

Enter a letter: e  
The letter is a vowel.  
  
==== Code Execution Successful ===

**Clear**

7. Take a list, say for example this one: `a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]` and write a program that prints out all the elements of the list that are less than 5. Extras:
- a Instead of printing the elements one by one, make a new list that has all the elements less than 5 from this list in it and print out this new list.
  - b) Write this in one line of Python.
  - c) Ask the user for a number and return a list that contains only elements from the original list `a` that are smaller than that number given by the user

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
```

```
# Part 1: Print elements less than 5 one by one
print("Elements less than 5:")
for number in a:
    if number < 5:
        print(number)
```

```
# Part 2: Create a new list with elements less than 5 and print it
new_list = [number for number in a if number < 5]
print("New list with elements less than 5:", new_list)
```

```
# Part 3: Ask the user for a number and filter the list based on that n
n = int(input("Enter a number: "))
filtered_list = [number for number in a if number < n]
print(f"Elements in the list less than {n}: {filtered_list}")
```

**Output** Clear

```

Elements less than 5:
1
1
2
3
New list with elements less than 5: [1, 1, 2, 3]
Enter a number: 6
Elements in the list less than 6: [1, 1, 2, 3, 5]

==== Code Execution Successful ====

```

8. Create a program that asks the user for a number and then prints out a list of all the divisors of that number.

```

num = int(input("Enter a number: "))
divisors = []

for i in range(1, num + 1):
    if num % i == 0:
        divisors.append(i)

print("The divisors of", num, "are:", divisors)

```

**Output** Clear

```

Enter a number: 36
The divisors of 36 are: [1, 2, 3, 4, 6, 9, 12, 18, 36]

==== Code Execution Successful ====

```

9. Take two lists, say for example these two: **a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89] b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]** and write a program that returns a list that contains only the elements that are common between the lists (without duplicates). Make sure your program works on two lists of different sizes

```

a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
b = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]

```

```
common_elements = []
```

```
for element in a:  
    if element in b and element not in common_elements:  
        common_elements.append(element)  
  
print("Common elements:", common_elements)
```

Output Clear

```
Common elements: [1, 2, 3, 5, 8, 13]  
==== Code Execution Successful ====
```

- 10.** Ask the user for a string and print out whether this string is a palindrome or not.

```
text = input("Enter a string: ")  
  
if text == text[::-1]:  
    print("The string is a palindrome.") else:  
    print("The string is not a palindrome.")
```

Output Clear

```
Enter a string: abba  
The string is a palindrome.  
==== Code Execution Successful ====
```

- 11.** Let's say I give you a list saved in a variable: `a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]`. Write one line of Python that takes this list `a` and makes a new list that has only the even elements of this list in it

```
a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100] even_elements = [x for x in a  
    if x % 2 == 0]  
print(even_elements)
```

Output Clear

```
[4, 16, 36, 64, 100]  
==== Code Execution Successful ====
```

- 12. Generate a random number between 1 and 9 (including 1 and 9). Ask the user to guess the number, then tell them whether they guessed too low, too high, or exactly right.**

```
import random

number = random.randint(1, 9)
guess = 0

while guess != number:
    guess = int(input("Guess a number between 1 and 9: "))
    if guess < number:
        print("Too low!")
    elif guess > number:
        print("Too high!")
    else:
        print("Exactly right!")
```

Output	Clear
Guess a number between 1 and 9: 7 Too high! Guess a number between 1 and 9: 5 Exactly right!  ==== Code Execution Successful ===	

- 13. Ask the user for a number and determine whether the number is prime or not.**

```
num = int(input("Enter a number: "))

if num <= 1:
    print("The number is not prime.") else:
    is_prime = True    for
    i in range(2, num):
        if num % i == 0:
            is_prime = False
            break
    if is_prime:
        print("The number is prime.")
    else:
        print("The number is not prime.")
```

**Output** Clear

```
Enter a number: 7
The number is prime.

==== Code Execution Successful ====
```

- 14. Write a program (function!) that takes a list and returns a new list that contains all the elements of the first list minus all the duplicates**

```
def remove_duplicates(lst):
    new_list = []
    for item in lst:
        if item not in new_list:
            new_list.append(item)
    return new_list

a = [1, 2, 2, 3, 4, 4, 5] result =
remove_duplicates(a)      print("List
without duplicates:", result)
```

**Output** Clear

```
List without duplicates: [1, 2, 3, 4, 5]

==== Code Execution Successful ====
```

- 15. Write a function that takes an ordered list of numbers (a list where the elements are in order from smallest to largest) and another number. The function decides whether or not the given number is inside the list and returns (then prints) an appropriate Boolean**

```
def check_number(ordered_list, number):
    for item in ordered_list:
        if item == number:
            return True
    return False
```

```
# Example usage
numbers = [1, 3, 5, 7, 9, 11, 13, 15]
num = int(input("Enter a number to search: "))

result = check_number(numbers, num) print(result)
```

**Output**

Enter a number to search: 11  
True  
==== Code Execution Successful ====

**Clear**

- 16. Implement a function that takes as input three variables, and returns the largest of the three.**

```
def largest(a, b, c):
    if a >= b and a >= c:
        return a    elif b >=
    a and b >= c:
        return b
    else:
        return c

# Example usage
x = int(input("Enter first number: ")) y =
int(input("Enter second number: "))
z = int(input("Enter third number: "))

print("The largest number is:", largest(x, y, z))
```

**Output**

Enter first number: 46  
Enter second number: 54  
Enter third number: 32  
The largest number is: 54  
==== Code Execution Successful ====

**Clear**

- 17. Python program to perform read and write operations on a file**

```
# Writing to the file
file = open("example.txt", "w")
file.write("Hello, this is a sample file.\n")
file.write("We are performing read and write operations.")
file.close()
```

```
# Reading from the file file =
open("example.txt", "r")
content = file.read()
file.close()

print("File content:")
print(content)
```

#### 18. Python program to copy the contents of a file to another file.

```
print("Enter the text to copy. Type 'END' on a new line to finish:")

lines = [] while
True:    line =
input()    if line ==
"END":      break
lines.append(line)

text = "\n".join(lines)

# Simulate copying to another "file" using a variable copied_text
= text

print("\nContents copied successfully. The copied content is:\n")
print(copied_text)
```

Output

Clear

```
Enter the text to copy. Type 'END' on a new line to finish:
Hi there
END

Contents copied successfully. The copied content is:

Hi there
```

#### 19. Python program to count frequency of characters in a given file.

```
print("Enter/paste your text below. Type 'END' on a new line to finish:")

lines = [] while
True:    line =
```

```

input()  if line ==
"END":    break
lines.append(line)

text = "\n".join(lines) char_freq
= {}

for char in text:  if char
in char_freq:
char_freq[char] += 1
else:
char_freq[char] = 1

print("\nCharacter Frequency:") for
char, freq in char_freq.items():  if
char == "\n":
print("\'\\n\' :", freq)
elif char == " ":
print("'space' :", freq)
else:
print(f"'{char}' :", freq)

```

**Output**

Clear

Enter/paste your text below. Type 'END' on a new line to finish:

Hello World  
END

Character Frequency:

'H'	:	1
'e'	:	1
'l'	:	3
'o'	:	2
'space'	:	1
'W'	:	1
'r'	:	1
'd'	:	1

## 20. Python program to print each line of a file in reverse order

```

print("Enter/paste your text below. Type 'END' on a new line to finish:")
lines = [] while True:  line = input()  if line == "END":    break
lines.append(line) print("\nLines in reverse order:") for line in
reversed(lines):
print(line)

```

**Output** Clear

```
Enter/paste your text below. Type 'END' on a new line to finish:  
Hi,there  
I m using python  
END  
  
Lines in reverse order:  
I m using python  
Hi,there  
  
==== Code Execution Successful ===
```

**21. Python program to compute the number of characters, words and lines in a file**

```
print("Enter/paste your text below. Type 'END' on a new line to finish:")  
  
lines = [] while  
True:  
    line = input()  
    if line == "END":  
        break  
    lines.append(line)  
  
text = "\n".join(lines)  
  
num_chars = len(text)  
num_words = len(text.split())  
  
num_lines = len(lines)  
print(f"\nNumber of characters: {num_chars}") print(f"Number  
of words: {num_words}")  
print(f"Number of lines: {num_lines}")
```

**Output** Clear

```
Enter/paste your text below. Type 'END' on a new line to finish:  
python program  
END  
  
Number of characters: 15  
Number of words: 2  
Number of lines: 1  
  
==== Code Execution Successful ===
```

**22. Write a program that prompts the user to enter his name. The program then greets the person with his name. But if the person's name is 'Rahul' and exception is thrown and he is asked to quit the program.**

```

class QuitProgramException(Exception):
    pass

try:
    name = input("Enter your name: ")
    if name == "Rahul":
        raise QuitProgramException("Access denied! Please quit the program.")
    print(f"Hello, {name}! Welcome!")
except QuitProgramException as e:
    print(e)
    print("Program terminated.")

```

**Output**

Clear

```

Enter your name: Rahul
Access denied! Please quit the program.
Program terminated.

== Code Execution Successful ==

```

- 23. Write a program that accepts date of birth along with the other personal details of a person. Throw an exception if an invalid date is entered**

```

from datetime import datetime

class InvalidDateException(Exception):
    pass

try:
    name = input("Enter your name: ")
    age = input("Enter your age: ")
    dob_input = input("Enter your date of birth (DD-MM-YYYY): ")
    try:
        dob = datetime.strptime(dob_input, "%d-%m-%Y")
    except ValueError:
        raise InvalidDateException("Invalid date entered!")

    print("\nPersonal Details:")
    print(f"Name: {name}")  print(f"Age: {age}")
    print(f"Date of Birth: {dob.strftime('%d-%m-%Y')}")

except InvalidDateException as e:
    print(e)
    print("Please enter a valid date of birth.")

```

Output	Clear
<pre>Enter your name: John Enter your age: 20 Enter your date of birth (DD-MM-YYYY): 24-7-05 Invalid date entered! Please enter a valid date of birth.  ==== Code Execution Successful ===</pre>	

- 24. Write a Regular Expression to represent all 10 digit mobile numbers. Rules: 1. Every number should contains exactly 10 digits. 2. The first digit should be 7 or 8 or 9 Write a Python Program to check whether the given number is valid mobile number or not?**

```
import re

pattern = r'^[789]\d{9}$'
mobile = input("Enter your 10-digit mobile number: ")

if re.match(pattern, mobile):
    print("Valid mobile number.") else:
    print("Invalid mobile number.")
```

Output	Clear
<pre>Enter your 10-digit mobile number: 9812356734 Valid mobile number.  ==== Code Execution Successful ===</pre>	

- 25. A spell checker can be a helpful tool for people who struggle to spell words correctly. In this exercise, you will write a program that reads a file and displays all of the words in it that are misspelled. Misspelled words will be identified by checking each word in the file against a list of known words. Any words in the user's file that do not appear in the list of known words will be reported as spelling mistakes. The user will provide the name of the file to check for spelling mistakes as a command line parameter. Your program should display an appropriate error message if the command line parameter is missing. An error message should also be displayed if your program is unable to open the user's file. Words followed by a comma, period or other punctuation mark are not reported as spelling mistakes. Ignore the capitalization of the words when checking their spelling.**

```
import string

known_words = {"this", "is", "a", "sample", "text", "with", "some", "words", "for", "testing"}
print("Enter/paste your text below. Type 'END' on a new line to finish:")
```

```
lines = [] while True:  
    line = input() if  
    line == "END":  
        break  
    lines.append(line)  
  
text = "\n".join(lines)  
  
translator = str.maketrans(string.punctuation, ' '*len(string.punctuation))  
clean_text = text.translate(translator).lower() words = clean_text.split()  
misspelled = [word for word in words if word not in known_words]  
  
if misspelled:  
    print("\nMisspelled words:")  
for word in misspelled:  
    print(word) else:  
    print("\nNo spelling mistakes found.")
```

Output Clear

Enter/paste your text below. Type 'END' on a new line to finish:

```
hi there i m using python  
END  
  
Misspelled words:  
hi  
there  
i  
m  
using  
python  
  
==== Code Execution Successful ===
```