**What is python?**

Python is one of the general purpose high level programming language, functional programming language, object oriented programming language and scripting programming language.

It is widely used in data science, machine learning and artificial intelligence.

It is easy to learn and require less code to develop the applications.

**What are the applications of python?**

The following are the various applications developed by python language:

1. Data visualization (graphs)
2. Web and internet development
3. Application related to automation with artificial intelligence
4. Application related to GUI (look and feel applications)
5. Applications related to scientific organizations (NASA)
6. Applications related to web scraping (web scraping is a process of extracting the required details from a particular website and store it in a local file system in the form of files)

The popular python web frameworks are Django, Pyramid, Flask.

Python’s scipy and numpy helps in scientific and computational application development.

Python’s tkinter library is used to create a desktop based GUI applications.

**What are the companies using python?**

Google, Yahoo, Youtube, Mozilla, Dropbox, Quora, Instagram, Microsoft, etc.

**Explain difference between map vs reduce vs filter?**

map(): For every element present in the given sequence, apply the function and generate new element.

map(function, sequence)

l **=** [1, 2, 3, 4, 5]  
**def doubleIt**(*x*)**:  
 return** 2**\****x*l1 **=** list(map(doubleIt, l))  
print(l1) *# [2, 4, 6, 8, 10]*

*# with lambda*l **=** [1, 2, 3, 4, 5]  
l1 **=** list(map(**lambda** x**:** 2**\***x,l))  
print(l1) *# [2, 4, 6, 8, 10]*

reduce(): it reduces sequence of elements into a single element, by applying function.

Reduce(function,sequence)

**from** functools **import** reduce  
l**=**[10,20,30,40,50]  
result**=**reduce(**lambda** x,y**:**x**+**y,l)  
print(result) *# 150*

filter(): it is used to filter values from the sequence based on some condition.

filter(function,sequence)

**def isEven**(*x*)**:  
 if** *x***%**2**==**0**:  
 return True  
 else:  
 return False**l**=**[0,5,10,15,20,25,30]  
l1**=**list(filter(isEven,l))  
print(l1) *# [0, 10, 20, 30]*

*# with lambda*l**=**[0,5,10,15,20,25,30]  
l1**=**list(filter(**lambda** x**:**x**%**2**==**0,l))  
print(l1) *# [0, 10, 20, 30]*

**Explain about assert statement?**

assert is a keyword which is used for evaluating one condition at a time.

**class Sathya:  
 def accept**(self)**:** self.n**=**int(input(**"enter a number between 1 to 10"**))  
 **assert** ((self.n**>=**1) **and** (self.n**<=**10)),**"don't enter a number which is not between 1-10"** print(**"u entered correct number: "**,self.n)  
  
s1**=**Sathya()  
**try:** s1.accept()  
**except** AssertionError **as** obj**:** print(obj)  
**except** ValueError**:** print(**"don't enter ANV/strings"**)  
**finally:** print(**"iam from finally block"**)

**What are lambda functions?**

Function without name is called as anonymous function.

To define anonymous functions we use a keyword lambda.

**def square**(*x*)**:  
 return** *x***\****x*print(square(2))  
  
*# with lambda*square**=lambda** x**:**x**\***x  
print(square(2))

max**=lambda** x,y**:**x **if** x**>**y **else** y  
print(max(10,20)) *# 20*

**What is list comprehension?**

list comprehension is way of creating list object based on existing iterable objects(like list,tuple,dictionary,range etc) based on some condititon.

list=[expression for item in list if condition]

*# by using for loop*h\_letters**=**[]  
**for** letter **in 'human':** h\_letters.append(letter)  
print(h\_letters) *# ['h', 'u', 'm', 'a', 'n']  
  
# with list comprehension*h\_letters**=**[letter **for** letter **in 'human'**]  
print(h\_letters) *# ['h', 'u', 'm', 'a', 'n']*

**What is \_\_init\_\_ functions?**

The purpose of constructor is to initialize the instance data members (IDM) of a class which are associated with an object.

The \_\_init\_\_ function is called a constructor or initializer and is automatically called when you create a object(new instance) of a class.

**What is a class?**

The purpose of class is to develop programmer defined data type (PDDT) and to develop real world applications.

A class is a collection of data members and methods.

**What is object?**

Instance of a class is called object (Instance is nothing but allocating the memory space for the data members and methods). (OR)

A grouped item is called an object.

**What are generators?**

Generator is a function which is responsible to generate a sequence of values. we can write generator functions just like ordinary functions, but it uses yield keyword to return values.

**def gen**()**:  
 yield 'A'  
 yield 'B'**g**=**gen()  
print(type(g))  
print(next(g))  
print(next(g))

**What are iterators?**

Iterator in python is simply an object, an object which will return data as one element at a time.

iterator object has two special methods \_\_iter\_\_() and \_\_next\_\_().

iter() function (which in turn calls \_\_iter\_\_() method) returns an iterator.

We use next() function to manually iterate all the items of an iterator. after last element if we try to use next() then we will get StopIteration error.

my\_list**=**[1,2,3,4]  
my\_iter**=**iter(my\_list)  
print(next(my\_iter))  
print(next(my\_iter))  
print(my\_iter.\_\_next\_\_())  
print(my\_iter.\_\_next\_\_())

**What is the difference between iterator and generator?**

Every generator is an iterator, but not vice versa.

iterators and generators are used for more memory efficient.

**What is inheritance and types of inheritance?**

The process of obtaining data members and methods of one class(base class or parent class) into another class(sub class or child class).

Types of inheritances:

1.single inheritance: contains only single base class and single derived class.

2.multilevel inheritance: contains single base class, single derived class and multiple intermediate base classes.

3.hierarchal inheritance: contains single base class and multiple derived classes.

4.multiple inheritance: contains multiple base classes and single derived class.

5.hybrid inheritance: it is a combination of available inheritance types.

**What is method resolution order?**

refer the notes for detail explanation.

**What is multithreading? what is GIL(Global interpreter lock) issue?**

The aim of multithreading is to provide concurrent execution(simultaneous/parallel execution).

Thread: A flow of control is known as thread.

The purpose of creating a thread is to execute the logic of the program which is written in the form of methods.

The python global interpreter lock is a lock that allows only one thread to be in a state of execution.

**What are decorators? how to create custom decorator?**

Decorator is a function which can take a function as argument and extend its functionality and returns the modified function with extended functionality.

**def decor**(*func*)**:  
 def inner**(*name*)**:  
 if** *name***=='sunny':** print(**"Hello sunny very good morning"**)  
 **else:** func(*name*)  
 **return** inner  
  
**@decor  
def wish**(*name*)**:** print(**"Hello"**,*name*,**"Good morning"**)  
  
wish(**'saisk'**)  
wish(**'sunny'**)  
  
output**:**Hello saisk Good morning  
Hello sunny very good morning

**How does python’s garbage collection work?**

Python’s memory allocation and deallocation is automatic, because of dynamic memory allocation.

Python uses two strategies for memory allocation:

1.Reference counting

2.Garbage collection

*# literal 9 is an object*b **=** 9  
  
*# reference count of object 9 becomes 0*b **=** 4

**What are regular expressions?**

A regular expression is one of the essential feature of python which is used for extracting or searching the given pattern in the given string data.

def: A regular expression is a collection of strings, chars, numbers, special symbols, etc.

**Differentiate between append() and extend() methods?**

append: adds the argument as a single element to the end of the list. The length of the list increases by one.

my\_list **=** [**'saisk'**]  
my\_list.append(**'sunny'**)  
print(my\_list) *# ['saisk', 'sunny']  
  
# we can append another list*my\_list **=** [**'saisk'**]  
another\_list **=** [**'sunny'**]  
my\_list.append(another\_list)  
print(my\_list) *# ['saisk', ['sunny']]*

extend: each element of the iterable gets appended to the old list.

my\_list **=** [**'saisk'**]  
another\_list **=** [**'sunny'**, **'mouli'**]  
my\_list.extend(another\_list)  
print(my\_list) *# ['saisk', 'sunny', 'mouli']*

**What is web scraping? How do you achieve it in python?**

The process of collecting information from web pages is called web scraping. In web scraping to match our required patterns like mail ids, mobile numbers we can use regular expressions.

**Explain the use of “with” statement in python?**

The with statement can be used while opening a file. The advantage of with statement is it will take care of closing the file, after completing all operations automatically even in the case of exceptions also.

**What are middlewares?**

middleware is the software that can link two separate applications together.

**What is the difference between remove() function and del statement?**

The del statement is used to remove list, dictionary or a key. We can also pass index to delete the element of list. del is faster way to remove elements from the list.

The remove() method is used to remove elements from the list. It searches the element before deleting. It is slower compared with del.

data **=** [1,2,3,4,5]  
**del** data *# which is used to remove the total list***del** data[0] *# removes element at o index*data.remove(5) *# removes 5 value from the list*print(data)

**Python string methods**

strip(): This method is used to delete all the leading and trailing characters.

lstrip(): This method is used to delete all leading characters.

rstrip(): This method is used to delete all the trailing characters.

s **= "----saisk----"**print(**"strip() used to delete all '-': "**, s.strip(**'-'**))  
print(**"lstrip() used to delete all leading '-': "**, s.lstrip(**'-'**))  
print(**"rstrip() used to delete all trailing '-': "**, s.rstrip(**'-'**))  
  
output**:**strip() used to delete all **'-':** saisk  
lstrip() used to delete all leading **'-':** saisk**----**rstrip() used to delete all trailing **'-': ----**saisk

min(“string”): returns the minimum value alphabet from string.

max(“string”): returns the maximum value alphabet from the string.

s **= "saisk"**print(**"minimum value:"**,min(s))  
print(**"maximum value:"**,max(s))  
  
output**:**minimum value**:** a  
maximum value**:** s

**Why do we use join() function in python?**

This method is used to concatenate a string with an iterable object. It returns a new string.

s **=** [**'s'**,**'a'**,**'i'**,**'s'**,**'k'**]  
print(**''**.join(s)) *# saisk*s1 **= "saisk"**s2 **= 'ab'**print(s1.join(s2)) *# asaiskb*