Manav Rachna International Institute of Research and Studies

School of Computer Applications

Department of Computer Applications

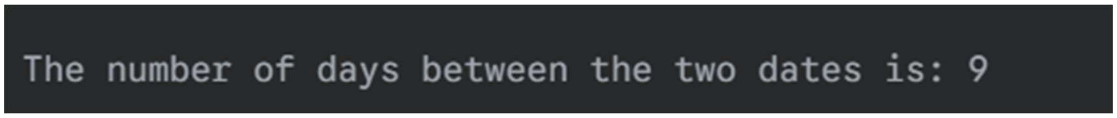
Submitted By	
Student Name	Mohammad Riaz
Roll No	24/SCA/BCA(AI/ML)/039
Program	BCA(AI&ML)
Subject	Python Programming
Semester	3
Section/Group	C
Department	School of Computer Applications
Session / Batch	2024-27
Submitted To	
Faculty Name	Dr.Sakshi Gupta

Question 1: Write a Python program to calculate the number of days between two dates. Sample dates: (2014, 7, 2), (2014, 7, 11).

Ans:

```
from datetime import date

date1 = date(2014, 7, 2)
date2 = date(2014, 7, 11)
delta = date2 - date1
print(f"The number of days between the two dates is: {delta.days}")
```

OUTPUT:



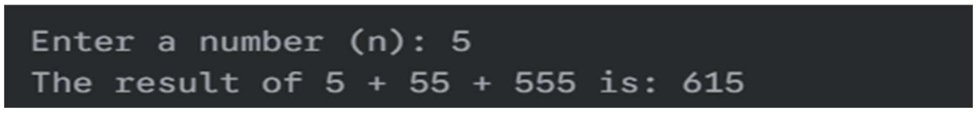
```
The number of days between the two dates is: 9
```

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

Ans:

```
n_str = input("Enter a number (n): ")
n1 = int(n_str)
n2 = int(n_str + n_str)
n3 = int(n_str + n_str + n_str)
result = n1 + n2 + n3
print(f"The result of {n1} + {n2} + {n3} is: {result}")
```

OUTPUT:



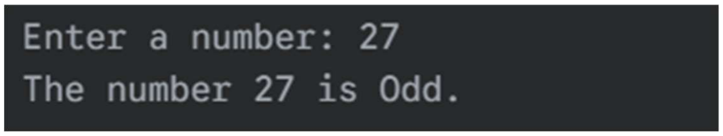
```
Enter a number (n): 5
The result of 5 + 55 + 555 is: 615
```

Question 3 : Write a Python program for accepting a number and depending on whether the number is even or odd, print out an appropriate message to the user.

ANS:

```
try:
  num = int(input("Enter a number: "))
  if num % 2 == 0:
    print(f"The number {num} is Even.")
  else:
    print(f"The number {num} is Odd.")
except ValueError:
  print("Invalid input. Please enter an integer.")
```

OUTPUT:



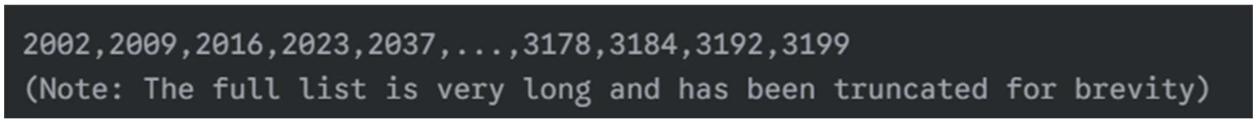
```
Enter a number: 27
The number 27 is Odd.
```

Question 4: Write a Python program which will find all such numbers which are divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included).

Ans:

```
result_list = []
for num in range(2000, 3201):
  if (num % 7 == 0) and (num % 5 != 0):
    result_list.append(str(num))


print(",".join(result_list))
```

OUTPUT:



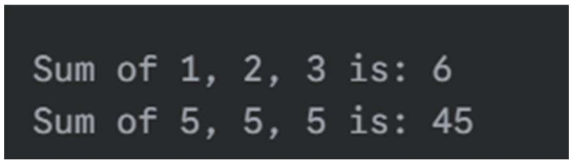
```
2002,2009,2016,2023,2037,...,3178,3184,3192,3199
(Note: The full list is very long and has been truncated for brevity)
```

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

ANS:

```
def calculate_sum(a, b, c):
  if a == b == c:
    return (a + b + c) * 3
  else:
    return a + b + c
print(f"Sum of 1, 2, 3 is: {calculate_sum(1, 2, 3)}")
print(f"Sum of 5, 5, 5 is: {calculate_sum(5, 5, 5)}")
```

OUTPUT:



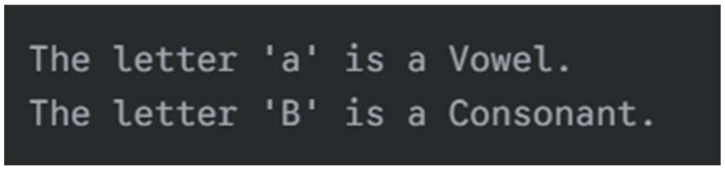
```
Sum of 1, 2, 3 is: 6
Sum of 5, 5, 5 is: 45
```

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

ANS:

```
def check_vowel(letter):
  vowels = "aeiou"
  if letter.lower() in vowels:
    print(f"The letter '{letter}' is a Vowel.")
    else:
        print(f"The letter '{letter}' is a Consonant.")


check_vowel('a')
check_vowel('B')
```

OUTPUT:



```
The letter 'a' is a Vowel.
The letter 'B' is a Consonant.
```

Question 7: Take a list, say for example this one: a = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89] and write the following program the following:

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 the new list.

b) Write this in one line of Python.

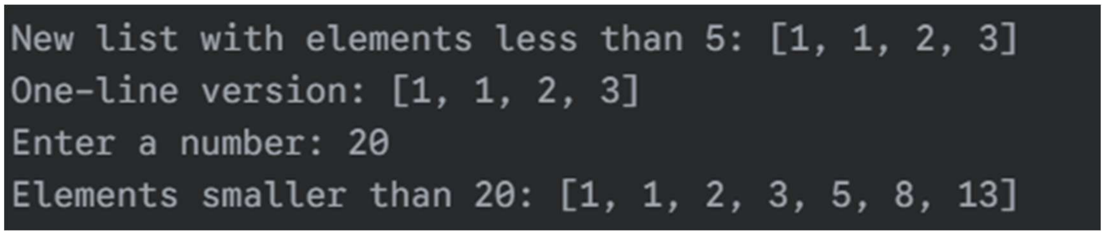
c) Ask the user for a number and return a list that contains only elements from the original list that are smaller than that number given by the user.

ANS:

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


new_list = []
for item in a:
    if item < 5:
        new_list.append(item)
print(f"a) New list with elements less than 5: {new_list}")
```

```
one_line_list = [item for item in a if item < 5]

print(f"b) One-line version: {one_line_list}")

try:

   user_num = int(input("c) Enter a number: "))

   user_list = [item for item in a if item < user_num]

   print (f"   Elements smaller than {user_num}: {user_list}")

except ValueError:

   print ("Invalid input.")
```

OUTPUT:



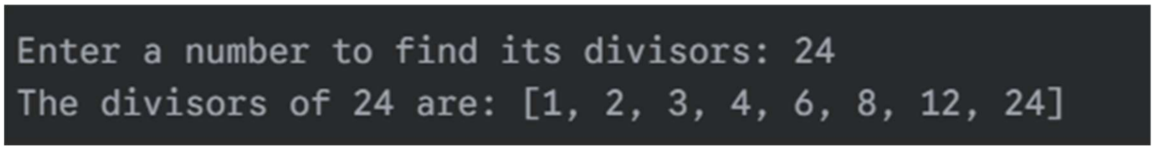
```
New list with elements less than 5: [1, 1, 2, 3]
One-line version: [1, 1, 2, 3]
Enter a number: 20
Elements smaller than 20: [1, 1, 2, 3, 5, 8, 13]
```

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

ANS:

```
try:

   num = int(input("Enter a number to find its divisors: "))

   divisors = [i for i in range(1, num + 1) if num % i == 0]

   print(f"The divisors of {num} are: {divisors}")

except ValueError:

   print("Please enter a valid integer.")
```

OUTPUT:



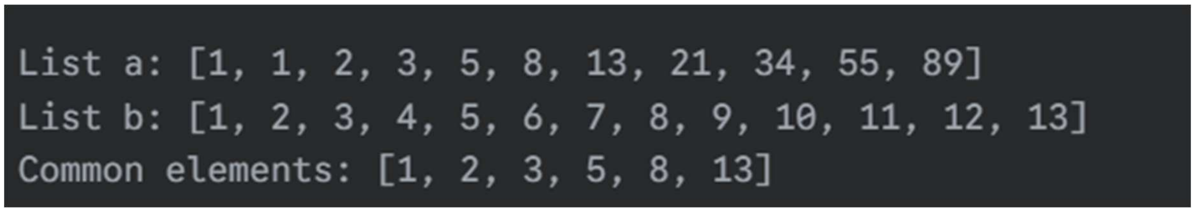
```
Enter a number to find its divisors: 24
The divisors of 24 are: [1, 2, 3, 4, 6, 8, 12, 24]
```

Question 9: Take two lists and write a program that returns a list that contains only the elements that are common between the lists (without duplicates).

ANS:

```
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 = list(set(a) & set(b))

print(f"List a: {a}")

print(f"List b: {b}")

print(f"Common elements: {common_elements}")
```

OUTPUT:



```
List a: [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]
List b: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13]
Common elements: [1, 2, 3, 5, 8, 13]
```

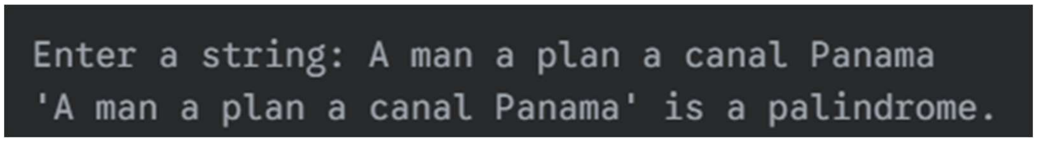
Question 10: Ask the user for a string and print out whether this string is a palindrome or not.

ANS:

```
text = input("Enter a string: ")
```

```
normalized_text = text.replace(" ", "").lower()


if normalized_text == normalized_text[::-1]:
   print(f"'{text}' is a palindrome.")
else:
   print(f"'{text}' is not a palindrome.")
```

OUTPUT:



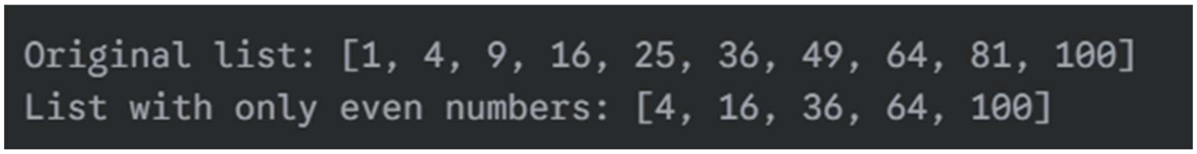
```
Enter a string: A man a plan a canal Panama
'A man a plan a canal Panama' is a palindrome.
```

Question 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.

ANS:

```
a = [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

even_list = [num for num in a if num % 2 == 0]

print(f"Original list: {a}")

print(f"List with only even numbers: {even_list}")
```

OUTPUT:



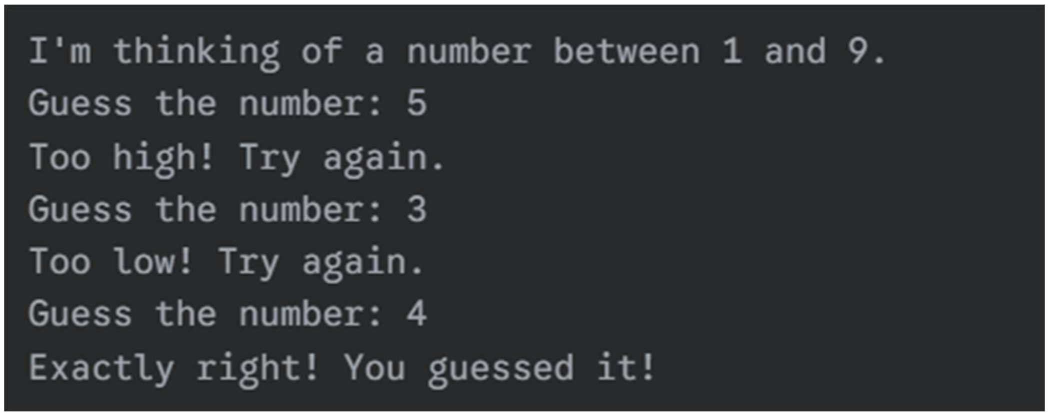
```
Original list: [1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
List with only even numbers: [4, 16, 36, 64, 100]
```

Question 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.

Ans:

```
import random


secret_number = random.randint(1, 9)

guess = 0


print("I'm thinking of a number between 1 and 9.")


while guess != secret_number:

  try:

    guess = int(input("Guess the number: "))

    if guess < secret_number:

      print("Too low! Try again.")

    elif guess > secret_number:

      print("Too high! Try again.")

    else:

      print("Exactly right! You guessed it!")

  except ValueError:

    print("Invalid input. Please enter a number.")
```

OUTPUT:



```
I'm thinking of a number between 1 and 9.
Guess the number: 5
Too high! Try again.
Guess the number: 3
Too low! Try again.
Guess the number: 4
Exactly right! You guessed it!
```

Q13: **Ask the user for a number and determine whether the number is prime or not.**

```python
import math

def is_prime(n):
    if n <= 1:
        return False
    if n <= 3:
        return True
    if n % 2 == 0:
        return False
    i = 3
    while i * i <= n:
        if n % i == 0:
            return False
        i += 2
    return True

if __name__ == '__main__':
    n = int(input('Enter a number: '))
    if is_prime(n):
        print(f"{n} is a prime number")
    else:
        print(f"{n} is not a prime number")
```

Input

17

Output

```
17 is a prime number
```

Q14. **Write a program that takes a list and returns a new list that contains all the elements of the first list minus duplicates.**

Program (q14_remove_duplicates.py)

```
def remove_duplicates(lst):
    seen = set()
    result = []
    for item in lst:
        if item not in seen:
            seen.add(item)
            result.append(item)
    return result

if __name__ == '__main__':
    raw = input('Enter numbers separated by space: ').strip()
    lst = list(map(int, raw.split())) if raw else []
    print('Result without duplicates:', remove_duplicates(lst))
```

Input

```
1 2 2 3 4 3 5 1
```

Output

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

Q15. **Write a function that takes an ordered list of numbers (ascending) and another number. Decide whether the given number is inside the list and return/print a boolean.**

```
def in_ordered_list(lst, num):
    lo, hi = 0, len(lst)-1
    while lo <= hi:
        mid = (lo + hi) // 2
        if lst[mid] == num:
            return True
        if lst[mid] < num:
            lo = mid + 1
        else:
            hi = mid - 1
    return False

if __name__ == '__main__':
    raw = input('Enter ascending numbers separated by space: ').strip()
    lst = list(map(int, raw.split())) if raw else []
    num = int(input('Enter number to search: '))
    print(in_ordered_list(lst, num))
```

Input

```
1 3 5 7 9 11
5
```

Output

```
True
```

Q16. **Implement a function that takes as input three variables and returns the largest of the three (without using built-in `max()` function).**

```
def largest_of_three(a, b, c):
    largest = a
    if b > largest:
        largest = b
    if c > largest:
        largest = c
    return largest

if __name__ == '__main__':
    a = int(input('Enter a: '))
    b = int(input('Enter b: '))
    c = int(input('Enter c: '))
    print('Largest is', largest_of_three(a,b,c))
```

Input

```
3
7
5
```

Output

```
Largest is 7
```

Q17. **Python program to perform read and write operations on a file.**

```
path = input('Enter file path (will be created if not exists): ').strip()
# Read (if exists) and append a line.
try:
    with open(path, 'r') as f:
        print('Current file contents:\n')
        print(f.read())
except FileNotFoundError:
    print('File not found — will create a new one.')

with open(path, 'a') as f:
    line = input('Enter a line to append: ') + '\n'
    f.write(line)

print('Done. New contents:')
with open(path, 'r') as f:
    print(f.read())
```

Q18. **Python program to copy the contents of a file to another file.**

```
src = input('Enter source file path: ').strip()
dst = input('Enter destination file path: ').strip()
with open(src, 'r') as s, open(dst, 'w') as d:
    d.write(s.read())
print('Copy complete.')
```

Sample Run: Copy `sample_file.txt` to `sample_copy.txt`.

Q19. **Python program to count frequency of characters in a given file.**

Program (q19_char_freq.py)

```
from collections import Counter
path = input('Enter file path: ').strip()
with open(path,'r', encoding='utf-8') as f:
    txt = f.read()
freq = Counter(txt)
for ch, cnt in freq.most_common():
    print(repr(ch), cnt)
```

Q20. **Python program to print each line of a file in reverse order (last line first).**

```
path = input('Enter file path: ').strip()
with open(path) as f:
    lines = f.readlines()
for line in reversed(lines):
    print(line.rstrip('\n'))
```

Q21. **Python program to compute the number of characters, words and lines in a file.**

```
path = input('Enter file path: ').strip()
with open(path) as f:
    content = f.read()
    lines = content.splitlines()
num_chars = len(content)
num_words = len(content.split())
num_lines = len(lines)
print('Characters:', num_chars)
print('Words:', num_words)
print('Lines:', num_lines)
```

Q22. **Write a program that prompts the user to enter his name. The program greets the person with his name. But if the person's name is 'Rahul', an exception is thrown and the program is asked to quit.**

```
name = input('Enter your name: ').strip()
if name == 'Rahul':
    raise ValueError('Name is Rahul — quitting as per requirement')
print('Hello,', name)
```

Sample: - Input `Riaz` -> `Hello, Riaz` - Input `Rahul` -> program raises exception.

Q23. **Write a program that accepts date of birth along with other personal details. Throw an exception if an invalid date is entered.**

```
from datetime import datetime
name = input('Enter name: ').strip()
dob_str = input('Enter DOB (YYYY-MM-DD): ').strip()
try:
    dob = datetime.strptime(dob_str, '%Y-%m-%d').date()
except ValueError:
    raise ValueError('Invalid date format. Expected YYYY-MM-DD')
print(f'Name: {name}, DOB: {dob}')
```

Q24. **Write a Regular Expression to represent all 10-digit mobile numbers. Rules: every number should contain exactly 10 digits. The first digit should be 7 or 8 or 9. Write a Python program to check whether the given number is valid.**

```
import re
pattern = re.compile(r'^[789]\d{9}$')
```

```
num = input('Enter 10-digit mobile number: ').strip()
if pattern.match(num):
    print('Valid mobile number')
else:
    print('Invalid mobile number')
```

Sample: `9123456789 -> Valid`, `6123456789 -> Invalid`.

Q25. A spell checker: program reads a file and displays all words that are not in a known-words list (common dictionary).

```
import re
# For practical file, include a small dictionary file or use an available wor
d list.
dict_path = input('Enter dictionary file path (one word per line): ').strip()
input_path = input('Enter file to check: ').strip()
with open(dict_path) as d:
    known = {w.strip().lower() for w in d if w.strip()}
with open(input_path) as f:
    text = f.read().lower()
words = re.findall(r"[a-z]+", text)
misspelled = sorted({w for w in words if w not in known})
if misspelled:
    print('Misspelled / unknown words:')
    for w in misspelled:
        print(w)
else:
    print('No unknown words found')
```