

PYTHON PROGRAMMING LAB
School of Computer Applications
Department of Computer Applications

SUBMITTED BY

Student Name	Deepak Mangla
Roll No	24/SCA/BCA(AI&ML)/015
Programme	BCA(AI&ML)
Semester	3rd Semester
Section/Group	3rd C
Department	Computer Applications
Batch	2024-2028

SUBMITTED TO

Faculty Name	Dr. Sakshi Gupta
---------------------	------------------

Ques 1)W.a.p. In python to calculate number of days between two dates.

Input:

```
from datetime import date
```

```
date1 = input("Enter the first date (YYYY-MM-DD): ")
```

```
date2 = input("Enter the second date (YYYY-MM-DD): ")
```

```
date1 = date.fromisoformat(date1)
```

```
date2 = date.fromisoformat(date2)
```

```
dd = abs((date2 - date1).days)
```

```
print(f"The number of days between the two dates is: {dd} days")
```

Output:

```
Enter the first date (YYYY-MM-DD): 2025-08-20
```

```
Enter the second date (YYYY-MM-DD): 2025-08-26
```

```
The number of days between the two dates is: 6 days
```

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

Ques 2)W.a.p. In python that accepts an integer(n) and computes the value of n+nn+nnn.

Input:

```
n = int(input("Enter an integer: "))

nn = int(str(n) * 2)
nnn = int(str(n) * 3)

result = n + nn + nnn

print(f"The result of n + nn + nnn is: {result}")
```

Output:

```
Enter an integer: 5
The result of n + nn + nnn is: 615
```

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

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

Input:

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

```
if number % 2 == 0:  
    print(f"{number} is even.")  
else:  
    print(f"{number} is odd.")
```

Output:

```
Enter a number: 5  
5 is odd.
```

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

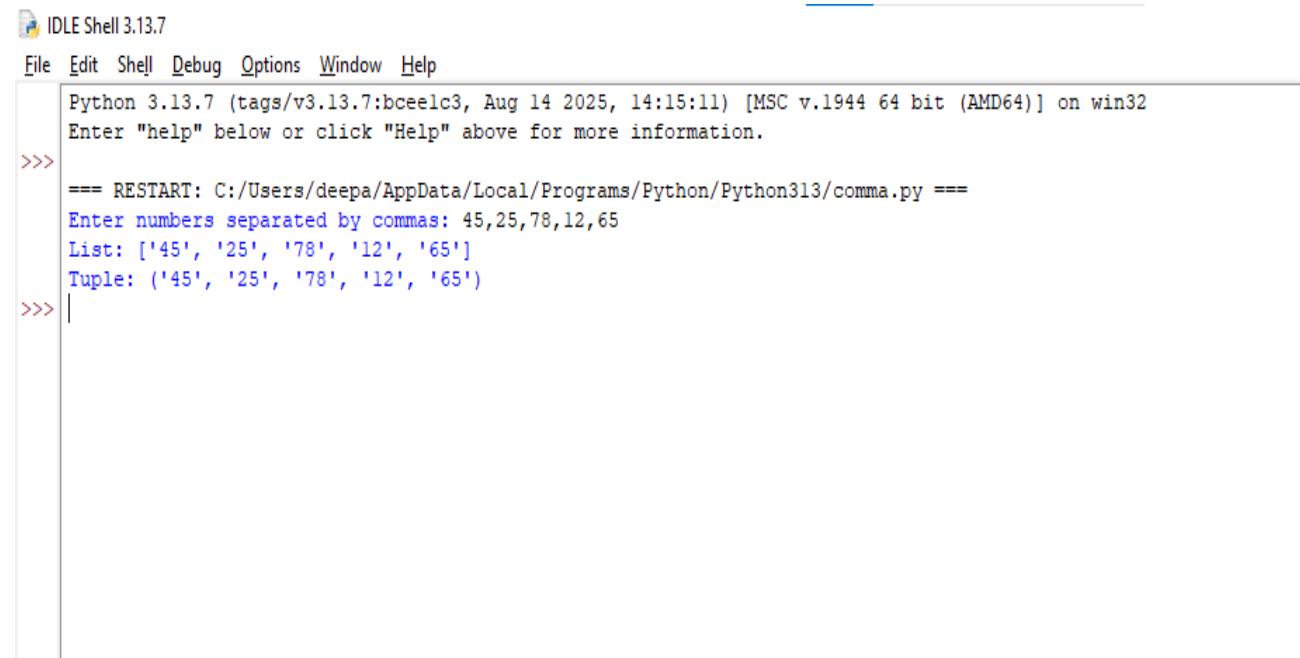
Ques 4) Write a Python program which accepts a sequence of comma-separated numbers from user and generate a list and a tuple with those numbers.

Input:

```
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:



The screenshot shows the Python IDLE Shell interface. The title bar reads "IDLE Shell 3.13.7". The menu bar includes "File", "Edit", "Shell", "Debug", "Options", "Window", and "Help". The main window displays the Python interpreter's prompt (">>>") and the output of a script named "comma.py". The script prompts the user for input, splits it into a list of numbers, and then prints both the list and the tuple formed from the list.

```
Python 3.13.7 (tags/v3.13.7:bceelc3, Aug 14 2025, 14:15:11) [MSC v.1944 64 bit (AMD64)] on win32
Enter "help" below or click "Help" above for more information.
>>>
==== RESTART: C:/Users/deepa/AppData/Local/Programs/Python/Python313/comma.py ====
Enter numbers separated by commas: 45,25,78,12,65
List: ['45', '25', '78', '12', '65']
Tuple: ('45', '25', '78', '12', '65')
```

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

Input:

```
def sum_of_three(a, b, c):
    total = a + b + c
    if a == b == c:
        return 3 * total
    else:
        return total

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

result = sum_of_three(x, y, z)
print("Result:", result)
```

Output:

 IDLE Shell 3.13.7

```
File Edit Shell Debug Options Window Help
Python 3.13.7 (tags/v3.13.7:bceelc3, Aug 14
AMD64) ] on win32
Enter "help" below or click "Help" above fo
>>>
===== RESTART: C:/Users/deepa/AppData/Local/
Enter first number: 45
Enter second number: 12
Enter third number: 25
Result: 82
```

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

Input:

```
def check_vowel(letter):
    vowels = "aeiouAEIOU"
    if letter in vowels:
        return True
    else:
        return False
```

```
ch = input("Enter a single letter: ")
```

```
if len(ch) == 1 and ch.isalpha():
    if check_vowel(ch):
        print(f"{ch} is a vowel.")
    else:
        print(f"{ch} is not a vowel.")
else:
    print("Please enter a valid single alphabet letter.")
```

Output:

 IDLE Shell 3.13.7

```
File Edit Shell Debug Options Window Help
Python 3.13.7 (tags/v3.13.7:bceelc3, Aug 14 2025, 14:15:11) [MSC
AMD64] on win32
Enter "help" below or click "Help" above for more information.
>>>
    === RESTART: C:/Users/deepa/AppData/Local/Programs/Python/Python
    Enter a single letter: a
    a is a vowel.
>>>
    === RESTART: C:/Users/deepa/AppData/Local/Programs/Python/Python
    Enter a single letter: c
    c is not a vowel.
>>> |
```

Ques7)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.

Input:

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

```
for number in a:
```

```
    if number < 5:
```

```
        print(number)
```

Output

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
for number in a:
    if number < 5:
        print(number)
```

```
1
1
2
3
```

a) Input:

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

```
new_list = []
```

```
for x in a:  
    if x < 5:  
        new_list.append(x)  
print(new_list)
```

Output:



The screenshot shows the IDLE Shell interface. The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. Below the menu is a command-line window with the following text:

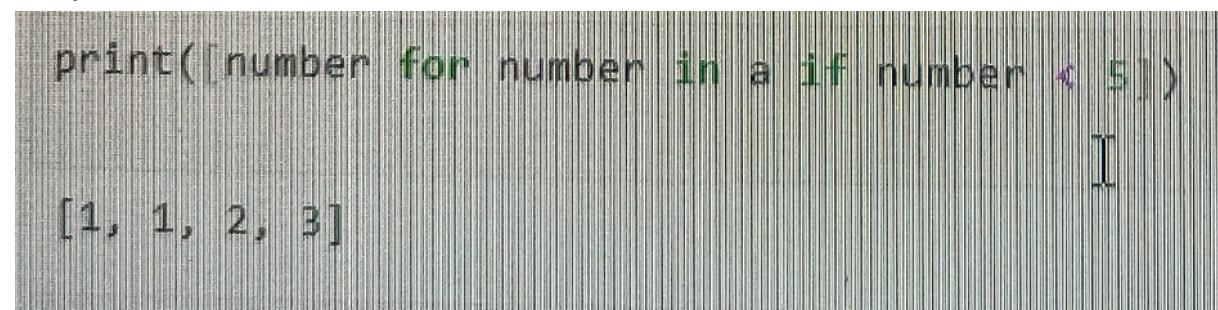
```
File Edit Shell Debug Options Window Help  
Python 3.13.7 (tags/v3.13.7:bceelc3, Aug 14 2023, 16:30:00)  
[GCC 11.2.0] on win32  
Enter "help" below or click "Help" above for more information.  
>>> ===== RESTART: C:/Users/deepa/AppData/Local/Programs/Python/Python313/test.py =====  
[1, 1, 2, 3]  
>>> |
```

b)

Input:

```
print([i for i in a if i < 5])
```

Output:



The screenshot shows a terminal window with the following text:

```
print([number for number in a if number < 5])  
[1, 1, 2, 3]
```

C)

Input:

```
a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
num = int(input("Enter a number: "))
filtered_list = [x for x in a if x < num]
print("Elements less than", num, ":", filtered_list)
```

Output:

```
Enter a number: 12
Elements less than 12 : [1, 1, 2, 3, 5, 8]

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

Ques 8) Create a program that asks the user for a number and then prints out a list of all the divisors of that number. (If you don't know what a divisor is, it is a number that divides evenly into another number. For example, 13 is a divisor of 26 because $26 / 13$ has no remainder.)

Input:

```
num = int(input("Enter a number: "))
divisors = [i for i in range(1, num + 1) if num % i == 0]
print("Divisors of", num, ":", divisors)
```

Output:

```
Enter a number: 34
Divisors of 34 : [1, 2, 17, 34]
```

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

Ques 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.

Input:

```
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 = list(set(a) & set(b))  
print("Common elements:", common)
```

Output:

```
Common elements: [1, 2, 3, 5, 8, 13]
```

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

Ques 10). Ask the user for a string and print out whether this string is a palindrome or not. (A palindrome is a string that reads the same forwards and backwards.)

Input:

```
s = input("Enter a string: ")  
if s == s[::-1]:  
    print("Palindrome")  
else:  
    print("Not a palindrome")
```

Output:

```
Enter a string: deepak  
Not a palindrome
```

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

Ques 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.

Input:

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

Output:

```
[4, 16, 36, 64, 100]
```

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

Ques 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. (Hint: remember to use the user input lessons from the very first exercise)

Input:

```
import random
```

```
num = random.randint(1, 9)
```

```
guess = int(input("Guess a number between 1 and 9: "))
```

```
if guess < num:
```

```
    print("Too low!")
```

```
elif guess > num:
```

```
    print("Too high!")
```

```
else:
```

```
    print("Exactly right! ")
```

Output:

```
Guess a number between 1 and 9: 7
```

```
Too high!
```

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

Ques 13) Ask the user for a number and determine whether the number is prime or not. (For those who have forgotten, a prime number is a number that has no divisors.).

Input:

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

```
if num > 1:  
    for i in range(2, int(num**0.5) + 1):  
        if num % i == 0:  
            print(num, "is not a prime number")  
            break  
    else:  
        print(num, "is a prime number")  
else:  
    print(num, "is not a prime number")
```

Output:

```
Enter a number: 45  
45 is not a prime number
```

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

Ques 14) Write a program to remove all duplicate elements from a list
nums = [1, 2, 2, 3, 4, 4, 5] and print the unique list in ascending order.

Input:

```
nums = [1, 2, 2, 3, 4, 4, 5]
```

```
unique_nums = sorted(set(nums))
```

```
print("Unique list in ascending order:", unique_nums)
```

Output:

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    Filter
[Running] python -u "c:\Users\deepa\OneDrive\Desktop\python\p1.py"
Unique list in ascending order: [1, 2, 3, 4, 5]
[Done] exited with code=0 in 27.712 seconds
```

Ques 15) Given a list of tuples students = [('Alice', 85), ('Bob', 90), ('Charlie', 78)], write a program to print the name of the student with the highest marks.

Input:

```
students = [('Alice', 85), ('Bob', 90), ('Charlie', 78)]
```

```
top_student = max(students, key=lambda x: x[1])
```

```
print("Student with the highest marks:", top_student[0])
```

Output:

```
Student with the highest marks: Bob
```

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

Ques 16) Given a list numbers = [1, 2, 3, 4, 5], create a dictionary where the key is the number and the value is the cube of the number.

Input:

```
numbers = [1, 2, 3, 4, 5]
```

```
cube_dict = {}
```

```
for n in numbers:
```

```
    cube_dict[n] = n ** 3
```

```
print("Dictionary with cubes:", cube_dict)
```

Output:

```
Dictionary with cubes: {1: 1, 2: 8, 3: 27, 4: 64, 5: 125}
```

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

Ques 17) Given a list fruits = ['apple', 'banana', 'apple', 'orange', 'banana', 'apple'], write a program to count the occurrence of each fruit and store it in a dictionary.

Input:

```
fruits = ['apple', 'banana', 'apple', 'orange', 'banana', 'apple']

fruit_count = {}

for fruit in fruits:
    if fruit in fruit_count:
        fruit_count[fruit] += 1
    else:
        fruit_count[fruit] = 1

print("Fruit count:", fruit_count)
```

Output:

```
Fruit count: {'apple': 3, 'banana': 2,
               'orange': 1}
```

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