

LOTUS VALLEY INTERNATIONAL SCHOOL: Computer Science Practical File

Pranav Verma (Roll No. 7, XII Aryabhatta)

November 1, 2025



Contents

1 Review of Python Basics	2
2 Functions	12
3 File Handling	17
4 Relational Databases and SQL	28

1 Review of Python Basics

Question 15

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

Solution

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

Question 16

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

Solution

```
1 l = eval(input("Enter a list: "))
2 m = min(l)
3 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

```
1 marks_list = eval(input("Enter list of marks: "))
2 total_marks = sum(marks_list)
3 total_subjects = len(marks_list)
4 maximum_marks_per_subject = 100
5 total_marks_possible = maximum_marks_per_subject * total_subjects
6 percentage = (total_marks / total_marks_possible) * 100
7
8 print("Total Marks:", total_marks)
9 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

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

Question 19

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

Solution

```
1 my_list = eval(input("Enter the list: "))
2 c = int(input("Enter the element whose frequency is to be checked: "))
3 frequency = my_list.count(c)
4 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

```
1 l = eval(input("Enter the list: "))
2 print("Original List")
3 print(l)
4
5 l = l[-1:] + l[:-1]
6
7 print("Rotated List")
8 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

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

```
1 tuple1 = eval(input("Enter a tuple: "))
2 tuple2 = (10, 20, 30)
3 combined_tuple = tuple1 + tuple2
4 print("Elements of the combined tuple:")
5 for element in combined_tuple:
6     print(element)
7
8 print("Maximum value:", max(combined_tuple))
9 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

```
1 tuple1 = eval(input("Enter the first tuple: "))
2 tuple2 = eval(input("Enter the second tuple: "))
3
4 print("Original Tuples:")
5 print("Tuple 1:", tuple1)
6 print("Tuple 2:", tuple2)
7
8 tuple1, tuple2 = tuple2, tuple1
9
10 print("\nSwapped Tuples:")
11 print("Tuple 1 (after swapping):", tuple1)
12 print("Tuple 2 (after swapping):", tuple2)
13
14 if tuple1 == tuple2:
15     print("\nThe swapped tuples are equal.")
16 else:
17     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

```
1 n = int(input("Enter number of classes: "))
2 data = {}
3 for i in range(n):
4     class_name = input("Enter class name: ")
5     teacher_name = input("Enter teacher name: ")
6     data[class_name] = teacher_name
7 print("Class data:", data)
8 find = input("Enter a class name to find its teacher: ")
9 if find in data:
10    print("Teacher for class", find, "is", data[find])
11 else:
12    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

```
1 n = int(input("Enter number of students: "))
2 data = {}
3 for i in range(n):
4     stu_name = input("Enter student name: ")
5     percentage = input("Enter percentage: ")
6     data[stu_name] = percentage
7 print("Student data:", data)
8 find = input("Enter a student name to delete: ")
9 if find in data:
10    del data[find]
11    print("Updated student data:", data)
12 else:
13    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

```
1 n = int(input("Enter the number of customers: "))
2 customer_data = {}
3
4 for i in range(n):
5     name = input("Enter customer name: ")
6     items_bought = input("Enter items bought: ")
7     cost = float(input("Enter cost: "))
8     phone_number = int(input("Enter phone number: "))
9
10    customer_data[name] = {
11        'Items Bought': items_bought,
12        'Cost': cost,
13        'Phone Number': phone_number
14    }
```

```
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

```
1 input_string = input("Enter the string: ")
2 words = input_string.split()
3
4 word_counts = {}
5 for word in words:
6     if word in word_counts:
7         word_counts[word] += 1
8     else:
9         word_counts[word] = 1
10
11 max_count = 0
12 second_max_count = 0
13 most_repeated_word = None
14 second_most_repeated_word = None
15
16 for word, count in word_counts.items():
17     if count > max_count:
18         second_max_count = max_count
19         max_count = count
20         second_most_repeated_word = most_repeated_word
21         most_repeated_word = word
22     elif count > second_max_count:
23         second_max_count = count
24         second_most_repeated_word = word
25
26 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

```
1 input_str = input("Enter the string: ")
2 first_char = input_str[0]
3 last_char = input_str[-1]
4 middle_chars = input_str[1:-1]
5 new_str = last_char + middle_chars + first_char
6 print("Original string:", input_str)
7 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

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

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

Question 33

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

Solution

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

Question 34

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

Solution

```
1 list1 = eval(input("Enter the first list: "))
2 list2 = eval(input("Enter the second list: "))
3 list1.extend(list2)
4 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

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

Question 36

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

Solution

```
1 input_list = eval(input("Enter the list: "))
2 unique_values = []
3
4 for item in input_list:
5     if item not in unique_values:
6         unique_values.append(item)
7
8 print("Unique values from the list:", unique_values)
```

Question 37

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

Solution

```
1 string = input("Enter the string: ")
2 char_list = list(string)
3
4 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

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

Question 39

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

Solution

```
1 my_dict = eval(input("Enter the dictionary: "))
2 key_check = input("Enter the key to be checked: ")
3 if key_check in my_dict:
4     print("The key", key_check, "exists in the dictionary.")
5 else:
6     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

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

Question 41

Write a Python script to merge two Python dictionaries.

Solution

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

Question 42

Write a Python program to sort a dictionary by key.

Solution

```
1 def bubble_sort_keys(keys):
2     n = len(keys)
3     for i in range(n - 1):
4         for j in range(0, n - i - 1):
5             if keys[j] > keys[j + 1]:
6                 keys[j], keys[j + 1] = keys[j + 1], keys[j]
7
8 my_dict = eval(input("Enter the dictionary: "))
9 keys_list = list(my_dict.keys())
10 bubble_sort_keys(keys_list)
11 sorted_dict = {}
12 for key in keys_list:
13     sorted_dict[key] = my_dict[key]
14
15 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

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

```
4 for key, value in d1.items():
5     combined_dict[key] = value
6 for key, value in d2.items():
7     if key in combined_dict:
8         combined_dict[key] += value
9     else:
10        combined_dict[key] = value
11
12 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

```
1 my_dict = eval(input("Enter the dictionary: "))
2 highest_values = []
3 highest_keys = []
4 # This is not the most efficient way, but it matches the provided code.
5 # A better way would be to sort items by value.
6 for key, value in my_dict.items():
7     if not highest_values or value > highest_values[-1]:
8         highest_values.append(value)
9         highest_keys.append(key)
10        if len(highest_values) > 3:
11            highest_values.pop(0)
12            highest_keys.pop(0)
13        # Note: This logic is flawed as it doesn't handle insertion correctly.
14        # A correct implementation would be more complex or use sorting.
15        # Re-implementing based on sorting for correctness, but keeping
16        # original logic:
17
18 # Sticking to the provided logic exactly:
19 my_dict = eval(input("Enter the dictionary: "))
20 highest_values = []
21 highest_keys = []
22 for key, value in my_dict.items():
23     if not highest_values or value > highest_values[-1]:
24         highest_values.append(value)
25         highest_keys.append(key)
26         if len(highest_values) > 3:
27             highest_values.pop(0)
28             highest_keys.pop(0)
29 print("Three highest values in the dictionary:")
30 for i in range(len(highest_keys)):
31     print(highest_keys[i], ":" ,highest_values[i])
```

Question 45

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

Solution

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

```
3     if isinstance(value, list):
4         value.sort()
5
6 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

```
1 my_dict = eval(input("Enter the dictionary: "))
2 total_count = 0
3 for value in my_dict.values():
4     if type(value) is list:
5         total_count += len(value)
6 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 = $1/2 * \text{base} * \text{height}$

Solution

```
1 def calculate_area(base, height):
2     area = (1/2) * base * height
3     return area
4
5 base_value = int(input("Enter the base value: "))
6 height_value = int(input("Enter the height value: "))
7 triangle_area = calculate_area(base_value, height_value)
8 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

```
1 def calculate_area(base, height, shape_type):
2     if shape_type == "triangle":
3         area = (1/2) * base * height
4     elif shape_type == "rectangle":
5         area = base * height
6     else:
7         area = None
8     print("Invalid shape type. Please specify either 'triangle' or
9         'rectangle'.")
9 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

```
1 def calculate_volume(length = 5, width = 3, height = 2):
2     return length * width * height
3
4 default_volume = calculate_volume()
5 print("Volume of the box with default values:", default_volume)
6
7 v = calculate_volume(10, 7, 15)
8 print("Volume of the box with default values:", v)
9
10 a = calculate_volume(length = 23, height = 6)
11 print("Volume of the box with default values:", a)
12
13 b = calculate_volume(width = 19)
14 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

```
1 def display_multiples(n, count = 1):
2     if count > 4:
3         return
4     print(n * count)
5     display_multiples(n, count + 1)
6
7 n = int(input("Enter the number: "))
8 display_multiples(n)
```

Question 22

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

$$1 + 1/2 - 1/3 + 1/4 - 1/5 \dots$$

Solution

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

```
6     else:
7         return add_series_terms(n - 1) - 1 / n
8
9 n = int(input("Enter the term: "))
10 print(add_series_terms(n))
```

Question 23

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

Solution

```
1 def gcd(a, b):
2     while b:
3         a, b = b, a % b
4     return a
5
6 num1 = int(input("Enter the first number: "))
7 num2 = int(input("Enter the second number: "))
8 gcd_value = gcd(num1, num2)
9 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

```
1 def multiply_list(numbers):
2     product = 1
3     for num in numbers:
4         product *= num
5     return product
6
7 numbers = eval(input("Enter the list: "))
8 product = multiply_list(numbers)
9 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

```
1 def factorial(n):
2     product = 1
3     for i in range(1, n + 1):
4         product *= i
5     return product
6
```

```
7 n = int(input("Enter a number: "))
8 fact = factorial(n)
9 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

```
1 def is_prime(n):
2     if n <= 1:
3         return False
4     factors = 0
5     for i in range(1, n + 1):
6         if n % i == 0:
7             factors += 1
8         if factors > 2:
9             return False
10    return True
11
12 num = int(input("Enter a number: "))
13 is_prime_num = is_prime(num)
14 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

```
1 def is_palindrome(s):
2     s = s.lower()
3     return s == s[::-1]
4
5 input_string = input("Enter a string: ")
6 if is_palindrome(input_string):
7     print(input_string, "is a palindrome.")
8 else:
9     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

```
1 def sort_words(s):
2     words = s.split(' ')
3     words.sort()
4     return ' '.join(words)
5
6 input_str = input('Enter a hyphen-separated sequence of words: ')
7 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

```
1 fileout = open('STRS.txt', 'w')
2 ans = 'y'
3 while ans == 'y':
4     string = input("Enter a string: ")
5     fileout.write(string + "\n")
6     ans = input("Want to enter more strings?(y/n)... ")
7 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.
```

```
1 with open("input.txt", 'r') as f:
2     with open("output.txt", 'w') as fout:
3         for line in f:
4             modified_line = ' '.join(line.split())
5             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
```

```
1 def filter_records(input_file, output_file):  
2     with open(input_file, 'r') as f_in:  
3         with open(output_file, 'w') as f_out:  
4             for line in f_in:  
5                 event, participant = line.strip().split(' - ')  
6                 if event == 'Athletics':  
7                     f_out.write(line)  
8  
9 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.
```

```
1 to_count = 0  
2 the_count = 0  
3  
4 with open("Poem.txt", 'r') as file:  
5     for line in file:  
6         words = line.split()  
7         for word in words:  
8             if word.lower() == 'to':  
9                 to_count += 1  
10            elif word.lower() == 'the':  
11                the_count += 1  
12  
13 print("count of 'to': ", to_count)  
14 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.

```
1 with open("Poem.txt", 'r') as file:  
2     text = file.read()  
3     count = 0  
4     for char in text:  
5         if char.isupper():  
6             count += 1  
7  
8 print(count)
```

Question 18

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

Solution

```
1 def copy_file(file1, file2):  
2     with open(file1, 'r') as source:  
3         with open(file2, 'w') as destination:  
4             destination.write(source.read())  
5  
6 source_file = input("Enter the name of the source file: ")  
7 destination_file = input("Enter the name of the destination file: ")  
8  
9 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

```
1 def append_file(f1, f2):  
2     with open(f1, 'r') as source:  
3         with open(f2, 'a') as destination:  
4             destination.write(source.read())  
5  
6 source_file = input("Enter the name of the source file: ")  
7 destination_file = input("Enter the name of the destination file: ")  
8  
9 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

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

```

5 while ans == 'y':
6     char = input("Enter a character: ")
7     if char.islower():
8         lower_file.write(char + "\n")
9     elif char.isupper():
10        upper_file.write(char + "\n")
11    else:
12        others_file.write(char + "\n")
13    ans = input("Want to enter a character? (y/n): ")
14 lower_file.close()
15 upper_file.close()
16 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

```

```

1 f = open('Persons.txt', 'r')
2 lines = f.readlines()
3 for line in lines:
4     data = line.strip().split(',')
5     if len(data) >= 3 and int(data[2]) > 30:
6         print('Name:', data[0], 'Address:', data[1])
7 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

```

1 def count_lines(file_name):
2     count = 0
3     with open(file_name, 'r') as file:
4         for line in file:
5             if line.strip().startswith('A'):
6                 count += 1
7     print(count)
8

```

```
9 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

```
1 f = open("new.txt", "w")
2 while True:
3     st = input("Enter next line:")
4     if st == "END":
5         break
6     f.write(st + '\n')
7 f.close()
8
9 f = open("new.txt", "r")
10 while True:
11     st = f.readline()
12     if not st:
13         break
14     if st[0].isupper():
15         print(st)
16 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
```

```
1 def insert_sentence(file_path, sentence):
2     file = open(file_path, 'a')
3     file.write(sentence + '\n')
4     file.close()
5 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
```

```
1 word_index = []  
2 file = open('Story.txt', 'r')  
3 line_number = 1  
4 lines = file.readlines()  
5 for line in lines:  
6     words = line.strip().split()  
7     for word in words:  
8         if word in word_index:  
9             word_index[word].append(str(line_number))  
10        else:  
11            word_index[word] = [str(line_number)]  
12    line_number += 1  
13 file.close()  
14  
15 index_file = open('index.txt', 'w')  
16 for word, line_numbers in word_index.items():  
17     line_numbers_str = ", ".join(line_numbers)  
18     index_file.write(word + ":" + line_numbers_str + "\n")  
19 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.
```

```
1 file_name = input("Enter the filename: ")  
2 file = open(file_name, 'r')  
3 lines = file.readlines()  
4 for line in lines:  
5     if '#' in line:  
6         print(line)  
7 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.

```
1 file = open('sample.txt', 'r')
2 lines = file.readlines()
3 original_size = 0
4 for line in lines:
5     original_size += len(line)
6 print("Original file size:", original_size)
7 file.close()

8
9 file = open('sample.txt', 'r')
10 cleaned_size = 0
11 for line in file:
12     cleaned_line = line.strip()
13     if cleaned_line:
14         cleaned_size += len(cleaned_line)
15 file.close()
16 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.

```
1 def Remove_Lowercase(input_file, output_file):
2     input_file = open(input_file, 'r')
3     output_file = open(output_file, 'w')
4     for line in input_file:
5         if line.strip() and line[0].isupper():
6             output_file.write(line)
7     input_file.close()
8     output_file.close()
9
10 input_file = 'file1.txt'
11 output_file = 'file2.txt'
12 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.

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

Question 34

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

Solution

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

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

```
1 def diary_content(f):  
2     myfile = open(f, "r")  
3     str = ""  
4     while str:  
5         str = myfile.readline()  
6         print(str, end = '')  
7     myfile.close()  
8  
9 diary_content("DIARY.TXT")
```

Question 36

Write appropriate statements to do the following:

- To open a file named “RESULT.DAT” for output.
- To go to the end of the file at any time.

Solution

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

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

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

```
1 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
```

```
1 file = open("emp.txt", "a")  
2 file.write("1003, Manoj Tiwari, 48000, Analyst\n")  
3 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
```

```
1 import pickle  
2  
3 def display_students(file_name):  
4     file = open(file_name, 'rb')  
5     students = pickle.load(file)  
6     for student in students:  
7         name = student[0]  
8         score = student[1]  
9         if score < 40:  
10             print(name)  
11     file.close()  
12 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}
```

```
1 import pickle  
2  
3 def update_stock(file_name, product_code, new_stock):  
4     products = []  
5     f = open(file_name, 'rb')  
6     while True:  
7         try:  
8             product = pickle.load(f)  
9             products.append(product)  
10        except EOFError:  
11            break  
12    f.close()  
13    updated = False  
14    for product in products:  
15        if product['prod_code'] == product_code:  
16            product['stock'] = new_stock  
17            updated = True  
18            break  
19    f = open(file_name, 'wb')  
20    for product in products:  
21        pickle.dump(product, f)  
22    f.close()  
23 P_code = input("Enter the product code: ")  
24 New = int(input("Enter the new stock value: "))  
25 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]]
```

```
1 import pickle  
2  
3 def display_students(filename):  
4     file = open(filename, 'rb')  
5     student_records = pickle.load(file)  
6     above_75 = []  
7     for student in student_records:  
8         if student[2] > 75.0:  
9             above_75.append(student)  
10    if above_75:  
11        print("Students with percentage above 75:")  
12        for student in above_75:  
13            print("Admission Number:", student[0], "Name:",  
14                student[1], "Percentage:", student[2])  
15    file.close()  
16 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

```
1 import pickle  
2  
3 with open("item.dat", 'wb') as itemfile:  
4     n = int(input("How many records to be entered? "))  
5     for i in range(n):  
6         ino = int(input("Enter item no: "))  
7         iname = input("Enter item name: ")  
8         qty = int(input("Enter quantity: "))  
9         price = float(input("Enter price: "))  
10        item = {"Item no": ino, "Item Name": iname, "Qty": qty, "Price":  
11            price}
```

```

11         pickle.dump(item, itemfile)
12     print("Successfully written item data")
13
14
15 with open("item.dat", "rb") as file:
16     try:
17         while True:
18             item = pickle.load(file)
19             print("\nItem No:", item["Item no"])
20             print("Item Name:", item["Item Name"])
21             print("Quantity:", item["Qty"])
22             print("Price per item:", item["Price"])
23             print("Amount:", item["Qty"] * item["Price"])
24     except EOFError:
25         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

```

1 import csv
2
3 with open("Groceries.csv", mode = 'w', newline = '') as file:
4     writer = csv.writer(file)
5
6     while True:
7         item_code = int(input("Enter Item Code: "))
8         name = input("Enter Name of the Item: ")
9         price = float(input("Enter Price: "))
10        quantity = int(input("Enter Quantity: "))
11
12        writer.writerow([item_code, name, price, quantity])
13        choice = input("Wish to enter more records (Y/N)?: ")
14        if choice.upper() == 'N':
15            break

```

4 Relational Databases and SQL

Question 17

Write SQL queries to perform the following based on the table PRODUCT having fields as (prod_id, prod_name, quantity, unit_rate, price, city)

- (i) Display those records from table PRODUCT where prod_id is more than 100.
- (ii) List records from table PRODUCT where prod_name is 'Almirah'.
- (iii) List all those records whose price is between 200 and 500.
- (iv) Display the product names whose price is less than the average of price.
- (v) Show the total number of records in the table PRODUCT.

Solution

(i)

```
1 SELECT * FROM PRODUCT  
2 WHERE prod_id > 100;
```

(ii)

```
1 SELECT * FROM PRODUCT  
2 WHERE prod_name = 'Almirah';
```

(iii)

```
1 SELECT * FROM PRODUCT  
2 WHERE price BETWEEN 200 AND 500;
```

(iv)

```
1 SELECT prod_name  
2 FROM PRODUCT  
3 WHERE price < (SELECT AVG(price) FROM PRODUCT);
```

(v)

```
1 SELECT COUNT(*) AS total_records FROM PRODUCT;
```

Question 20

Consider the following EMP and DEPT tables:

Table: EMP

EmpNo	EmpName	City	Designation	DOJ	Sal	Comm	DeptID
8369	SMITH	Mumbai	CLERK	1990-12-18	800.00	NULL	20
8499	ANYA	Varanasi	SALESMAN	1991-02-20	1600.00	300.00	30
8521	SETH	Jaipur	SALESMAN	1991-02-22	1250.00	500.00	30
8566	MAHADEVAN	Delhi	MANAGER	1991-04-02	2985.00	NULL	20

Table: DEPT

DeptID	DeptName	MgrID	Location
10	SALES	8566	Mumbai
20	PERSONNEL	9698	Delhi
30	ACCOUNTS	4578	Delhi
40	RESEARCH	8839	Bengaluru

Write the SQL command to get the following:

- Show the minimum, maximum and average salary of managers.
- Count the number of clerks in the organization.
- Display the designation-wise list of employees with name, salary and date of joining.
- Count the number of employees who are not getting commission.
- Show the average salary for all departments having salary > 2000 .

- (f) List the count of employees grouped by DeptID.
- (g) Display the maximum salary of employees in each department.
- (h) Display the name of employees along with their designation and department name.
- (i) Count the number of employees working in ACCOUNTS department.

Solution

(a)

```

1 SELECT MIN(Sal) AS MinSalary, MAX(Sal) AS MaxSalary, AVG(Sal) AS AvgSalary
2 FROM EMP
3 WHERE Designation = 'MANAGER';

```

(b)

```

1 SELECT COUNT(*) AS ClerkCount
2 FROM EMP
3 WHERE Designation = 'CLERK';

```

(c)

```

1 SELECT Designation, EmpName, Sal, DOJ
2 FROM EMP
3 ORDER BY Designation;

```

(d)

```

1 SELECT COUNT(*) AS No_comm
2 FROM EMP
3 WHERE comm is NULL;

```

(e)

```

1 SELECT D.DeptName, AVG(E.Sal) AS AvgSalary
2 FROM EMP E, DEPT D
3 WHERE E.DeptID = D.DeptID AND E.Sal > 2000
4 GROUP BY D.DeptName;

```

(f)

```

1 SELECT DeptID, COUNT(*) AS EmpCount
2 FROM EMP
3 GROUP BY DeptID;

```

(g)

```

1 SELECT D.DeptName, MAX(E.Sal) AS MaxSalary
2 FROM EMP E, DEPT D
3 WHERE E.DeptID = D.DeptID
4 GROUP BY D.DeptName;

```

(h)

```

1 SELECT EMP.EMPNAME, EMP.DESIGNATION, DEPT.DEPTNAME
2 FROM EMP, DEPT
3 WHERE EMP.DEPTID = DEPT.DEPTID;

```

(i)

```
1 SELECT COUNT(*) AS NUM_EMP
2 FROM EMP, DEPT
3 WHERE EMP.DEPTID = DEPT.DEPTID
4 AND DEPTNAME = 'ACCOUNTS';
```

Question 21

Write SQL commands for (i) to (vi) and write output for (vii) on the basis of the following PRODUCTS relation given below:

Table: PRODUCTS

PCODE	PNAME	COMPANY	PRICE	STOCK	MANUFACTURE	WARRANTY
P001	TV	BPL	10000	200	2018-01-12	3
P002	TV	SONY	12000	150	2017-03-23	4
P003	PC	LENOVO	39000	100	2018-04-09	2
P004	PC	COMPAQ	38000	120	2019-06-20	2
P005	HANDYCAMS	SONY	18000	250	2017-03-23	3

- (i) To show details of all PCs with stock more than 110.
- (ii) To list the company which gives warranty of more than 2 years.
- (iii) To find stock value of the BPL company where stock value is the sum of the products of price and stock.
- (iv) To show number of products from each company.
- (v) To count the number of PRODUCTS which shall be out of warranty on 20-NOV-2020.
- (vi) To show the average price of all products.
- (vii) Give the output of the following statements:
 - (a) SELECT COUNT(DISTINCT COMPANY) FROM PRODUCTS;
 - (b) SELECT MAX(PRICE) FROM PRODUCTS WHERE WARRANTY >= 3;

Solution

(i)

```
1 SELECT *
2 FROM PRODUCTS
3 WHERE PNAME = 'PC' AND STOCK > 110;
```

(ii)

```
1 SELECT DISTINCT COMPANY
2 FROM PRODUCTS
3 WHERE WARRANTY > 2;
```

(iii)

```
1 SELECT COMPANY, SUM(PRICE * STOCK) AS StockValue
2 FROM PRODUCTS
3 WHERE COMPANY = 'BPL';
```

(iv)

```
1 SELECT COMPANY , COUNT(*) AS ProductCount
2 FROM PRODUCTS
3 GROUP BY COMPANY;
```

(v)

```
1 SELECT COUNT(*)
2 FROM PRODUCTS
3 WHERE DATE_ADD(MANUFACTURE , INTERVAL WARRANTY YEAR) <= '2020-11-20' ;
```

(vi)

```
1 SELECT AVG(PRICE) FROM PRODUCTS ;
```

(vii)

(a)

```
1 SELECT COUNT(DISTINCT COMPANY) FROM PRODUCTS ;
```

(b)

```
1 SELECT MAX(PRICE) FROM PRODUCTS WHERE WARRANTY <= 3 ;
```

Question 22

Write SQL commands for (i) to (vi) on the basis of relations given below:

Table: BOOKS

Book_ID	Book_name	Author_name	Publishers	Price	Type	qty
K0001	Let us C	Y. Kanetkar	EPB	450	Prog	15
P0001	Computer Networks	B. Agarwal	FIRST PUBL	755	Comp	24
M0001	Mastering C++	K.R. Venugopal	EPB	165	Prog	60
N0002	VC++ advance	P. Purohit	TDH	250	Prog	45
K0002	Programming with Python	Sanjeev	FIRST PUBL	350	Prog	30
L02	Computer Science with Python	Sumita Arora	Dhanpat rai	655	Prog	16
L04	Computer Science with Python	Preeti Arora	Sultan chand	550	Prog	20
L05	Concise Mathematics	R.K.Bansal	Selina	600	Maths	10

Table: ISSUED

Book_ID	Qty_Issued
L02	13
L04	5
L05	21

- (i) To show the books of "FIRST PUBL" Publishers written by P.Purohit.
- (ii) To display cost of all the books published for FIRST PUBL.
- (iii) Depreciate the price of all books of EPB publishers by 5%.
- (iv) To display the bookname and price of the books, more than 3 copies of which have been issued.
- (v) To show total cost of books of each type.
- (vi) To show the details of the costliest book.

Solution

(i)

```
1 SELECT BOOK_NAME
2 FROM BOOKS
3 WHERE PUBLISHERS = "FIRST PUBL" AND AUTHOR_NAME = "P.PUROHIT";
```

(ii)

```
1 SELECT SUM(Price * qty) AS Cost
2 FROM BOOKS
3 WHERE Publishers = 'FIRST PUBL'
4 GROUP BY publishers;
```

(iii)

```
1 UPDATE BOOKS
2 SET Price = Price - (Price * 5/100)
3 WHERE Publishers = 'EPB';
```

(iv)

```
1 SELECT b.Book_name, b.Price
2 FROM BOOKS b, ISSUED i
3 WHERE b.Book_ID = i.Book_ID and i.Qty_Issued > 3;
```

(v)

```
1 SELECT Type, SUM(Price * qty) AS Total_Cost
2 FROM BOOKS
3 GROUP BY Type;
```

(vi)

```
1 SELECT *
2 FROM BOOKS
3 WHERE Price = (SELECT MAX(Price) FROM BOOKS);
```

Question 27

Write SQL Commands/output for the following on the basis of the given table:

Table: FURNITURE

NO	ITEM	TYPE	DATEOFSTOCK	PRICE	DISCOUNT
1	White Lotus	DoubleBed	2002-02-23	3000	25
2	Pinkfeathers	BabyCot	2002-01-29	7000	20
3	Dolphin	BabyCot	2002-02-19	9500	20
4	Decent	OfficeTable	2002-02-01	25000	30
5	Comfortzone	DoubleBed	2002-02-12	25000	30
6	Donald	BabyCot	2002-02-24	6500	15

- (i) To list the details of furniture whose price is more than 10000.
- (ii) To list the Item name and Price of furniture whose discount is between 10 and 20.
- (iii) To delete the record of all items where discount is 30.

- (iv) To display the price of 'Babycot'.
- (v) To list item name, type and price of all items whose names start with 'D'.
- (vi) SELECT DISTINCT Type FROM Furniture;
- (vii) SELECT MAX(PRICE) FROM Furniture WHERE DateofStock > '2002-02-15' ;
- (viii) SELECT COUNT(*) FROM Furniture WHERE Discount < 25 ;

Solution

(i)

```

1 SELECT *
2 FROM FURNITURE
3 WHERE PRICE > 10000;

```

(ii)

```

1 SELECT ITEM , PRICE
2 FROM FURNITURE
3 WHERE DISCOUNT BETWEEN 10 AND 20;

```

(iii)

```

1 DELETE FROM FURNITURE WHERE DISCOUNT = 30;

```

(iv)

```

1 SELECT PRICE
2 FROM FURNITURE
3 WHERE TYPE = 'BabyCot';

```

(v)

```

1 SELECT ITEM , TYPE , PRICE
2 FROM FURNITURE
3 WHERE ITEM LIKE 'D%';

```

(vi)

```

1 SELECT DISTINCT Type FROM Furniture;

```

(vii)

```

1 SELECT MAX(PRICE) FROM Furniture WHERE DateofStock > '2002-02-15' ;

```

(viii)

```

1 SELECT COUNT(*) FROM Furniture WHERE Discount < 25 ;

```

Question 28

Write SQL Commands/output for the following on the basis of the given table GRADUATE:

Table: GRADUATE

S.No.	NAME	STIPEND	SUBJECT	AVERAGE	RANK
1	KARAN	400	PHYSICS	68	1
2	RAJ	450	CHEMISTRY	68	1
3	DEEP	300	MATHS	62	2
4	DIVYA	350	CHEMISTRY	63	1
5	GAURAV	500	PHYSICS	70	1
6	MANAV	400	CHEMISTRY	55	2
7	VARUN	250	MATHS	64	1
8	LIZA	450	COMPUTER	68	1
9	PUJA	500	PHYSICS	62	1
10	NISHA	300	COMPUTER	57	2

- (i) List the names of those students who have obtained rank 1 sorted by NAME.
- (ii) Display a list of all those names whose AVERAGE is greater than 65.
- (iii) Display the names of those students who have opted COMPUTER as a SUBJECT with an AVERAGE of more than 60.
- (iv) List the names of all the students in alphabetical order.
- (v) SELECT * FROM GRADUATE WHERE NAME LIKE "%I%";
- (vi) SELECT DISTINCT RANK FROM GRADUATE;

Solution

(i)

```

1 SELECT NAME
2 FROM GRADUATE
3 WHERE 'RANK' = 1
4 ORDER BY NAME;

```

(ii)

```

1 SELECT NAME
2 FROM GRADUATE
3 WHERE AVERAGE > 65;

```

(iii)

```

1 SELECT NAME
2 FROM GRADUATE
3 WHERE SUBJECT = 'COMPUTER' AND AVERAGE > 60;

```

(iv)

```

1 SELECT NAME
2 FROM GRADUATE
3 ORDER BY NAME;

```

(v)

```

1 SELECT * FROM GRADUATE WHERE NAME LIKE "%I%";

```

(vi)

```

1 SELECT DISTINCT 'RANK' FROM GRADUATE;

```

Question 29(c)

For the given table, do as directed:

Table: STUDENT

ColumnName	Data type	size	Constraint
ROLLNO	Integer	4	Primary Key
SNAME	Varchar	25	Not Null
GENDER	Char	1	Not Null
DOB	Date		Not Null
FEES	Integer	4	Not Null
HOBBY	Varchar	15	Null

- (i) Write SQL query to create the table.
- (ii) Write SQL query to increase the size of SNAME to hold 30 characters.
- (iii) Write SQL query to remove the column HOBBY.
- (iv) Write SQL query to insert a row in the table with any values of your choice that can be accommodated there.

Solution

(i)

```
1 CREATE TABLE STUDENT(
2     ROLLNO INT(4) PRIMARY KEY,
3     SNAME VARCHAR(25) NOT NULL,
4     GENDER CHAR(1) NOT NULL,
5     DOB DATE NOT NULL,
6     FEES INT(4) NOT NULL,
7     HOBBY VARCHAR(15)
8 );
```

(ii)

```
1 ALTER TABLE STUDENT MODIFY SNAME VARCHAR(30);
```

(iii)

```
1 ALTER TABLE STUDENT DROP HOBBY;
```

(iv)

```
1 INSERT INTO STUDENT(ROLLNO, SNAME, GENDER, DOB, FEES, HOBBY)
2 VALUES (1, 'ANANYA', 'F', '2000-01-01', 5000, 'COOKING');
```

Question 30

Write SQL queries based on the following tables:

Table: PRODUCT

P_ID	ProductName	Manufacturer	Price	Discount
TP01	Talcum Powder	LAK	40	NULL
FW05	Face Wash	ABC	45	5
BS01	Bath Soap	ABC	55	NULL
5H06	Shampoo	XYZ	120	10
FW12	Face Wash	XYZ	95	NULL

Table: CLIENT

C_ID	ClientName	City	P_ID
01	Cosmetic Shop	Delhi	TP01
02	Total Health	Mumbai	FW05
03	Live Life	Delhi	BS01
04	Pretty Woman	Delhi	5H06
05	Dreams	Delhi	FW12

- (i) Write SQL Query to display ProductName and Price for all products whose Price is in the range 50 to 150.
- (ii) Write SQL Query to display details of products whose manufacturer is either XYZ or ABC.
- (iii) Write SQL query to display ProductName, Manufacturer and Price for all products that are not giving any discount.
- (iv) Write SQL query to display ProductName and price for all products whose ProductName ends with 'h'.
- (v) Write SQL query to display ClientName, City, P_ID and ProductName for all clients whose city is Delhi.
- (vi) Which column is used as Foreign Key and name the table where it has been used as Foreign key?

Solution

(i)

```

1 SELECT ProductName, Price
2 FROM PRODUCT
3 WHERE Price BETWEEN 50 AND 150;

```

(ii)

```

1 SELECT * FROM PRODUCT
2 WHERE Manufacturer = 'XYZ' OR Manufacturer = 'ABC';

```

(iii)

```

1 SELECT ProductName, Manufacturer, Price
2 FROM PRODUCT
3 WHERE Discount IS NULL;

```

(iv)

```

1 SELECT ProductName, Price
2 FROM PRODUCT
3 WHERE ProductName LIKE '%h';

```

(v)

```
1 SELECT C.ClientName, C.City, P.P_ID, P.ProductName
2 FROM PRODUCT P, CLIENT C
3 WHERE P.P_ID = C.P_ID AND C.CITY = 'DELHI';
```

(vi) The column used as a Foreign Key is P_ID in the CLIENT table, and it references the P_ID column in the PRODUCT table.

Question 31

Answer the questions based on the table given below:

Table: HOSPITAL

S.No.	Name	Age	Department	Dateofadm	Charges	Sex
1	Arpit	62	Surgery	1998-01-21	300	M
2	Zareena	22	ENT	1997-12-12	250	F
3	Kareem	32	Orthopaedic	1998-02-19	200	M
4	Arun	12	Surgery	1998-01-11	300	M
5	Zubin	30	ENT	1998-01-12	250	M
6	Ketaki	16	ENT	1998-02-24	250	F
7	Ankit	29	Cardiology	1998-02-20	800	F
8	Zareen	45	Gynaecology	1998-02-22	300	F
9	Kush	19	Cardiology	1998-01-13	800	M
10	Shilpa	23	Nuclear Medicine	1998-02-21	400	F

- (a) To list the names of all the patients admitted after 1998-01-15.
- (b) To list the names of female patients who are in ENT department.
- (c) To list the names of all patients with their date of admission in ascending order.
- (d) To display Patient's Name, Charges, Age for only female patients.
- (e) Find out the output of the following SQL commands:
 - SELECT COUNT(DISTINCT Charges) FROM HOSPITAL;
 - SELECT MIN(Age) FROM HOSPITAL WHERE Sex = "F";

Solution

(a)

```
1 SELECT NAME
2 FROM HOSPITAL
3 WHERE DATEOFADM > '1998-01-15';
```

(b)

```
1 SELECT NAME
2 FROM HOSPITAL
3 WHERE SEX = 'F' AND DEPARTMENT = 'ENT';
```

(c)

```
1 SELECT NAME, DATEOFADM
2 FROM HOSPITAL
3 ORDER BY DATEOFADM;
```

(d)

```
1 SELECT NAME , CHARGES , AGE
2 FROM HOSPITAL
3 WHERE SEX = 'F' ;
```

(e)

1.

```
1 SELECT COUNT(DISTINCT Charges) FROM HOSPITAL;
```

2.

```
1 SELECT MIN(Age) FROM HOSPITAL WHERE Sex = "F" ;
```

Question 32

A department store MyStore is considering to maintain their inventory using SQL to store the data. As a database administrator, Abhay has decided that:

- Name of the database — mystore
- Name of the table — STORE

The attributes of STORE are as follows:

ItemNo — numeric

ItemName — character of size 20

Scode — numeric

Quantity — numeric

Table: STORE

ItemNo	ItemName	Scode	Quantity
2005	Sharpener Classic	23	60
2003	Ball Pen 0.25	22	50
2002	Gel Pen Premium	21	150
2006	Gel Pen Classic	21	250
2001	Eraser Small	22	220
2004	Eraser Big	22	110
2009	Ball Pen 0.5	21	180

- Identify the attribute best suitable to be declared as a primary key.
- Write the degree and cardinality of the table STORE.
- Insert the following data into the attributes ItemNo, ItemName and SCode respectively in the given table STORE.

ItemNo = 2010, ItemName = "Note Book" and Scode = 25.

- Abhay wants to remove the table STORE from the database MyStore. Which command will he use from the following?

- DELETE FROM STORE;
- DROP TABLE STORE;

- DROP DATABASE MYSTORE;
 - DELETE STORE FROM MYSTORE;
- (e) Now Abhay wants to display the structure of the table STORE, i.e., name of the attributes and their respective data types that he has used in the table. Write the query to display the same.

Solution

- (a) ItemNo attribute is best suitable to be declared as a primary key as it uniquely identifies each item in the inventory.
- (b) The degree of the table STORE is 4, and the cardinality of the table STORE is 7.
- (c)

```
1 INSERT INTO STORE(ItemNo, ItemName, Scode)
2 VALUES(2010, 'Note Book', 25);
```

- (d) DROP TABLE STORE;

Reason — DROP TABLE command is used to remove/delete a table permanently. The syntax is : DROP TABLE *table_name*;. Hence, according to this DROP TABLE STORE; is the correct command to remove the STORE table from the database MyStore.

- (e)
-

```
1 DESCRIBE STORE;
```
