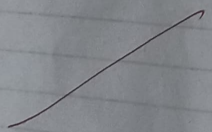
Practical No. 1

Aim :- Print only the words that start with letter 's' in the following statement :- st 'print only the word that start with s in sentence'





GCOEN
Page No: 2
Date: / /

Aim :- Print only the words that start with letter 's' in the following statement :- st 'print only the word that start with s in sentence'

Theory :- Python is widely used general-purpose, high level programming language. It was created by Guido van Rossum in 1991 and further developed by the Python Software Foundation. It was designed with an emphasis on code redability, and its syntax allows programmers to express their concepts in fewer lines of code.

Python is a programming language that lets you work quickly and integrate systems more efficiently.

There are two major Python versions :

- Python 2
- Python 3

Features of Python :

- Free and Open Source
- Easy to code
- Easy to read
- Object Oriented language
- GUI Programming Support
- High-Level language
- Large community Support
- Easy to Debug
- Large Standard library

## Strings in Python :

Like many other popular programming languages, strings in Python are arrays of bytes representing unicode characters.

Strings in python are surrounded by either single quotation marks, or double quotation marks
'hello' is same as "hello"
we can display string literal with print() functio
Example : print ("Hello")
print ('hello')

A string is a data structure in Python that represents a sequence of characters.

## String methods in Python :

1) upper() :- It will return given string to upper case

e.g str1 = "python is cool"
print (str1.upper())

Output :-

Starts
s
Sentence

Conclusion :- We have successfully programm python code which prints only words which starts with s.

16/1/24



2) lower() :- It will return given string to lower case

e.g. str1 = "PYTHON IS COOL"
print (str1.lower())

3) split() :- It will splits string and return a list. can also seperator and maxsplit (optional)

e.g. str1 = "python is cool"
print (str1.split())

4) startswith() :- It will return boolean value of string start with specified value

e.g. str1 = "python"
print (str.startswith('p'))

5) capitalize() :- It will return string with first character to upper case

e.g. str1 = "python"
print (str1.capitalize())

Code :-
```
str1 = "print only the word that starts with s in this sentence".
for word in str1.split():
    if word.startswith('s'):
        print(word)
```

Conclusion :- we successfully program python code which prints only words which starts with s.

Practical No. 2

Aim :- Print every word from the below sentence which has even number of letters.
Str - 'print only the word that starts with s in these sentence'.





Aim :- Print every word from the below sentence which has even number of letters.
Str - 'print only the word that starts with s in these sentence'.

Theory :- What is List ? How to define list in Python?

List is a collection of multiple items stored at single variable name. In general list is comparable to array. It is a sequence data type which is used to store the collection of data.
Python Lists are just like dynamically sized arrays, declared other languages (vector in C++, ArrayList in Java).

In Python list is declared in square brackets and elements are seperated by commas.

Syntax → list = [1, 2, 3, 4]
print ( type (list) )

List can be used to store multiple types of data.

Ex :- list = [1, "Hello", True]
print (type (list))

Output :- print only the word that start with s in this sentence.
list converted string.
even length words.
list of word : only
word
that
with
this
in
sentence



List can also be created by list() constructor.

Ex :- num = list ((1,2,3)) # item surrounded with double ()
print (num)

Length function (len()) :- It will return an integer which is the size of list passed as parameter.

Ex :- num = [1,2,3,4,5]
print (len(num))

Code :-
```
str = "print only the word that starts with s in this sentence"
word = list (str.split())
print ("str: ", str)
print ("list converted string", word)
print ("even length words")
for i in word:
    if leng(i) % 2 == 0:
        print (i)
```

Conclusion :- We successfully studied about list and len() function and wrote code to find even length words from given statement.

Nagware 23/11/24 B

# Practical No. 3

Page No. 7
Date

**Aim** :- Write a program that prints the integer from 1 to 100, but for multiples of 3 print 'FIZZ' instead of number and for multiples of five print 'BUZZ'. For numbers which are multiples of both 3 and 5 print 'FIZZBUZZ'.

**Theory** :- Conditional Statements

In programming these statements are used to control the flow of program on certain conditions.

These statements allow the execution of different code block depending on whether a specified condition evaluates to true or false providing a mechanism to make decision in algorithm.

In python there are three conditional statements.

i) if statement
ii) elif statement (else if)
iii) else statement

Condition could be an expression or boolean value if true will execute code defined under the statement.

i) **if-statement** :- It is most basic conditional statement if true will be execute if false the block of code will skipped.

Ex:- x = 0
if x == 0 :
print ("True")

ii) elif - statement :- In elif is always declared after if statement it is used to get more than one output of single condition it can be used multiple times in program.

Ex :- x = 1
if x == 0 :
print ("condition 1")
elif x == 1 :
print ("condition 2")

iii) else - statement :- There are no condition passed through else statement if use declare after if - statement or sequence of elif statements. If all statements are false then else statement will execute definitely.

Ex :- x = 1
if x == 0 :
print ("x is 0")
else :
print ("x is not 0") :

Indentation :- refers to the spaces at the begining of a code line. In other programming languages they used {} brackets.

Ex :- x = 0
if x == 0:
print ("True")

Code :-
```
for i in range (1,101):
    if (i%.3 ==0 and i%.5 ==0):
        print ("FIZZBUZZ")
    elif (i%.3 ==0):
        print ("FIZZ")
    elif (i%.5 ==0):
        print ("BUZZ")
    else:
        print (i)
```

Output :-
```
1
2
FIZZ
4
BUZZ
FIZZ
7
8
FIZZ
BUZZ
11
FIZZ
13
14
FIZZBUZZ
16
17
:
:
:
100
```

Conclusion :- We successfully studied about conditional statements and wrote code to print 'FIZZ' when number is divisible by 3 'BUZZ' when number is divisible by 5 and 'FIZZBUZZ' when number is divisible by both 3 and 5 for number 1 to 100.

Raguvale 30/1/24 A

# Practical No. 4.

GCOEN
Page No:
Date: / /

**Aim** :- Write a python funstion to check who is employee of the month.

**Theory** :- Set is an unordered collection of unique elements. It is singular to a mathematical set that has following features.

**Unique element** :- Sets do not allow duplicate elements. If you try to add a duplicate elements to set, it will be ignored.

**Unordered** : Unlike list of tuples, set do not maintain the order of elements. The elements are stored in unordered manner.

**Mutable** :- Sets are mutable, meaning you can modify them by adding or removing elements.

**Immutable elements** :- While sets themselves are mutable, the elements they contain must be immutable.

Operations on set include union, intersection, difference and Symmetric difference.



Teacher's Signature

type() : From Python's perspective sets are defined as object with the data type 'set'.

Ex mySet = {"apple", "kiwi", "orange"}
print (type (myset))

Output : <class 'set'>.

Union operation on sets : Two sets can be merged using union() function or | operator. Both Hash table values are accessed and traversed with merge operation perform on them to combine the elements, at the same duplicates are removed.

Ex : Set 1 = {1, 2, 3, 4, 5}

Set 2 = {3, 5, 6, 7, 8}

print (Set 1. union (Set 2))

Output : {1, 3, 2, 5, 4, 6, 7, 8}


Teacher's Signature

Output :-

Employee of the month is : Sahil

Conclusion :- Hence we had successfully performed program using function to check who is employee of the month

GCOEN
Page No:
Date: / /

Code :-

```
def employee_of_month(employees):
    max_score = 0
    emp_of_none = None
    for i in employee:
        score = i['attendance'] + i['performance']
    if (score > max_score):
        max_score = score
        emp_of_mont = i['Name']
    return emp_of_month

Employees_d = [{'Name': 'Sahil', 'attendance': 99,
                'performance': 100}
               {'Name': 'Jane', 'attendance': 95,
                'Performance': 95}
               {'Name': 'John', 'attendance': 90,
                'Performance': 90}]

eom = employee_of_month(Employees_d)
print("Employee of month is :", eom)
```

Conclusion :- Hence we had successfully performed program using function to check who is employee of month.

Payal 6/12/24 (A)

# Practical No. 5



**Aim :-** Write a program to print next 5 days starting from today.

**Theory :-** Python datetime module :

In Python, date and time are not data types of their own, but a module name Datetime in Python can be imported to work with the date as well as time. Python Datetime module comes built into Python. So there is no need to install it externally.

Python Datetime module supplies classes to work with date and time. These classes provide several functions to deal with dates, times and time intervals. Date and Datetime are an object in Python, so when you manipulate them, you are manipulating object and not strings or timestamps.

The Datetime module is categorized into 6 main class.

- **date** : An idealized naive date, assuming the current gregomen calender always was, and always was, and always will be in effect. its attributes are year, month and day. You can use date class.
- **time** : An idealized time, independent of any particular day assuming that every day has externally $24*60*60$ seconds. His attributes are hour, minute, second, macro-second and tzinfo.

Output :-

```
Today : 2024-04-03
Day 1 : 2024-04-04
Day 2 : 2024-04-05
Day 3 : 2024-04-06
Day 4 : 2024-04-07
Day 5 : 2024-04-08
```

Conclusion :- We successfully saw usage of date time module through program.



- date time : It is a combination of date and time along with the attributes year, month, hour, minute, microsecond and tzinfo.
- timedelta : A duration expressing the difference two date, time or datetime instances to microsecond resolution.
- tzinfo : It provide time zone information objects.
- timezone : A class that implements the tzinfo abstract base class as a fixed offset from the UTC.

Code :

```
import datetime
today = datetime.date.today()
print("Today: ", today)

for i in range(1,6):
    next_day = today + datetime.timedelta
print("Day"+i, ":", next_day)
```

Conclusion :- We successfully saw usage of date module through program

Regular 13/12/24 A

GCOEN
Page No:
Date: / /

# Practical No. 6

**Aim :-** Write a program that return the lesser of two given number if both numbers are even, but return the greater value if one or both numbers are odd.

**Theary :-** Min() and max() fumtions are used to compute the minimum and maximum of the passed as argument or it also gives lexicographically largest value and lexicagraphically small value respectively when String or list of string is passed.

i) min() :- It is used to get smallest value or lexicographically smaller values where an iterable is passed.

Syntax :- min (iterable) or
min (a, b, c, key, default)

Parameters :- a, b, c, ....., similar types of data

key : Key function where the iterable are passed and comparison is performed.

default :- default value is passed if the given iterable is empty.

Output :-

Enter the first number : 13
Enter the second number : 17
Result = 17

Conclusion :- Hence, we successfully performed python program that returns the lesser of two given number if both numbers are even but returns the greater if one or both numbers are odd.

Nagrale 5/3/24 (A)

GCOEN
Page No:
Date: / /

ii) max() :- The function is used to compute the maximum value passed in its arguments and lexicographically largest value if strings are passed as arguements.

Syntax : max (a, b, c, ... key, default)
Parameters : ab, b, c smaller types of data
Key : key function where the iterable are passed and comparison performed.
default :- default value is passed if given iterable is empty.

Code :-

```
def check (n1, n2):
    if (n1 % 2 == 0) and (n2 % 2 == 0):
        res = min(n1, n2)
    else:
        res = max(n1, n2)
    return res
num1 = int (input ("Enter the first number: "))
num2 = int (input ("Enter the second number: "))
result = check (num1, num2).
print ("Result = ", result)
```

Conclusion :- Hence, we successfully performed python program that returns the lesser of two given number if both numbers are even but returns the greater if one or both numbers are odd.

Practical No. 17

Page No.
Date

Aim :- Write a python function that accepts a string and calculate the number of upper case and lower case letters

Theory :- A function is a block of code which only runs when it is called.

- you can pass data as parameters seperated by commas ",".
- you can get data as return value.
- Syntax : def name-fn (parameters):
  code :....
  return result

i) A function is defined by "def" keyword
ii) A name is given after "def" keyword
iii) Parameters are passed through "( )" (if only)
iv) At least will return a result (if any)

- We can call a function by defining name and providing set of parenthesis and provide data to it.

Example

```
def Add one (X):
    return x + 1
print (Add one (10))
```

Output : Enter String
"Python is best Programming language"

Uppercase letters : 2
Lowercase letters : 29

Conclusion :- We successfully studied about functions and wrote a program to calculate uppercase and lowercase letters in string.

Grade
12/13 x 4
(A)

Functions used in code.

i) isupper() : will return true if given string is in uppercase

Ex :- print ("M", isupper())

ii) islower() : will return true if given string is in lowercase

Ex :- print ("m". islower())

Note :- both funtion ignores Numbers, Symbols and Spaces

Code :-

```
str = input ("Enter String")
upper = 0
lower = 0
for x in str :
    if x.isupper():
        upper += 1
    elif x.islower():
        lower += 1
print ("Uppercase letters", upper)
print ("Lowercase letters", lower)
```

Conclusion :- We successfully studied about functions and wrote a program to calculate uppercase and lower case letters in string.

# Practical No. 8

**Aim** :- Write a Python function that takes a list and return a new list with unique element of the first list. For example, Sample list = [1,1,1,2,2,3,3,4] unique list = [1,2,3,4]

**Theory** :- Method to add element in Python list.

i) Append :- This method is used to add single element at end of list.

Syntax

```
list.append(element)
```

Example :-

```
num = [1,2,3,4]
num.append(5)
print(num)
```

ii) Insert :- This method used to add single element at specific index of list.

Syntax :

```
list.insert(index, element)
```

Example :-

```
list = [1,2,3]
list.insert(0,0)
print(list)
```

Output :-

Enter list : 1 2 2 3 3 4 4 5 6 7 7

list : ['1', '2', '2', '3', '3', '4', '4', '5', '6', '7', '7']

unique list : ['1', '2', '3', '4', '5', '6', '7']

onclusion :- Hence we have successfully studied python function list methods with examples.

iii) Extend :- This method is used to join two list or iterate behind a list (can also join set, tuple etc.)

Syntax :- list.extend (iterate)

Example :-
```
a = [1,2,3]
b = [4,5,6]
a.extend(b)
print(a)
```

Code :-
```
def find_unique(list):
    unique_list = []
    for i in list
        if i not in unique_list:
            unique_list.append(i)
    return unique_list
sample_list = list(input("Enter list : ").split())
print("list : ", sample_list)
unique_list = find_unique(sample_list)
print("unique list : ", unique_list)
```

Conclusion :- Hence we have successfully studied python function list methods with examples.

Dayesh 19/3/24 A

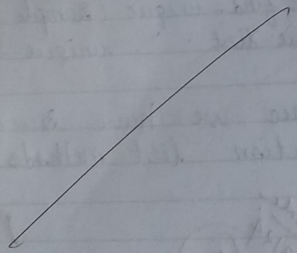
Practical No. 9

Aim :- Write a python function to multiply all the numbers in the list.

i) Output :-
2
3
5
7
11

ii) Output :-
7
3
5
7
9

iii) Output :-
7
2
3
4



Practical No. 9

GCOEN
Page No:
Date: / /

Aim :- Write a python function to multiply all the numbers in the list

Theory :- Traversing :
Traversing in terms of programming means visiting every single element of list or collection of data once or at best once.

Different methods to Traverse list in python

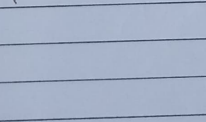
i) Loops :- We can use for while loop to access data in list once.

Example :-
```
x = [2,3,5,7,11]
for i in x : print(i)

x,i = [2,3,5,7,11],0
while(i < len(x)) : print(x[i]);
    i += 1
```

ii) Recursion :- We use a function which call itself to access data in list once.

Example :-
```
def recursion(num, begin):
    if len(num) > begin :
        Print(num[begin])
    recursion(num, begin+1)
x = [1,3,5,7,9]
recursion(x,0)
```



Piyush Enterprises

Teacher's Signature

Output :- Enter Number of elements 4
Enter element : 1
Enter element : 2
Enter element : 6
Enter element : 6
Number to multiply 10
[10, 20, 60, 50]

Conclusion :- We successfully studied about traversing method in python and write a program to multiply all the numbers in the list.



iii) List comprehension :- Python provide this feature with shorter syntax and without any comprehensive we can use to traverse list in python.

Example :- x = [1,2,3,4]
[print(i) for i in x]

Code :-
```
def multi(array, num):
    return [x * num for x in array]

array = []
iter = input("Enter Number of elements")
for i in range(0, int(iter)):
    array.append(int(input("Enter Elements :")))

num = int(input("Numbers to multiply"))
print(multi(array, num))
```

Conclusion :- We successfully studied about traversing method in python and WAP to multiply all the numbers in the list

Payoda 26/3/24 B

# Practical No. 10

GCOEN
Page No:
Date: / /

**Aim :-** Write a function that asks for an integer and prints a square of it. Use a while loop with a try, except, else block to account for incorrect inputs.

**Theory :-** Loops are used for iterating over a sequence like a list, tuple, dictionary, set or string) or other iterable objects. Iterating over a sequence is called traversal. The while loop is used to iterate over a block as long as the test condition is true. When the condition becomes false, the program control passes to the loop immediately following the loop.

**Try and Except :-** The try and except block in python is used for error handling catches and handles exceptions - errors that occurs during the execution of program.

The try block contains a segment of code that might raise an exception. If an exception is raised in the try block, the flow of control is passed to the except block, where the exception is handled. If no exception occurs the except block is stopped and normal flow continues. But if any exception occurs it is caught by except block.

Output :-

Please enter an integer : 4.2
That not a valid integer ! Please try again

Please enter an integer : 4
The square of number is 16.

Conclusion :- We have successfully understood error handling using try and except with help of the code.

In the code the while loop is used to keep for asking input until a value integer is provided. The try block has code that might raise a value error (when non-integer is entered). If error occurs, except block is executed and when it doesn't the else block is executed.

Code :-

```
def square_integer():
    while True:
        try:
            num = int(input("Please enter an integer : "))
        except ValueError:
            print("That is not a valid integer ! Please try again")
        else:
            print("The square of number is", num**2)
            break
square_integer()
```

Conclusion :- We have successfully understood error handling using try and except with help of the code.

Dayanka 16/4/24 A