LOTUS VALLEY INTERNATIONAL SCHOOL: Computer Science Practical File

Pranav Verma (Roll No. 7, XII Aryabhatta)

November 1, 2025

Contents

1 Review of Python Basics

Question 15

Write a program to calculate the mean of a given list of numbers.

Solution

```
numbers = eval(input("Enter a list: "))
if len(numbers) == 0:
    print("List is empty, cannot calculate mean.")
else:
    total = sum(numbers)
    n = len(numbers)
    mean = total / n
    print("Mean of the numbers:", mean)
```

Question 16

Write a program to calculate the minimum element of a given list of numbers.

Solution

```
l = eval(input("Enter a list: "))
m = min(l)
print("Minimum element in the list:", m)
```

Question 17

Write a code to calculate and display total marks and percentage of a student from a given list storing the marks of a student.

Solution

```
marks_list = eval(input("Enter list of marks: "))
total_marks = sum(marks_list)
total_subjects = len(marks_list)
maximum_marks_per_subject = 100
total_marks_possible = maximum_marks_per_subject * total_subjects
percentage = (total_marks / total_marks_possible) * 100

print("Total Marks:", total_marks)
print("Percentage:", percentage)
```

Question 18

Write a program to multiply an element by 2 if it is an odd index for a given list containing both numbers and strings.

Solution

```
mixed = eval(input("Enter the list: "))
for index in range(1, len(mixed), 2):
        mixed[index] *= 2
print("Modified List:", mixed)
```

Question 19

Write a program to count the frequency of an element in a list.

Solution

```
my_list = eval(input("Enter the list: "))
c = int(input("Enter the element whose frequency is to be checked: "))
frequency = my_list.count(c)
print("The frequency of", c, "in the list is: ", frequency)
```

Question 20

Write a program to shift elements of a list so that the first element moves to the second index and second index moves to the third index, and so on, and the last element shifts to the first position.

Suppose the list is:

```
[10, 20, 30, 40]
```

After shifting, it should look like:

```
[40, 10, 20, 30]
```

Solution

```
l = eval(input("Enter the list: "))
print("Original List")
print(l)

l = l[-1:] + l[:-1]

print("Rotated List")
print(l)
```

Question 21

A list Num contains the following elements:

```
3, 25, 13, 6, 35, 8, 14, 45
```

Write a program to swap the content with the next value divisible by 5 so that the resultant list will look like:

```
25, 3, 13, 35, 6, 8, 45, 14
```

Solution

```
Num = [3, 25, 13, 6, 35, 8, 14, 45]
for i in range(len(Num) - 1):
    if Num[i] % 5 != 0 and Num[i + 1] % 5 == 0:
        Num[i], Num[i + 1] = Num[i + 1], Num[i]
print("Resultant List:", Num)
```

Question 22

Write a program to accept values from a user in a tuple. Add a tuple to it and display its elements one by one. Also display its maximum and minimum value.

Solution

```
tuple1 = eval(input("Enter a tuple: "))
tuple2 = (10, 20, 30)
combined_tuple = tuple1 + tuple2
print("Elements of the combined tuple:")
for element in combined_tuple:
    print(element)

print("Maximum value:", max(combined_tuple))
print("Minimum value:", min(combined_tuple))
```

Question 23

Write a program to input any values for two tuples. Print it, interchange it and then compare them.

Solution

```
tuple1 = eval(input("Enter the first tuple: "))
tuple2 = eval(input("Enter the second tuple: "))

print("Original Tuples:")
print("Tuple 1:", tuple1)
print("Tuple 2:", tuple2)

tuple1, tuple2 = tuple2, tuple1

print("\nSwapped Tuples:")
print("Tuple 1 (after swapping):", tuple1)
print("Tuple 2 (after swapping):", tuple2)

if tuple1 == tuple2:
    print("\nThe swapped tuples are equal.")
else:
    print("\nThe swapped tuples are not equal.")
```

Question 24

Write a Python program to input 'n' classes and names of class teachers to store them in a dictionary and display the same. Also accept a particular class from the user and display the name of the class teacher of that class.

Solution

```
n = int(input("Enter number of classes: "))
data = {}
for i in range(n):
    class_name = input("Enter class name: ")
    teacher_name = input("Enter teacher name: ")
    data[class_name] = teacher_name
print("Class data:", data)
find = input("Enter a class name to find its teacher: ")
if find in data:
    print("Teacher for class", find, "is", data[find])
else:
    print("Class not found in the data.")
```

Question 25

Write a program to store student names and their percentage in a dictionary and delete a particular student name from the dictionary. Also display the dictionary after deletion.

Solution

```
n = int(input("Enter number of students: "))
data = {}
for i in range(n):
    stu_name = input("Enter student name: ")
    percentage = input("Enter percentage: ")
    data[stu_name] = percentage
print("Student data:", data)
find = input("Enter a student name to delete: ")
if find in data:
    del data[find]
    print("Updated student data:", data)
else:
    print("Student not found in the data.")
```

Question 26

Write a Python program to input names of 'n' customers and their details like items bought, cost and phone number, etc., store them in a dictionary and display all the details in a tabular form.

Solution

```
n = int(input("Enter the number of customers: "))
customer_data = {}

for i in range(n):
    name = input("Enter customer name: ")
    items_bought = input("Enter items bought: ")
    cost = float(input("Enter cost: "))
    phone_number = int(input("Enter phone number: "))

    customer_data[name] = {
        'Items Bought': items_bought,
        'Cost': cost,
        'Phone Number': phone_number
    }
```

```
15
16 print("Customer Details:")
17 print("Name\t\tItems Bought\t\tCost\t\tPhone Number")
18 for name, details in customer_data.items():
19     print(name, "\t\t", details['Items Bought'], "\t\t", details['Cost'],
       "\t\t", details['Phone Number'])
```

Question 27

Write a Python program to capitalize first and last letters of each word of a given string.

Solution

```
1 input_string = input("Enter the string: ")
2 words = input_string.split()
3 result = []
4
5 for word in words:
6     if len(word) > 1:
7         modified_word = word[0].upper() + word[1:-1] + word[-1].upper()
8     else:
9         modified_word = word.upper()
10    result.append(modified_word)
11
12 capitalized_string = ' '.join(result)
13 print(capitalized_string)
```

Question 28

Write a Python program to remove duplicate characters of a given string.

Solution

```
1 input_string = input("Enter the string: ")
2 unique_chars = {}
3 for char in input_string:
4     if char not in unique_chars:
5         unique_chars[char] = True
6 result = ''.join(unique_chars.keys())
7 print(result)
```

Question 29

Write a Python program to compute sum of digits of a given number.

Solution

```
1 number = int(input("Enter a number: "))
2 sum_of_digits = 0
3 while number > 0:
4     digit = number % 10
5     sum_of_digits += digit
6     number = number // 10
7
8 print("Sum of digits:", sum_of_digits)
```

Question 30

Write a Python program to find the second most repeated word in a given string.

Solution

```
input_string = input("Enter the string: ")
words = input_string.split()

word_counts = {}
for word in words:
    if word in word_counts:
        word_counts[word] += 1
    else:
        word_counts[word] = 1

max_count = 0
second_max_count = 0
most_repeated_word = None
second_most_repeated_word = None

for word, count in word_counts.items():
    if count > max_count:
        second_max_count = max_count
        max_count = count
        second_most_repeated_word = most_repeated_word
        most_repeated_word = word
    elif count > second_max_count:
        second_max_count = count
        second_most_repeated_word = word

print(second_most_repeated_word)
```

Question 31

Write a Python program to change a given string to a new string where the first and last characters have been exchanged.

Solution

```
input_str = input("Enter the string: ")
first_char = input_str[0]
last_char = input_str[-1]
middle_chars = input_str[1:-1]
new_str = last_char + middle_chars + first_char
print("Original string:", input_str)
print("New string after swapping first and last characters:", new_str)
```

Question 32

Write a Python program to multiply all the items in a list.

Solution

```
lst = eval(input("Enter the list: "))
result = 1
for item in lst:
```

```
    result *= item
print("Result:", result)
```

Question 33

Write a Python program to get the smallest number from a list.

Solution

```
numbers = eval(input("Enter the list: "))
smallest = min(numbers)
print("Smallest Number:", smallest)
```

Question 34

Write a Python program to append a list to the second list.

Solution

```
list1 = eval(input("Enter the first list: "))
list2 = eval(input("Enter the second list: "))
list1.extend(list2)
print("Appended List:", list1)
```

Question 35

Write a Python program to generate and print a list of first and last 5 elements where the values are square of numbers between 1 and 30 (both included).

Solution

```
squares = []
for num in range(1, 31):
    squares.append(num ** 2)
first_5 = squares[:5]
last_5 = squares[-5:]
combined_list = first_5 + last_5
print("Combined list:", combined_list)
```

Question 36

Write a Python program to get unique values from a list.

Solution

```
input_list = eval(input("Enter the list: "))
unique_values = []

for item in input_list:
    if item not in unique_values:
        unique_values.append(item)

print("Unique values from the list:", unique_values)
```

Question 37

Write a Python program to convert a string to a list.

Solution

```
string = input("Enter the string: ")
char_list = list(string)

print("String converted to list:", char_list)
```

Question 38

Write a Python script to concatenate the following dictionaries to create a new one:

```
d1 = {'A': 1, 'B': 2, 'C': 3}
d2 = {'D': 4 }
```

Output should be:

```
{'A': 1, 'B': 2, 'C': 3, 'D' : 4}
```

Solution

```
d1 = {'A': 1, 'B': 2, 'C': 3}
d2 = {'D': 4}
d1.update(d2)
print("Concatenated dictionary: ", d1)
```

Question 39

Write a Python script to check if a given key already exists in a dictionary.

Solution

```
my_dict = eval(input("Enter the dictionary: "))
key_check = input("Enter the key to be checked: ")
if key_check in my_dict:
    print("The key", key_check, "exists in the dictionary.")
else:
    print("The key", key_check, "does not exist in the dictionary.")
```

Question 40

Write a Python script to print a dictionary where the keys are numbers between 1 and 15 (both included) and the values are square of keys.

Sample Dictionary

```
{1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81,
 10: 100, 11: 121, 12: 144, 13: 169, 14: 196, 15: 225}
```

Solution

```
result_dict = {}
for num in range(1, 16):
    result_dict[num] = num ** 2
print("Resulting dictionary:", result_dict)
```

Question 41

Write a Python script to merge two Python dictionaries.

Solution

```
dict1 = eval(input("Enter the first dictionary: "))
dict2 = eval(input("Enter the second dictionary: "))
dict1.update(dict2)
print("Merged dictionary:", dict1)
```

Question 42

Write a Python program to sort a dictionary by key.

Solution

```
def bubble_sort_keys(keys):
    n = len(keys)
    for i in range(n - 1):
        for j in range(0, n - i - 1):
            if keys[j] > keys[j + 1]:
                keys[j], keys[j + 1] = keys[j + 1], keys[j]

my_dict = eval(input("Enter the dictionary: "))
keys_list = list(my_dict.keys())
bubble_sort_keys(keys_list)
sorted_dict = {}
for key in keys_list:
    sorted_dict[key] = my_dict[key]

print("Dictionary sorted by key:", sorted_dict)
```

Question 43

Write a Python program to combine two dictionaries adding values for common keys.

```
d1 = {'a': 100, 'b': 200, 'c':300}
d2 = {'a': 300, 'b': 200, 'd':400}
```

Sample output:

```
Counter({'a': 400, 'b': 400, 'c': 300, 'd': 400})
```

Solution

```
d1 = {'a': 100, 'b': 200, 'c': 300}
d2 = {'a': 300, 'b': 200, 'd': 400}
combined_dict = {}
```

```
for key, value in d1.items():
    combined_dict[key] = value
for key, value in d2.items():
    if key in combined_dict:
        combined_dict[key] += value
    else:
        combined_dict[key] = value

print("Combined dictionary with added values for common keys:",
    combined_dict)
```

Question 44

Write a Python program to find the three highest values in a dictionary.

Solution

```
my_dict = eval(input("Enter the dictionary: "))
highest_values = []
highest_keys = []
# This is not the most efficient way, but it matches the provided code.
# A better way would be to sort items by value.
for key, value in my_dict.items():
    if not highest_values or value > highest_values[-1]:
        highest_values.append(value)
        highest_keys.append(key)
        if len(highest_values) > 3:
            highest_values.pop(0)
            highest_keys.pop(0)
    # Note: This logic is flawed as it doesn't handle insertion correctly.
    # A correct implementation would be more complex or use sorting.
    # Re-implementing based on sorting for correctness, but keeping
       original logic:

# Sticking to the provided logic exactly:
my_dict = eval(input("Enter the dictionary: "))
highest_values = []
highest_keys = []
for key, value in my_dict.items():
    if not highest_values or value > highest_values[-1]:
        highest_values.append(value)
        highest_keys.append(key)
        if len(highest_values) > 3:
            highest_values.pop(0)
            highest_keys.pop(0)
print("Three highest values in the dictionary:")
for i in range(len(highest_keys)):
    print(highest_keys[i], ":" ,highest_values[i])
```

Question 45

Write a Python program to sort a list alphabetically in a dictionary.

Solution

```
my_dict = eval(input("Enter the dictionary: "))
for key, value in my_dict.items():
```

```
    if isinstance(value, list):
        value.sort()

print("Sorted dictionary:", my_dict)
```

Question 46

Write a Python program to count number of items in a dictionary value that is a list.

Solution

```
my_dict = eval(input("Enter the dictionary: "))
total_count = 0
for value in my_dict.values():
    if type(value) is list:
        total_count += len(value)
print("Total number of items in lists within the dictionary:",
    total_count)
```

2 Functions

Question 2

Write a function called calculate_area() that takes base and height as input arguments and returns area of a triangle as an output. The formula used is: Triangle Area = 1/2 * base * height

Solution

```
def calculate_area(base, height):
    area = (1/2) * base * height
    return area

base_value = int(input("Enter the base value: "))
height_value = int(input("Enter the height value: "))
triangle_area = calculate_area(base_value, height_value)
print("Area of the triangle:", triangle_area)
```

Question 3

Modify the above function to take a third parameter called shape type. Shape type should be either triangle or rectangle. Based on the shape, it should calculate the area. Formula used: Rectangle Area = length * width

Solution

```
def calculate_area(base, height, shape_type):
    if shape_type == "triangle":
        area = (1/2) * base * height
    elif shape_type == "rectangle":
        area = base * height
    else:
        area = None
        print("Invalid shape type. Please specify either 'triangle' or
            'rectangle'.")
    return area
```

```
10
11 shape_type = input("Enter the shape type, triangle or rectangle: ")
12 base_value = int(input("Enter the base value: "))
13 height_value = int(input("Enter the height value: "))
14 area = calculate_area(base_value, height_value, shape_type)
15 print("Area of the", shape_type, "is ", area)
```

Question 4

Write a function called print_pattern() that takes integer number as argument and prints the following pattern if the input number is 3.

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

If input is 4, then it should print:

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

Solution

```
1 def print_pattern(num):
2     for i in range(1, num + 1):
3         print("*" * i)
4
5 num = int(input("Enter a number: "))
6 print("Pattern for input", num, ":")
7 print_pattern(num)
```

Question 18

Write a function that takes amount-in-dollars and dollar-to-rupee conversion price; it then returns the amount converted to rupees. Create the function in both void and non-void forms.

Solution

```
1 def convert_dollars_to_rupees(amount_in_dollars, conversion_rate):
2     amount_in_rupees = amount_in_dollars * conversion_rate
3     return amount_in_rupees
4
5 def convert_dollars_to_rupees_void(amount_in_dollars, conversion_rate):
6     amount_in_rupees = amount_in_dollars * conversion_rate
7     print("Amount in rupees:", amount_in_rupees)
8
9 amount = float(input("Enter amount in dollars "))
10 conversion_rate = float(input("Enter conversion rate "))
11
12 # Non-void function call
13 converted_amount = convert_dollars_to_rupees(amount, conversion_rate)
14 print("Converted amount (non-void function):", converted_amount)
15
16 # Void function call
17 convert_dollars_to_rupees_void(amount, conversion_rate)
```

Question 19

Write a function to calculate volume of a box with appropriate default values for its parameters. Your function should have the following input parameters:

- Length of box
- Width of box
- Height of box

Test it by writing a complete program to invoke it.

Solution

```
def calculate_volume(length = 5, width = 3, height = 2):
    return length * width * height

default_volume = calculate_volume()
print("Volume of the box with default values:", default_volume)

v = calculate_volume(10, 7, 15)
print("Volume of the box with default values:", v)

a = calculate_volume(length = 23, height = 6)
print("Volume of the box with default values:", a)

b = calculate_volume(width = 19)
print("Volume of the box with default values:", b)
```

Question 20

Write a program to display first four multiples of a number using recursion.

Solution

```
def display_multiples(n, count = 1):
    if count > 4:
        return
    print(n * count)
    display_multiples(n, count + 1)

n = int(input("Enter the number: "))
display_multiples(n)
```

Question 22

Write a recursive function to add the first 'n' terms of the series:

$$1 + 1/2 - 1/3 + 1/4 - 1/5...$$

Solution

```
def add_series_terms(n):
    if n == 1:
        return 1
    elif n % 2 == 0:
        return add_series_terms(n - 1) + 1 / n
```

```
    else:
        return add_series_terms(n - 1) - 1 / n

n = int(input("Enter the term: "))
print(add_series_terms(n))
```

Question 23

Write a program to find the greatest common divisor between two numbers.

Solution

```
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
gcd_value = gcd(num1, num2)
print("The greatest common divisor of", num1, "and", num2, "is",
    gcd_value)
```

Question 24

Write a Python function to multiply all the numbers in a list.

Sample List: (8, 2, 3, -1, 7)

Expected Output: -336

Solution

```
def multiply_list(numbers):
    product = 1
    for num in numbers:
        product *= num
    return product

numbers = eval(input("Enter the list: "))
product = multiply_list(numbers)
print("The product of the numbers in the list is", product)
```

Question 25

Write a Python function to calculate the factorial of a number (a non-negative integer). The function accepts the number whose factorial is to be calculated as the argument.

Solution

```
def factorial(n):
    product = 1
    for i in range(1, n + 1):
        product *= i
    return product

```

```
n = int(input("Enter a number: "))
fact = factorial(n)
print("The factorial of", n, "is", fact)
```

Question 26

Write a Python function that takes a number as a parameter and checks whether the number is prime or not.

Solution

```
def is_prime(n):
    if n <= 1:
        return False
    factors = 0
    for i in range(1, n + 1):
        if n % i == 0:
            factors += 1
        if factors > 2:
            return False
    return True

num = int(input("Enter a number: "))
is_prime_num = is_prime(num)
print("Is", num, "prime?", is_prime_num)
```

Question 27

Write a Python function that checks whether a passed string is a palindrome or not.

Note: A palindrome is a word, phrase, or sequence that reads the same backward as forward, e.g., madam or nurses run.

Solution

```
def is_palindrome(s):
    s = s.lower()
    return s == s[::-1]

input_string = input("Enter a string: ")
if is_palindrome(input_string):
    print(input_string, "is a palindrome.")
else:
    print(input_string, "is not a palindrome.")
```

Question 28

Write a Python program that accepts a hyphen-separated sequence of words as input and prints the words in a hyphen-separated sequence after sorting them alphabetically.

Sample Items: green-red-yellow-black-white

Expected Result: black-green-red-white-yellow

Solution

```
def sort_words(s):
    words = s.split('-')
    words.sort()
    return '-'.join(words)

input_str = input('Enter a hyphen-separated sequence of words: ')
print(sort_words(input_str))
```

3 File Handling

Question 9

Write a code snippet that will create an object called fileout for writing; associate it with the filename 'STRS'. The code should keep on writing strings to it as long as the user wants.

Solution

```
fileout = open('STRS.txt', 'w')
ans = 'y'
while ans == 'y':
    string = input("Enter a string: ")
    fileout.write(string + "\n")
    ans = input("Want to enter more strings?(y/n)...")
fileout.close()
```

Question 14

Write a program that reads a text file and creates another file that is identical except that every sequence of consecutive blank spaces is replaced by a single space.

Solution

Let the file "input.txt" include the following sample text:

```
In   the beginning  there  was    chaos.
Out of  the chaos  came order.
The universe  began to take shape.
Stars  formed  and galaxies were  born.
Life   emerged  in the  vast  expanse.
```

```
with open("input.txt", 'r') as f:
    with open("output.txt", 'w') as fout:
        for line in f:
            modified_line = ' '.join(line.split())
            fout.write(modified_line + '\n')
```

Question 15

A file 'sports.dat' contains information in the following format: EventName, Participant

Write a function that would read contents from file 'sports.dat' and create a file named 'Athletic.dat' copying only those records from 'sports.dat' where the event name is "Athletics".

Solution

Let the file "sports.dat" include the following sample records:

```
Athletics - Rahul
Swimming - Tanvi
Athletics - Akash
Cycling - Kabir
Athletics - Riya
```

```
def filter_records(input_file, output_file):
    with open(input_file, 'r') as f_in:
        with open(output_file, 'w') as f_out:
            for line in f_in:
                event, participant = line.strip().split(' - ')
                if event == 'Athletics':
                    f_out.write(line)

filter_records('sports.dat', 'Athletic.dat')
```

Question 16

Write a program to count the words "to" and "the" present in a text file "Poem.txt".

Solution

Let the file "Poem.txt" include the following sample text:

```
To be or not to be, that is the question.
The quick brown fox jumps over the lazy dog.
To infinity and beyond!
The sun sets in the west.
To be successful, one must work hard.
```

```
to_count = 0
the_count = 0

with open("Poem.txt", 'r') as file:
    for line in file:
        words = line.split()
        for word in words:
            if word.lower() == 'to':
                to_count += 1
            elif word.lower() == 'the':
                the_count += 1

print("count of 'to': ", to_count)
print("count of 'the': ", the_count)
```

Question 17

Write a program to count the number of uppercase alphabets present in a text file "Poem.txt".

Solution

Let the file "Poem.txt" include the following sample text:

```
PYTHON is a Popular Programming Language.
```

```
with open("Poem.txt", 'r') as file:
    text = file.read()
    count = 0
    for char in text:
        if char.isupper():
            count += 1

print(count)
```

Question 18

Write a program that copies one file to another. Have the program read the file names from user.

Solution

```
def copy_file(file1, file2):
    with open(file1, 'r') as source:
        with open(file2, 'w') as destination:
            destination.write(source.read())

source_file = input("Enter the name of the source file: ")
destination_file = input("Enter the name of the destination file: ")

copy_file(source_file, destination_file)
```

Question 19

Write a program that appends the contents of one file to another. Have the program take the file names from the user.

Solution

```
def append_file(f1, f2):
    with open(f1, 'r') as source:
        with open(f2, 'a') as destination:
            destination.write(source.read())

source_file = input("Enter the name of the source file: ")
destination_file = input("Enter the name of the destination file: ")

append_file(source_file, destination_file)
```

Question 20

Write a program that reads characters from the keyboard one by one. All lower case characters get stored inside the file 'LOWER', all upper case characters get stored inside the file 'UPPER' and all other characters get stored inside file 'OTHERS'.

Solution

```
lower_file = open("LOWER.txt", 'w')
upper_file = open("UPPER.txt", 'w')
others_file = open("OTHERS.txt", 'w')
ans = 'y'
```

```
while ans == 'y':
    char = input("Enter a character: ")
    if char.islower():
        lower_file.write(char + "\n")
    elif char.isupper():
        upper_file.write(char + "\n")
    else:
        others_file.write(char + "\n")
    ans = input("Want to enter a character? (y/n): ")
lower_file.close()
upper_file.close()
others_file.close()
```

Question 21

Write a program to search the names and addresses of persons having age more than 30 in the data list of persons stored in a text file.

Solution

Let the file "Persons.txt" include the following sample text:

```
Samyukta, Mumbai, 35
Anubhav, Chennai, 28
Aniket, Hyderabad, 42
Sarth, Bangalore, 31
```

```
f = open('Persons.txt', 'r')
lines = f.readlines()
for line in lines:
    data = line.strip().split(',')
    if len(data) >= 3 and int(data[2]) > 30:
        print('Name:', data[0], 'Address:', data[1])
f.close()
```

Question 22

Write a function in Python to count and display the number of lines starting with alphabet 'A' present in a text file "LINES.TXT", e.g., the file "LINES.TXT" contains the following lines:

```
A boy is playing there.
There is a playground.
An aeroplane is in the sky.
A cricket match is being played.
```

The function should display the output as 3.

Solution

```
def count_lines(file_name):
    count = 0
    with open(file_name, 'r') as file:
        for line in file:
            if line.strip().startswith('A'):
                count += 1
    print(count)

```

```
count_lines("LINES.TXT")
```

Question 26

Write a program to accept string/sentences from the user till the user enters "END". Save the data in a text file and then display only those sentences which begin with an uppercase alphabet.

Solution

```
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()
```

Question 27

Write a function to insert a sentence in a text file, assuming that text file is very big and can't fit in computer's memory.

Solution

Let the file "insert.txt" include the following sample text:

```
Bees hum
leaves rustle
Waves crash
nature's voice whispers
```

```
def insert_sentence(file_path, sentence):
    file = open(file_path, 'a')
    file.write(sentence + '\n')
    file.close()
insert_sentence("insert.txt", "life's essence glimmers")
```

Question 28

Write a program to read a file 'Story.txt' and create another file, storing an index of 'Story.txt', telling which line of the file each word appears in. If word appears more than once, then index should show all the line numbers containing the word.

Solution

Let the file "Story.txt" include the following sample text:

```
The cat sleeps
The dog barks
The cat jumps
The sun shines
```

```
word_index = {}
file = open('Story.txt', 'r')
line_number = 1
lines = file.readlines()
for line in lines:
    words = line.strip().split()
    for word in words:
        if word in word_index:
            word_index[word].append(str(line_number))
        else:
            word_index[word] = [str(line_number)]
    line_number += 1
file.close()

index_file = open('index.txt', 'w')
for word, line_numbers in word_index.items():
    line_numbers_str = ", ".join(line_numbers)
    index_file.write(word + ":" + line_numbers_str + "\n")
index_file.close()
```

Question 29

Write a program to accept a filename from the user and display all the lines from the file which contain Python comment character '#'.

Solution

Let the file "notes.txt" include the following sample text:

```
Welcome to the Garden of Dreams
#where the ordinary becomes extraordinary
#the impossible becomes possible.
```

```
file_name = input("Enter the filename: ")
file = open(file_name, 'r')
lines = file.readlines()
for line in lines:
    if '#' in line:
        print(line)
file.close()
```

Question 31

Write a Python program to display the size of a file after removing EOL characters, leading and trailing white spaces and blank lines.

Solution

Let the file "sample.txt" include the following sample text:

```
The sky is blue\n
```

```
Clouds float gently in the sky.

Birds sing sweet melodies.
```

```
file = open('sample.txt', 'r')
lines = file.readlines()
original_size = 0
for line in lines:
    original_size += len(line)
print("Original file size:", original_size)
file.close()

file = open('sample.txt', 'r')
cleaned_size = 0
for line in file:
    cleaned_line = line.strip()
    if cleaned_line:
        cleaned_size += len(cleaned_line)
file.close()
print("Cleaned file size:", cleaned_size)
```

Question 32

Write a function Remove_Lowercase() that accepts two file names, and copies all lines that do not start with lowercase letter from the first file into the second file.

Solution

Let the file "file1.txt" include the following sample text:

```
Dew on petals, morning's gift.
silent moon, silver glow.
Winds whisper, secrets shared.
rain's embrace, earth's renewal.
```

```
def Remove_Lowercase(input_file, output_file):
    input_file = open(input_file, 'r')
    output_file = open(output_file, 'w')
    for line in input_file:
        if line.strip() and line[0].isupper():
            output_file.write(line)
    input_file.close()
    output_file.close()

input_file = 'file1.txt'
output_file = 'file2.txt'
Remove_Lowercase(input_file, output_file)
```

Question 33

Write a program to display all the records in a file along with line/record number.

Solution

Let the file "f1.txt" include the following sample text:

```
Soft whispers of the wind.
A melody in the trees.
Sun-kissed petals dance.
A garden's quiet song.
```

```
file_name = 'f1.txt'
line_number = 1
f = open(file_name, 'r')
for line in f:
    print("Line", line_number, ":", line.strip())
    line_number += 1
f.close()
```

Question 34

Write a method in Python to write multiple line of text contents into a text file "mylife.txt".

Solution

```
def write_to_file(file_path):
    lines_to_write = ["The sun sets over the horizon.", "Birds chirp in
        the morning.", "Raindrops patter on the roof.", "Leaves rustle in
        the breeze."]
    with open(file_path, "w") as file:
        for line in lines_to_write:
            file.write(line + '\n')

write_to_file("mylife.txt")
```

Question 35

Write a method in Python to read the content from a text file "DIARY.TXT" line by line and display the same on the screen.

Solution

```
def diary_content(f):
    myfile = open(f, "r")
    str = " "
    while str:
        str = myfile.readline()
        print(str, end = '')
    myfile.close()

diary_content("DIARY.TXT")
```

Question 36

Write appropriate statements to do the following:

(a) To open a file named "RESULT.DAT" for output.

(b) To go to the end of the file at any time.

Solution

(a) To open a file named "RESULT.DAT" for output:

```
file = open("RESULT.DAT")
```

(b) To go to the end of the file at any time:

```
file.seek(0, 2)
```

Question 37

Write a program to add two more employees' details (empno, ename, salary, designation) to the file "emp.txt" already stored in disk.

Solution

Let the file "emp.txt" include the following sample text:

```
1001, Ankit Singh, 50000, Manager
1002, Neha Patel, 45000, Developer
```

```
file = open("emp.txt", "a")
file.write("1003, Manoj Tiwari, 48000, Analyst\n")
file.write("1004, Aarti Gupta, 52000, Engineer\n")
```

Question 42

Anant has been asked to display all the students who have scored less than 40 for Remedial Classes. Write a user-defined function to display all those students who have scored less than 40 from the binary file "Student.dat".

Solution

Let the file "Student.dat" include the following sample data:

```
Radhika 80
Shaurya 35
Sonia 38
Anirudh 45
```

```
import pickle

def display_students(file_name):
        file = open(file_name, 'rb')
        students = pickle.load(file)
        for student in students:
            name = student[0]
            score = student[1]
            if score < 40:
                print(name)
        file.close()
display_students('student.dat')
```

Question 43

Following is the structure of each record in a data file named "PRODUCT.DAT".

`{"prod_code": value, "prod_desc": value, "stock": value}`

The values for prod_code and prod_desc are strings and the value for stock is an integer.

Write a function in Python to update the file with a new value of stock. The stock and the product_code, whose stock is to be updated, are to be inputted during the execution of the function.

Solution

Let the file "PRODUCT.dat" include the following sample data:

```
{'prod_code': 'AB', 'prod_desc': 'Books', 'stock': 50}
{'prod_code': 'AC', 'prod_desc': 'Pens', 'stock': 75}
{'prod_code': 'AD', 'prod_desc': 'Pencils', 'stock': 30}
```

```
import pickle

def update_stock(file_name, product_code, new_stock):
    products = []
    f = open(file_name, 'rb')
    while True:
        try:
            product = pickle.load(f)
            products.append(product)
        except EOFError:
            break
    f.close()
    updated = False
    for product in products:
        if product['prod_code'] == product_code:
            product['stock'] = new_stock
            updated = True
            break
    f = open(file_name, 'wb')
    for product in products:
       pickle.dump(product, f)
    f.close()
P_code = input("Enter the product code: ")
New = int(input("Enter the new stock value: "))
update_stock('PRODUCT.DAT', P_code, New)
```

Question 44

Given a binary file "STUDENT.DAT", containing records of the following type:

`[S_Admno, S_Name, Percentage]`

Where these three values are:

S_Admno — Admission Number of student (string)

S_Name — Name of student (string)

Percentage — Marks percentage of student (float)

Write a function in Python that would read contents of the file "STUDENT.DAT" and display the details of those students whose percentage is above 75.

Solution

Let the file "STUDENT.dat" include the following sample data:

```
[['101', 'Aishwarya', 97.0],
['102', 'Sakshi', 85.0],
['103', 'Prateek', 70.0]]
```

```
import pickle

def display_students(filename):
        file = open(filename, 'rb')
        student_records = pickle.load(file)
        above_75 = []
        for student in student_records:
            if student[2] > 75.0:
                above_75.append(student)
        if above_75:
            print("Students with percentage above 75:")
            for student in above_75:
                print("Admission Number:", student[0], "Name:",
                    student[1], "Percentage:", student[2])
        file.close()
display_students("STUDENT.DAT")
```

Question 50

Write a program to enter the following records in a binary file:

- Item No — integer
- Item_Name — string
- Qty — integer
- Price — float

Number of records to be entered should be accepted from the user. Read the file to display the records in the following format:

```
Item No:
Item Name:
Quantity:
Price per item:
Amount: (to be calculated as Price * Qty)
```

Solution

```
import pickle

with open("item.dat", 'wb') as itemfile:
    n = int(input("How many records to be entered? "))
    for i in range(n):
        ino = int(input("Enter item no: "))
        iname = input("Enter item name: ")
        qty = int(input("Enter quantity: "))
        price = float(input("Enter price: "))
        item = {"Item no": ino, "Item Name": iname, "Qty": qty, "Price":
            price}
```

```
        pickle.dump(item, itemfile)
    print("Successfully written item data")


with open("item.dat", "rb") as file:
    try:
        while True:
            item = pickle.load(file)
            print("\nItem No:", item["Item no"])
            print("Item Name:", item["Item Name"])
            print("Quantity:", item["Qty"])
            print("Price per item:", item["Price"])
            print("Amount:", item["Qty"] * item["Price"])
    except EOFError:
        pass
```

Question 51

Create a CSV file "Groceries" to store information of different items existing in a shop. The information is to be stored w.r.t. each item code, name, price, qty. Write a program to accept the data from user and store it permanently in CSV file.

Solution

```
import csv

with open("Groceries.csv", mode = 'w', newline = '') as file:
    writer = csv.writer(file)

    while True:
        item_code = int(input("Enter Item Code: "))
        name = input("Enter Name of the Item: ")
        price = float(input("Enter Price: "))
        quantity = int(input("Enter Quantity: "))

        writer.writerow([item_code, name, price, quantity])
        choice = input("Wish to enter more records (Y/N)?: ")
        if choice.upper() == 'N':
            break
```

4 Data Structures

Question 7

Write an interactive menu-driven program implementing Stack using list. The list is storing numeric data.

Solution

```
def push(num):
    h = int(input("Enter a number: "))
    num.append(h)

def pop(num):
    if len(num) == 0:
        print("No number to delete")
```

```
    else:
        print("Deleted number is:", num.pop())

def display(num):
    print(num)

num = []
while True:
    print("1. Add number")
    print("2. Delete number")
    print("3. Display numbers")
    print("4. Exit")
    op = int(input("Enter the Choice: "))
    if op == 1:
        push(num)
    elif op == 2:
        pop(num)
    elif op == 3:
        display(num)
    elif op == 4:
        print("Exiting program.")
        break
    else:
        print("Invalid choice. Please enter a number between 1 and 4.")
```

Question 8

Write an interactive menu-driven program to implement Stack using list. The list contains the names of students.

Solution

```
def push(student):
    h = input("Enter a name: ")
    student.append(h)

def pop(student):
    if len(student) == 0:
        print("No name to delete")
    else:
        print("Deleted name is:", student.pop())

def display(student):
    print(student)

student = []
while True:
    print("1. Add name")
    print("2. Delete name")
    print("3. Display names")
    print("4. Exit")
    op = int(input("Enter the Choice: "))
    if op == 1:
        push(student)
    elif op == 2:
        pop(student)
    elif op == 3:
```

```
        display(student)
    elif op == 4:
        print("Exiting program.")
        break
    else:
        print("Invalid choice. Please enter a number between 1 and 4.")
```

Question 10

Write a menu-driven Python program using queue to implement movement of shuttlecock in its box.

Solution

```
queue = []

def display_queue():
    if not queue:
        print("Queue is empty")
    else:
        print("Current queue:", queue)

def enqueue(item):
    queue.append(item)

def dequeue():
    if queue:
        removed = queue.pop(0)
        print("Removed", removed, "from the queue")
    else:
        print("Queue is empty, cannot remove")

while True:
    print("\nMENU:")
    print("1. Add movement to the queue")
    print("2. Remove movement from the queue")
    print("3. Display current queue")
    print("4. Exit")

    choice = input("Enter your choice (1-4): ")

    if choice == '1':
        movement = input("Enter the movement (up/down): ")
        enqueue(movement)
        print("Added", movement, "to the queue")
    elif choice == '2':
        dequeue()
    elif choice == '3':
        display_queue()
    elif choice == '4':
        print("Exiting program...")
        break
    else:
        print("Invalid choice, please try again")
```

Question 11

Give the necessary declaration of a list implemented Stack containing float type numbers. Also, write a user-defined function to pop a number from this Stack.

Solution

```
# Declaration for list implemented stack
Stack = list()

# Function body to pop a number from the stack
def pop_element(Stack):
    if not Stack:
        print("Stack is empty")
    else:
        Num = Stack.pop()
        print("Value deleted from stack is", Num)

Stack = [1.5, 2.7, 3.2, 4.9]
print("Initial Stack:", Stack)

pop_element(Stack)
print("Stack after pop operation:", Stack)
```

Question 12

A linear Stack called Directory contains the following information as contacts:

- Pin code of city
- Name of city

Write add(Directory) and delete(Directory) methods in Python to add and remove contacts using append() and pop() operations in Stack.

Solution

```
def add(Directory):
    pincode = int(input("Enter pincode of city: "))
    cityname = input("Enter the city name: ")
    contact = [pincode, cityname]
    Directory.append(contact)

def delete(Directory):
    if (Directory == []):
        print("Directory is empty")
    else:
        print("Deleted contact:", Directory.pop())

Directory = []
add(Directory)
add(Directory)
print("Initial Directory Stack:", Directory)

delete(Directory)
print("Directory Stack after deletion:", Directory)
```

Question 13

Write AddClient(Client) and DeleteClient(Client) methods in Python to add a new client and delete a client from a list client name, considering them to act as insert and delete operations of the Queue data structure.

Solution

```
def AddClient(client):
    client_name = input("Enter client name: ")
    client.append(client_name)

def DeleteClient(client):
    if client == []:
        print("Queue is empty")
    else:
        print("Deleted client:", client.pop(0))

client = []

AddClient(client)
AddClient(client)
AddClient(client)
print("Initial Client Queue:", client)

DeleteClient(client)
print("Client Queue after deletion:", client)
```

Question 15

Write Addscore(Game) and Delscore(Game) methods in Python to add new Score in the list of score in a game and remove a score from a list of score of a game considering these methods to act as PUSH and POP operation of data structure Stack.

Solution

```
def Addscore(Game):
    score = int(input("Enter the score to add: "))
    Game.append(score)

def Delscore(Game):
    if Game == []:
        print("Game's score list is empty")
    else:
        print("Deleted score:", Game.pop())

Game = []

Addscore(Game)
Addscore(Game)
Addscore(Game)
print("Initial Game Stack:", Game)

Delscore(Game)
print("Game Stack after deletion:", Game)
```

Question 16

Write a Python program to sort a Stack in ascending order without using an additional Stack.

Solution

```
def pushSorted(s, num):
    if len(s) == 0 or num > s[-1]:
        s.append(num)
        return
    else:
        t = s.pop()
        pushSorted(s, num)
        s.append(t)

def doSort(s):
    if len(s) != 0:
        t = s.pop()
        doSort(s)
        pushSorted(s, t)

s = []
s.append(12)
s.append(5)
s.append(-3)
s.append(4)
s.append(10)

print("Stack before sorting: ")
print(s)

doSort(s)

print("\nStack after sorting: ")
print(s)
```

Question 19

Write a program to create a Stack for storing only odd numbers out of all the numbers entered by the user. Display the content of the Stack along with the largest odd number in the Stack.

(Hint: Keep popping out the elements from Stack and maintain the largest element retrieved so far in a variable. Repeat till Stack is empty.)

Solution

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

def pop(stack):
    if stack == []:
        return None
    return stack.pop()

def oddStack(num):
    if num % 2 == 1:
        push(stack, num)

```

```
def GetLargest(stack):
    elem = pop(stack)
    large = elem
    while elem != None:
        if large < elem:
            large = elem
        elem = pop(stack)
    return large

n = int(input("How many numbers?"))
stack = []
for i in range(n):
    number = int(input("Enter number:"))
    oddStack(number)
print("Stack created is", stack)
bigNum = GetLargest(stack)
print("Largest number in stack", bigNum)
```

Question 21

Write a program that, depending upon the user's choice, either pushes or pops an element in a Stack. The elements are shifted towards the right so that the top always remains at 0th (zeroth) index.

Solution

```
def push(stack, element):
    stack.insert(0, element)

def pop(stack):
    if stack == []:
        print("Stack is empty")
    else:
        print("Popped element:", stack.pop(0))

stack = []

while True:
    print("STACK OPERATIONS")
    print("1. Push element")
    print("2. Pop element")
    print("3. Exit")
    choice = input("Enter your choice (1-3): ")

    if choice == '1':
        element = input("Enter the element to push: ")
        push(stack, element)
    elif choice == '2':
        pop(stack)
    elif choice == '3':
        print("Exiting program")
        break
    else:
        print("Invalid choice. Please enter a valid option.")
```