**Practical No 3**

**PART A**

**A.1 AIM: - To study and understand Strings, Lists in python**

**A.2 Prerequisite**

Programming for problem-solving and Object Oriented Programming

**A.3 Outcome**

After successful completion of this experiment, students will be able to understand and implement

1. Strings and string operations
2. Lists and list operations

**A.4 Theory**

In computer programming, a string is a sequence of characters. For example, "hello" is a string containing a sequence of characters 'h', 'e', 'l', 'l', and 'o'.

We use single quotes or double quotes to represent a string in Python. For example,

# create a string using double quotes

string1 = "Python programming"

# create a string using single quotes

string1 = 'Python programming'

Here, we have created a string variable named string1. The variable is initialized with the string Python Programming.

Example: Python String

# create string type variables

name = "Python"

print(name)

message = "I love Python."

print(message)

Output

Python

I love Python.

In the above example, we have created string-type variables: name and message with values "Python" and "I love Python" respectively.

Here, we have used double quotes to represent strings but we can use single quotes too.

Access String Characters in Python

We can access the characters in a string in three ways.

* Indexing: One way is to treat strings as a list and use index values. For example,

greet = 'hello'

# access 1st index element

print(greet[1]) # "e"

* Negative Indexing: Similar to a list, Python allows negative indexing for its strings. For example,

greet = 'hello'

# access 4th last element

print(greet[-4]) # "e"

* Slicing: Access a range of characters in a string by using the slicing operator colon :. For example,

greet = 'Hello'

# access character from 1st index to 3rd index

print(greet[1:4]) # "ell"

Note: If we try to access an index out of the range or use numbers other than an integer, we will get errors.

Python Strings are immutable

In Python, strings are immutable. That means the characters of a string cannot be changed. For example,

message = 'Hola Amigos'

message[0] = 'H'

print(message)

Output

TypeError: 'str' object does not support item assignment

However, we can assign the variable name to a new string. For example,

message = 'Hola Amigos'

# assign new string to message variable

message = 'Hello Friends'

prints(message); # prints "Hello Friends"

Python Multiline String

We can also create a multiline string in Python. For this, we use triple double quotes """ or triple single quotes '''. For example,

# multiline string

message = """

Never gonna give you up

Never gonna let you down

"""

print(message)

Output

Never gonna give you up

Never gonna let you down

In the above example, anything inside the enclosing triple-quotes is one multiline string.

Python String Operations

There are many operations that can be performed with strings which makes it one of the most used data types in Python.

1. Compare Two Strings

We use the == operator to compare two strings. If two strings are equal, the operator returns True. Otherwise, it returns False. For example,

str1 = "Hello, world!"

str2 = "I love Python."

str3 = "Hello, world!"

# compare str1 and str2

print(str1 == str2)

# compare str1 and str3

print(str1 == str3)

Output

False

True

In the above example,

* str1 and str2 are not equal. Hence, the result is False.
* str1 and str3 are equal. Hence, the result is True.

2. Join Two or More Strings

In Python, we can join (concatenate) two or more strings using the + operator.

greet = "Hello, "

name = "Jack"

# using + operator

result = greet + name

print(result)

# Output: Hello, Jack

In the above example, we have used the + operator to join two strings: greet and name.

Iterate Through a Python String

We can iterate through a string using a for loop. For example,

greet = 'Hello'

# iterating through greet string

for letter in greet:

print(letter)

Output

H

e

l

l

o

Python String Length

In Python, we use the len() method to find the length of a string. For example,

greet = 'Hello'

# count length of greet string

print(len(greet))

# Output: 5

String Membership Test

We can test if a substring exists within a string or not, using the keyword in.

print('a' in 'program') # True

print('at' not in 'battle') False

Methods of Python String

Besides those mentioned above, there are various string methods present in Python. Here are some of those methods:

|  |  |
| --- | --- |
| **Methods** | **Description** |
| upper() | converts the string to uppercase  message = 'python is fun'  # convert message to uppercase  print(message.upper())  # Output: PYTHON IS FUN  **Where it can be used:**  # example string  string = "this should be uppercase!"  print(string.upper())  # string with numbers  # all alphabets should be lowercase  string = "Th!s Sh0uLd B3 uPp3rCas3!"  print(string.upper()) |
| lower() | converts the string to lowercase  message = 'PYTHON IS FUN'  # convert message to lowercase  print(message.lower())  # Output: python is fun |
| partition() | returns a tuple  string = "Python is fun"  # 'is' separator is found  print(string.partition('is '))  # 'not' separator is not found  print(string.partition('not '))  string = "Python is fun, isn't it"  # splits at first occurence of 'is'  print(string.partition('is'))  Output:  ('Python ', 'is ', 'fun')  ('Python is fun', '', '')  ('Python ', 'is', " fun, isn't it")  ('Python ', 'is ', 'fun')  ('Python is fun', '', '')  ('Python ', 'is', " fun, isn't it") |
| replace() | replaces substring inside  text = 'bat ball'  # replace b with c  replaced\_text = text.replace('b', 'c')  print(replaced\_text)  # Output: cat call |
| find() | returns the index of first occurrence of substring  message = 'Python is a fun programming language'  # check the index of 'fun'  print(message.find('fun'))  # Output: 12 |
| rstrip() | removes trailing white space  title = 'Python Programming '  # remove trailing whitespace from title  result = title.rstrip()  print(result)  # Output: Python Programming |
| split() | splits string from left  text = 'Python is a fun programming language'  # split the text from space  print(text.split(' '))  # Output: ['Python', 'is', 'a', 'fun', 'programming', 'language'] |
| startswith() | checks if string starts with the specified string  message = 'Python is fun'  # check if the message starts with Python  print(message.startswith('Python'))  # Output: True |
| isnumeric() | checks numeric characters  pin = "523"  # checks if every character of pin is numeric  print(pin.isnumeric())  # Output: True |
| index() | returns index of substring  text = 'Python is fun'  # find the index of is  result = text.index('is')  print(result)  # Output: 7 |

Escape Sequences in Python

The escape sequence is used to escape some of the characters present inside a string.

Suppose we need to include both double quote and single quote inside a string,

example = "He said, "What's there?""

print(example) # throws error

Since strings are represented by single or double quotes, the compiler will treat "He said, " as the string. Hence, the above code will cause an error.

To solve this issue, we use the escape character \ in Python.

# escape double quotes

example = "He said, \"What's there?\""

# escape single quotes

example = 'He said, "What\'s there?"'

print(example)

# Output: He said, "What's there?"

Here is a list of all the escape sequences supported by Python.

|  |  |
| --- | --- |
| Escape Sequence | Description |
| \\ | Backslash |
| \' | Single quote |
| \" | Double quote |

Python String Formatting (f-Strings)

Python f-Strings make it really easy to print values and variables. For example,

name = 'Cathy'

country = 'UK'

print(f'{name} is from {country}')

Output

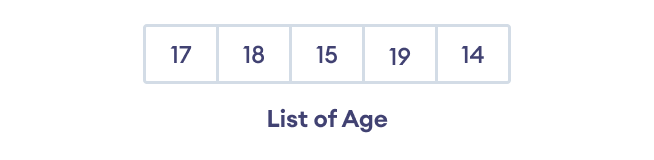
Cathy is from UK

Here, f'{name} is from {country}' is an f-string.

**Python List**

A list is a collection of similar or different types of data. For example,

Suppose we need to record the age of 5 students. Instead of creating 5 separate variables, we can simply create a list:

Elements of a list

Create a Python List

A list is created in Python by placing items inside [], separated by commas . For example,

# A list with 3 integers

numbers = [1, 2, 5]

print(numbers)

# Output: [1, 2, 5]

Here, we have created a list named numbers with 3 integer items.

A list can have any number of items and they may be of different types (integer, float, string, etc.). For example,

# empty list

my\_list = []

# list with mixed data types

my\_list = [1, "Hello", 3.4]

Access Python List Elements

In Python, each item in a list is associated with a number. The number is known as a list index.

We can access elements of an array using the index number (0, 1, 2 …). For example,

languages = ["Python", "Swift", "C++"]

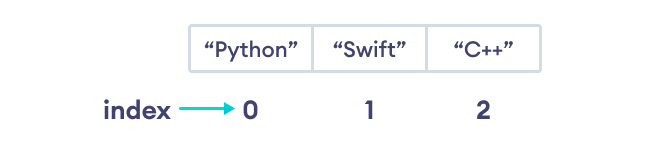
# access item at index 0

print(languages[0]) # Python

# access item at index 2

print(languages[2]) # C++

In the above example, we have created a list named languages.



List Indexing in Python

Here, we can see each list item is associated with the index number. And, we have used the index number to access the items.

Note: The list index always starts with 0. Hence, the first element of a list is present at index 0, not 1.

Negative Indexing in Python

Python allows negative indexing for its sequences. The index of -1 refers to the last item, -2 to the second last item and so on.

Let's see an example,

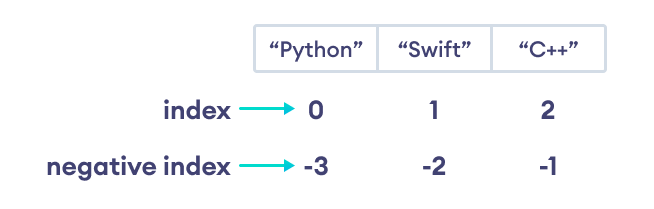
languages = ["Python", "Swift", "C++"]

# access item at index 0

print(languages[-1]) # C++

# access item at index 2

print(languages[-3]) # Python

Python Negative Indexing

Note: If the specified index does not exist in the list, Python throws the IndexError exception.

Slicing of a Python List

In Python it is possible to access a section of items from the list using the slicing operator :, not just a single item. For example,

# List slicing in Python

my\_list = ['p','r','o','g','r','a','m','i','z']

# items from index 2 to index 4

print(my\_list[2:5])

# items from index 5 to end

print(my\_list[5:])

# items beginning to end

print(my\_list[:])

Output

['o', 'g', 'r']

['a', 'm', 'i', 'z']

['p', 'r', 'o', 'g', 'r', 'a', 'm', 'i', 'z']

Here,

* my\_list[2:5] returns a list with items from index 2 to index 4.
* my\_list[5:] returns a list with items from index 1 to the end.
* my\_list[:] returns all list items

Note: When we slice lists, the start index is inclusive but the end index is exclusive.

Add Elements to a Python List

Python List provides different methods to add items to a list.

1. Using append()

The append() method adds an item at the end of the list. For example,

numbers = [21, 34, 54, 12]

print("Before Append:", numbers)

# using append method

numbers.append(32)

print("After Append:", numbers)

Output

Before Append: [21, 34, 54, 12]

After Append: [21, 34, 54, 12, 32]

In the above example, we have created a list named numbers. Notice the line,

numbers.append(32)

Here, append() adds 32 at the end of the array.

2. Using extend()

We use the extend() method to add all items of one list to another. For example,

prime\_numbers = [2, 3, 5]

print("List1:", prime\_numbers)

even\_numbers = [4, 6, 8]

print("List2:", even\_numbers)

# join two lists

prime\_numbers.extend(even\_numbers)

print("List after append:", prime\_numbers)

Output

List1: [2, 3, 5]

List2: [4, 6, 8]

List after append: [2, 3, 5, 4, 6, 8]

In the above example, we have two lists named prime\_numbers and even\_numbers. Notice the statement,

prime\_numbers.extend(even\_numbers)

Here, we are adding all elements of even\_numbers to prime\_numbers.

# languages list

languages = ['French', 'English']

# another list of language

languages1 = ['Spanish', 'Portuguese']

# appending language1 elements to language

languages.extend(languages1)

print('Languages List:', languages)

You can also append all elements of an iterable to the list using:

1. the + operator

a = [1, 2]

b = [3, 4]

a += b # a = a + b

a=a+b

a=[1,2]+[3,4]

a=[1,2,3,45]

# Output: [1, 2, 3, 4]

print('a =', a)

Output

a = [1, 2, 3, 4]

2. the list slicing syntax

a = [1, 2]

b = [3, 4]

a[len(a):] = b

# Output: [1, 2, 3, 4]

print('a =', a)

Output

a = [1, 2, 3, 4]

Python extend() Vs append()

If you need to add an element to the end of a list, you can use the append() method.

a1 = [1, 2]

a2 = [1, 2]

b = (3, 4)

# a1 = [1, 2, 3, 4]

a1.extend(b)

print(a1)

# a2 = [1, 2, (3, 4)]

a2.append(b)

print(a2)

Output

[1, 2, 3, 4]

[1, 2, (3, 4)]

Change List Items

Python lists are mutable. Meaning lists are changeable. And, we can change items of a list by assigning new values using = operator. For example,

languages = ['Python', 'Swift', 'C++']

# changing the third item to 'C'

languages[2] = 'C'

print(languages) # ['Python', 'Swift', 'C']

Here, initially the value at index 3 is 'C++'. We then changed the value to 'C' using

languages[2] = 'C'

Remove an Item From a List

1. Using del()

In Python we can use [the del statement](https://www.programiz.com/python-programming/del) to remove one or more items from a list. For example,

languages = ['Python', 'Swift', 'C++', 'C', 'Java', 'Rust', 'R']

# deleting the second item

del languages[1]

print(languages) # ['Python', 'C++', 'C', 'Java', 'Rust', 'R']

# deleting the last item

del languages[-1]

print(languages) # ['Python', 'C++', 'C', 'Java', 'Rust']

# delete first two items

del languages[0 : 2] # ['C', 'Java', 'Rust']

print(languages)

2. Using remove()

We can also use the remove() method to delete a list item. For example,

languages = ['Python', 'Swift', 'C++', 'C', 'Java', 'Rust', 'R']

# remove 'Python' from the list

languages.remove('Python')

print(languages) # ['Swift', 'C++', 'C', 'Java', 'Rust', 'R']

Here, languages.remove('Python') removes 'Python' from the languages list.

Python List Methods

Python has many useful [list methods](https://www.programiz.com/python-programming/methods/list) that makes it really easy to work with lists.

|  |  |
| --- | --- |
| **Method** | **Description** |
| append() | add an item to the end of the list |
| extend() | add items of lists and other iterables to the end of the list |
| insert() | inserts an item at the specified index  # create a list of vowels  vowel = ['a', 'e', 'i', 'u']  # 'o' is inserted at index 3 (4th position)  vowel.insert(3, 'o')  print('List:', vowel)  # Output: List: ['a', 'e', 'i', 'o', 'u'] |
| remove() | removes item present at the given index  # create a list  del  # Output: Updated List: [2, 3, 5, 7, 11] |
| pop() | returns and removes item present at the given index  # create a list of prime numbers  prime\_numbers = [2, 3, 5, 7]  # remove the element at index 2  removed\_element = prime\_numbers.pop(2)  print('Removed Element:', removed\_element)  print('Updated List:', prime\_numbers)  # Output:  # Removed Element: 5  # Updated List: [2, 3, 7] |
| clear() | removes all items from the list  prime\_numbers = [2, 3, 5, 7, 9, 11]  # remove all elements  prime\_numbers.clear()  # Updated prime\_numbers List  print('List after clear():', prime\_numbers)  # Output: List after clear(): [] |
| index() | returns the index of the first matched item  animals = ['cat', 'dog', 'rabbit', 'horse']  # get the index of 'dog'  index = animals.index('dog')  print(index)  # Output: 1  \* |
| count() | returns the count of the specified item in the list  # create a list  numbers = [2, 3, 5, 2, 11, 2, 7]  # check the count of 2  count = numbers.count(2)  print('Count of 2:', count)  # Output: Count of 2: 3 |
| sort() | sort the list in ascending/descending order  prime\_numbers = [11, 3, 7, 5, 2]  # sorting the list in ascending order  prime\_numbers.sort()  print(prime\_numbers)  # Output: [2, 3, 5, 7, 11] |
| reverse() | reverses the item of the list  # create a list of prime numbers  prime\_numbers = [2, 3, 5, 7]  # reverse the order of list elements  prime\_numbers.reverse()  print('Reversed List:', prime\_numbers)  # Output: Reversed List: [7, 5, 3, 2] |
| copy() | returns the shallow copy of the list  # mixed list  prime\_numbers = [2, 3, 5]  # copying a list  numbers = prime\_numbers.copy()  print('Copied List:', numbers)  # Output: Copied List: [2, 3, 5] |

Iterating through a List

We can use the [for loop](https://www.programiz.com/python-programming/for-loop) to iterate over the elements of a list. For example,

languages = ['Python', 'Swift', 'C++']

# iterating through the list

for language in languages:

print(language)

Output

Python

Swift

C++

Check if an Item Exists in the Python List

We use the in keyword to check if an item exists in the list or not. For example,

languages = ['Python', 'Swift', 'C++']

print('C' in languages) # False

print('Python' in languages) # True

Here,

* 'C' is not present in languages, 'C' in languages evaluates to False.
* 'Python' is present in languages, 'Python' in languages evaluates to True.

Python List Length

In Python, we use the len() method to find the number of elements present in a list. For example,

languages = ['Python', 'Swift', 'C++']

print("List: ", languages)

print("Total Elements: ", len(languages)) # 3

Output

List: ['Python', 'Swift', 'C++']

Total Elements: 3

**Tasks:**

1. **Write a code for the following: (String Operations)**

#### Reverse a sentence accepted as input by User

#### Find the characters at an odd position in string input by User

#### Check string starts with a specific character entered by the user

#### Remove all newlines from the String

#### Replace all occurrence of substring in string

#### Remove punctuation mark from list of string

#### Find the number of matching characters in two string

#### Convert a string into a list.

#### To convert all string elements of the list to int.

* Count Total numbers of upper case and lower case characters in input string
* To find vowels in a string

#### To sort a list of string in Python

#### To print input string in upper case and lower case

#### Convert Int To String In Python

1. **Write a Python program to accept a string and replace all spaces by ‘#’ symbol.**
2. **Write a program to accept two strings from the user and display the common words (ignore case)**
3. **Write a program to accept a string and count the frequency of each vowel.**

#### **Create a list in python for storing supermarket bill details and perform the following operations on it:**

#### Add an item to the end of the list

#### Insert an item at a given position

#### Modify an element by using the index of the element

#### Remove an item from the list

#### Remove all items from the list

#### Slice Elements from a List

#### Remove the item at the given position in the list, and return it

#### Return the number of times 'x' appear in the list

#### Sort the items of the list in place

#### Reverse the elements of the list in place

1. Write a python program to count unique values inside a list
2. Write a python program to print a list excluding the duplicates
3. Write a python program to count positive and negative numbers in a list
4. Write a python program to sum all the elements in a list.
5. Write a Python program to find the list of words that are longer than n from a given list of words.
6. Write a Python program to compute the difference between two lists.  
   Sample data: ["red", "orange", "green", "blue", "white"], ["black", "yellow", "green", "blue"]  
   Expected Output:  
   Color1-Color2: ['white', 'orange', 'red']  
   Color2-Color1: ['black', 'yellow']
7. Write a Python program to concatenate elements of a list.
8. Write a Python program to insert a given string at the beginning of all items in a list.  
   Sample list : [1,2,3,4], string : emp  
   Expected output : ['emp1', 'emp2', 'emp3', 'emp4']
9. Write a Python program to find the list in a list of lists whose sum of elements is the highest.  
   Sample lists: [1,2,3], [4,5,6], [10,11,12], [7,8,9]  
   Expected Output: [10, 11, 12]

PART B

(PART B: TO BE COMPLETED BY STUDENTS)

**(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Teams or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)**

|  |  |
| --- | --- |
| Roll No. C137 | Name: Naisha Shetty |
| Program: BTI CS | Division: D |
| Semester: 4 | Batch : D1 |
| Date of Experiment: 10/1/2024 | Date of Submission: 11/1/2024 |
| Grade : |  |

B.1 Software Code written by student:

***(Paste your Python code completed during the 2 hours of practical in the lab here)***

Q1a.

sentence = input("Enter a sentence: ")

word\_list = sentence.split()

reversed\_list = word\_list[:: -1]

reversed\_sentence = " ".join(reversed\_list)

print(reversed\_sentence)

Q1b.

string = input("Enter a string: ")

print("Odd Position:", string[1::2])

Q1c.

user\_input = input("Enter a string: ")

start\_character = input("Enter the character to check for: ")

if user\_input.startswith(start\_character):

print("String starts with", start\_character)

else:

print("String does not start with", start\_character)

Q1d.

txt = "Naisha\n is good in playing\n cricket."

cleaned\_string = txt.replace("\n","")

print("String without newlines: ", cleaned\_string)

Q1e.

user\_input = input("Enter a string: ")

old\_substring = input("Enter the substring to replace: ")

new\_substring = input("Enter the new substring: ")

updated\_string = user\_input.replace(old\_substring, new\_substring)

print("Updated string: ", updated\_string)

Q1f.

import string

str\_1 = input ("Enter a string: ")

no\_punc = str\_1.translate (str.maketrans ('', '', string.punctuation))

print ("The string without punctuation is:", no\_punc)

Q1g.

# Input two strings

str1 = input("Enter the first string: ")

str2 = input("Enter the second string: ")

# Convert strings to lowercase to perform case-insensitive comparison

str1\_lower = str1.lower()

str2\_lower = str2.lower()

# Initialize a counter for matching characters

matching\_characters = 0

# Iterate through each character in both strings

for char1, char2 in zip(str1\_lower, str2\_lower):

# Check if characters match

if char1 == char2:

matching\_characters += 1

# Print the result

print(f"Number of matching characters: {matching\_characters}")

Q1h.

user\_input = input("Enter a string: ")

string\_list = list(user\_input)

print("String as a list: ", string\_list)

Q1i.

string\_list = ['1', '2', '3', '4', '5']

int\_list = list(map(int, string\_list))

print("List of integers: ", int\_list)

Q1j.

user\_input = input("Enter a string: ")

uppercase\_count = sum(1 for char in user\_input if char.isupper())

lowercase\_count = sum(1 for char in user\_input if char.islower())

print("Uppercase count:", uppercase\_count)

print("Lowercase count:", lowercase\_count)

Q1k.

user\_input = input("Enter a string: ")

vowels = [char for char in user\_input if char.lower() in 'aeiou']

print("Vowels in the string: ", vowels)

Q1l.

user\_input = input("Enter a string: ")

sorted\_string = ''.join(sorted(user\_input))

print("sorted string: ", sorted\_string)

Q1m.

user\_input = input("Enter a string: ")

print("Uppercase: ", user\_input.upper())

print("Lowercase: ", user\_input.lower())

Q1n.

integer\_value = 42

string\_value = str(integer\_value)

print("Converted string: ", string\_value)

Q2.

string = input("Enter a string: ")

new\_string = string.replace(" ", "#")

print("String with replaced spaces: ", new\_string)

Q3.

string1 = input("Enter the first string: ").lower()

string2 = input("Enter the second string: ").lower()

words1 = set(string1.split())

words2 = set(string2.split())

common\_words = words1.intersection(words2)

print("Common words: ", common\_words)

Q4.

vowels = 'aeiou'

string = input("Enter a string: ").lower()

vowel\_count = {}

for vowel in vowels:

vowel\_count[vowel] = string.count(vowel)

print("Vowel frequencies: ", vowel\_count)

Q5.

supermarket\_bill = ["Milk", "Bread", "Eggs", "Vegetables", "Fruits"]

supermarket\_bill.append("Cheese")

print("After adding an item at the end:", supermarket\_bill)

supermarket\_bill.insert(2, "Juice")

print("After inserting an item at position 2:", supermarket\_bill)

supermarket\_bill[3] = "Fresh Vegetables"

print("After modifying an element at index 3:", supermarket\_bill)

supermarket\_bill.remove("Bread")

print("After removing 'Bread' from the list:", supermarket\_bill)

supermarket\_bill.clear()

print("After removing all items from the list:", supermarket\_bill)

supermarket\_bill.extend(["Cereal", "Coffee", "Snacks", "Milk", "Bread", "Eggs"])

sliced\_elements = supermarket\_bill[1:4]

print("Sliced elements from index 1 to 3:", sliced\_elements)

removed\_item = supermarket\_bill.pop(2)

print("Removed item at position 2:", removed\_item)

print("List after removal:", supermarket\_bill)

count\_milk = supermarket\_bill.count("Milk")

print("Number of times 'Milk' appears in the list:", count\_milk)

supermarket\_bill.sort()

print("Sorted list:", supermarket\_bill)

supermarket\_bill.reverse()

print("Reversed list:", supermarket\_bill)

Q6.

list1 = [1, 2, 2, 3, 4, 4, 5]

unique\_count = len(set(list1))

print("Number of unique values: ", unique\_count)

Q7.

list1 = [1, 2, 2, 3, 4, 4, 5]

unique\_list = list(set(list1))

print("List without duplicates: ", unique\_list)

Q8.

list1 = [-2, -1, 0, 1, 2, -3, 4]

positive\_count = sum(1 for num in list1 if num>0)

negative\_count = sum(1 for num in list1 if num<0)

print("Positive numbers: ", positive\_count)

print("Negative numbers: ", negative\_count)

Q9.

list1 = [1, 2, 3, 4, 5]

sum\_of\_elements = sum(list1)

print("Sum of elements: ", sum\_of\_elements)

Q10.

words\_list = ["apple", "banana", "orange", "grape", "kiwi", "melon"]

n\_value = 5

long\_words = []

for word in words\_list:

if len(word) > n\_value:

long\_words.append(word)

print("Words longer than", n\_value, "characters:", long\_words)

Q11.

color1 = ["red", "orange", "green", "blue", "white"]

color2 = ["black", "yellow", "green", "blue"]

color1\_minus\_color2 = list(set(color1) - set(color2))

color2\_minus\_color1 = list(set(color2) - set(color1))

print("Color1-Color2:", color1\_minus\_color2)

print("Color2-Color1:", color2\_minus\_color1)

Q12.

my\_list = ["Hello", " ", "World", "!"]

result = ''.join(my\_list)

print("Concatenated String:", result)

Q13.

my\_list = [1, 2, 3, 4]

prefix\_string = "emp"

result\_list = [prefix\_string + str(item) for item in my\_list]

print("Expected Output:", result\_list)

Q14.

lists\_of\_lists = [[1, 2, 3], [4, 5, 6], [10, 11, 12], [7, 8, 9]]

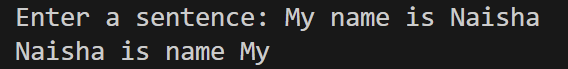
max\_sum\_list = max(lists\_of\_lists, key=sum)

print("List with the highest sum:", max\_sum\_list)

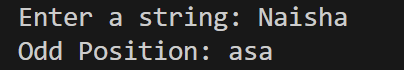
B.2 Input and Output:

***(Paste your program input and output in following format. If there is error then paste the specific error in the output part. In case of error with due permission of the faculty extension can be given to submit the error free code with output in due course of time. Students will be graded accordingly.)***

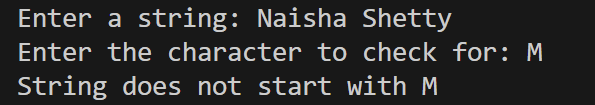
Q1a.



Q1b.



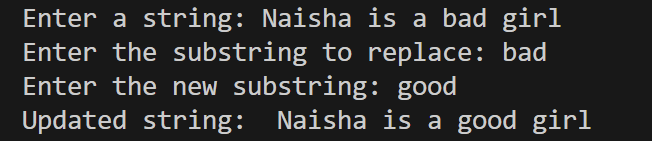
Q1c.



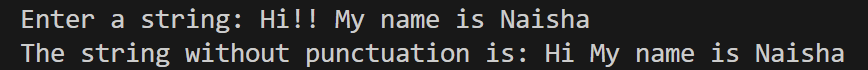
Q1d.



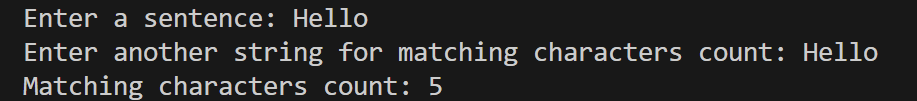
Q1e.



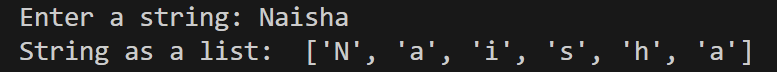
Q1f.



Q1g.



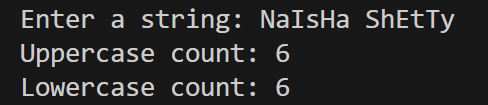
Q1h.



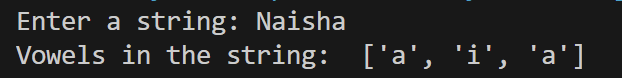
Q1i.



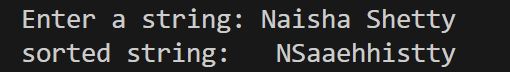
Q1j.



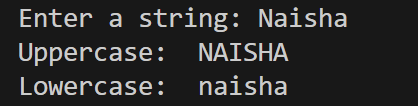
Q1k.



Q1l.



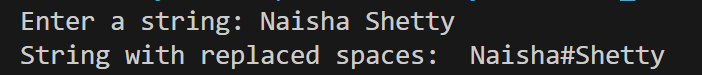
Q1m.



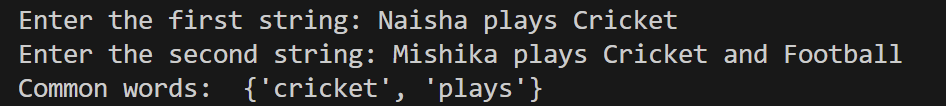
Q1n.



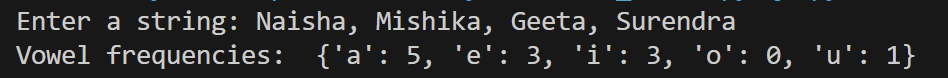
Q2.



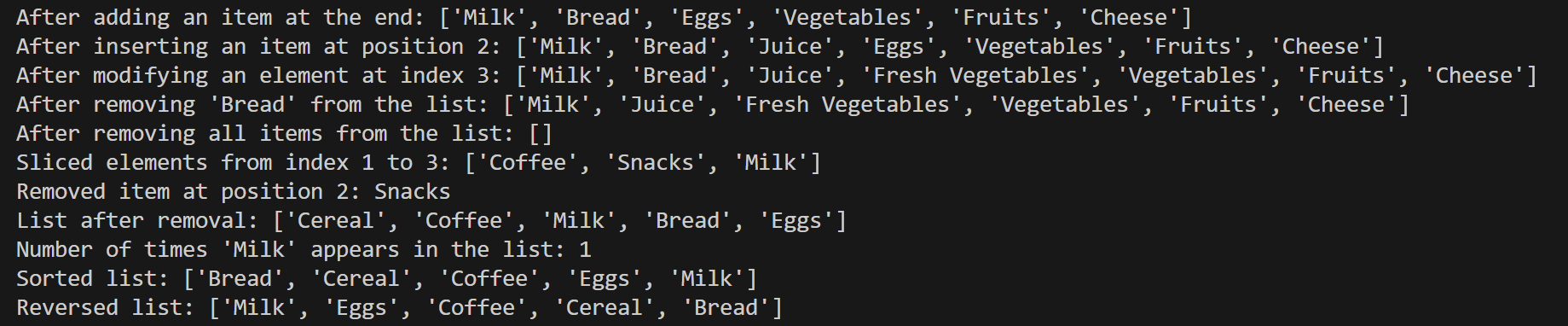
Q3.



Q4.



Q5.



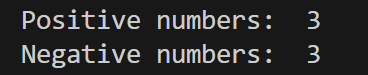
Q6.



Q7.



Q8.



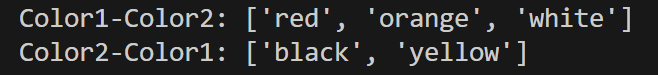
Q9.



Q10.



Q11.



Q12.



Q13.



Q14.



B.3 Conclusion:

*(****Students must write the conclusion as per the attainment of individual outcome listed above and learning/observation noted in section B.1)***

*We learnt how to write a program for problem solving and Object Oriented Programming.*

*We learnt Python Programming for Strings and Lists.*