THEORY QUESTIONS ASSIGNMENT Software Stream

How does Object Oriented Programming differ from Process Oriented Programming?

Procedural programming is a programming model that evolved from structured programming and is based on the concept of invoking procedures. Procedures, often known as routines, subroutines, or functions, are essentially a set of instructions to be followed. Any procedure in a programme can be invoked at any time throughout its execution, including by other procedures or by itself.

Object-oriented programming (OOP) is a programming paradigm that is built on the concept of objects. Data is stored in the form of attributes, while code is stored in the form of methods. Computer programmes are built utilising the concept of objects that interact with the real world in object oriented programming. The most common object-oriented programming languages are class based

Languages used in Object Oriented Programming:

Java, C++, C#, Python,

PHP, JavaScript, Ruby, Perl,

Objective-C, Dart, Swift, Scala.

The programme is broken down into little chunks called functions in procedural programming. Object-oriented programming divides a programme into discrete pieces called objects. The top-down technique is used in procedural programming. The bottom-up method is used in object-oriented programming. In procedural programming, there is no access specifier. Access specifiers such as private, public, and protected are used in object-oriented programming. In procedural programming, adding new data and functions is more difficult than in object-oriented programming, because procedural programming lacks a good mechanism for concealing data, it is insecure. Data hiding is a feature of object-oriented programming, making it more safe. Overloading is not possible with procedural programming. In object-oriented programming, overloading is possible. In procedural programming, function takes precedence over data. Data is more important than function in object-oriented programming. Procedural programming is predicated on a reality that isn't real. Object-oriented programming (OOP) is grounded in reality.

What's polymorphism in OOP?

Polymorphism is an object-oriented programming language feature that allows a procedure to use variables of various types at different periods. Polymorphism is a programming language's ability to display the same interface for many underlying data types. It is a concept in computer science that objects of various types can be accessed through the same interface. This interface can be implemented in a variety of ways by each type. It's a fundamental notion in object-oriented programming (OOP).

A child class inherits all of the parent class's methods. However, in some cases, the parent class's inherited function does not fully fit within the child class. You'll have to re-implement the method in the child class in such circumstances. Polymorphism can be used in Python in a variety of ways. Polymorphism can be defined using a variety of functions, class methods, and objects. So, let's take a closer look at each of these strategies individually.

Functions and Objects Polymorphism

You can write a function that accepts any object, which allows for polymorphism.

class Tomato():

def type(self):

print("Vegetable")

def color(self):

print("Red")

class Apple():

def type(self):

print("Fruit")

def color(self):

print("Red")

def func(obj):

obj.type()

obj.color()

obj\_tomato = Tomato()

obj\_apple = Apple()

func(obj\_tomato)

func(obj\_apple)

Python employs two distinct class types in the same way. It is necessary to design a for loop that iterates through a tuple of elements. Furthermore, you must call the methods regardless of the object's class type. We take it for granted that each class has these methods.

class India():

def capital(self):

print("New Delhi")

def language(self):

print("Hindi and English")

class USA():

def capital(self):

print("Washington, D.C.")

def language(self):

print("English")

obj\_ind = India()

obj\_usa = USA()

for country in (obj\_ind, obj\_usa):

country.capital()

country.language()

In Python, polymorphism defines methods in the child class that have the same name as the parent class's methods. In inheritance, the methods of the parent class are passed down to the child class. It's also feasible to change a method that a child class has inherited from its parent. This is typically used when a method inherited from the parent class isn't appropriate for the child class. Method Overriding is the process of re-implementing a method in a child class. Here's an example of polymorphism in the context of inheritance.

Whats inheritance in OOP?

Inheritance allows us to create a class that inherits all of the functionality of a parent class while also allowing us to add new features.

In object-oriented programming, inheritance is a powerful feature.

It refers to creating a new class with minimal or no changes to an existing one. The new class is known as the derived (or child) class, whereas the one it inherits from is known as the base (or parent) class.

class BaseClass:

Body of base class

class DerivedClass(BaseClass):

Body of derived class

Derived classes inherit characteristics from the base class and can be extended with new ones. As a result, code can be reused.

Multiple Classes Inheritance

Multiple inheritance is supported by Python, which is one of the few modern computer languages that does it. The ability to derive a class from many base classes at the same time is known as multiple inheritance.

Multiple inheritance isn't supported by most modern programming languages. Modern programming languages, on the other hand, support the concept of interfaces. You inherit from a single base class in such languages and then implement various interfaces so that your class can be reused in diverse contexts. Your designs will be constrained by this method. Only one class's implementation can be inherited by directly inheriting from it. Multiple interfaces can be implemented, but multiple classes' implementations cannot be inherited.

If you had to make a program that could vote for the top three funniest people in the office, how would you do that? How would you make it possible to vote on those people?

Say there are six people in the office –

Clair

Phil

Emma

Rabia

Dua

Anah

b1 = Ballot(['Emma','Phil','Clair']

b2 = Ballot(['Rabia','Dua','Anah']

After making your ballots with for loops, randomization, add them to a ballot registry :

r = BallotRegistry()

r.addBallot(b1)

r.addBallot(b2)

v = VoteCount(r)

v.plurality()

v.instantRunoff()

v.bordaCount()

v.condorcet()

Within each of the VoteCount methods, a count \_register instance of the Ballot class is created within the function that is separate from the registry. Its candidates parameter is identical to that of all other ballots in the registry, but its votes parameter is not a rank. Instead, it's a list of each candidate's counts/scores/total votes. The count\_register is most useful at the end of each procedure, when the winner is selected by the candidate with the greatest count register score.

def plurality(self):

count\_register = self.\_\_makecountregister\_\_()

for b in self.registry.br:

for r in range(len(b.votes)):

if b.votes[r] == 1:

count\_register.votes[r] += 1

break

winner = 0

for c in range(len(count\_register.votes)):

if count\_register.votes[c] > count\_register.votes[winner]:

winner = c

print("Winner is", count\_register.candidates[winner])

What's the software development cycle?

The application of normal business procedures to the development of software applications is known as the Software Development Life Cycle. Planning, Requirements, Design, Build, Document, Test, Deploy, and Maintain are the six to eight processes that are usually followed. Depending on the scope of the project, some project managers will combine, split, or eliminate steps. These are the essential components for any software development project.

The SDLC is a method for evaluating and improving the development process. It enables for a fine-grained study of each process phase. As a result, businesses are able to maximise efficiency at each level. As processing capacity grows, the demand for software and developers grows as well. Companies must cut expenses, deploy software more quickly, and meet or exceed consumer expectations. SDLC aids in achieving these objectives.

Project leaders review the project's terms during the planning phase. Calculating labour and material expenses, developing a schedule with specific deadlines, and forming the project's teams and leadership structure are all part of this process.

Stakeholder feedback can be incorporated into the planning process. Anyone who stands to benefit from the application is referred to as a stakeholder. Obtain feedback from prospective consumers, developers, subject matter experts, and sales representatives.

The scope and objective of the application should be clearly defined during planning. It charts a course and equips the team to produce software efficiently. It also establishes limits to prevent the project from expanding or diverging from its initial goal.

Defining requirements is part of the planning process to figure out what the application is supposed to perform and what it needs. A social media programme, for example, would necessitate the ability to connect with a buddy. A search feature may be required by an inventory programme. The resources required to complete the project are also defined in the requirements.

Design and Prototyping

Programming language, industry norms, overall design, and use of any templates or boilerplate are all examples of architecture.

User Interface (UI) - Defines how customers interact with software and how it responds to input.

Platforms - These are the operating systems that the software will run on, such as Apple, Android, Windows, Linux, and even game consoles.

Programming - This term encompasses not only the programming language, but also the methods for solving problems and carrying out duties in the application.

Communications — Defines how the app can communicate with other assets, such as a central server or other instances of the app.

Security — Describes the safeguards put in place to keep the application safe.

Software Development -This is where the software is really written. A small project may be created by a single developer, whereas a large project may be divided into numerous teams. During this phase, use an Access Control or Source Code Management solution. Developers can use these tools to keep track of code modifications. They also assist in ensuring that different team initiatives are compatible and that target goals are met.

Testing –

It's vital to test an app before releasing it to the public. Much of the testing, such as security testing, can be automated. Other testing can only be done in a specific context; for complex deployments, consider developing a simulated production environment. Each function should be tested to ensure that it functions properly. The testing process assists in reducing the number of faults and glitches seen by consumers. As a result, there is a higher level of user satisfaction and a higher rate of utilisation.

Deployment-

The application is made available to users during the deployment phase. Many businesses prefer to have the deployment step automated. This might be as straightforward as a payment portal and download link on the company's website. It could also be the installation of an app on a smartphone. Deployment can be difficult as well. One example is migrating a company-wide database to a freshly designed application.

Operations and Maintenance

The development cycle is practically complete at this stage. The application has been completed and is currently being utilised in the field. However, the period of operation and maintenance is still crucial. Users find flaws that were missed during testing during this phase. These issues must be addressed, which may result in new development cycles.

What is the difference between agile and waterfall?

Agile and waterfall approaches are two distinct methodologies for completing projects or work items. Agile is an iterative methodology with a cyclic and collaborative approach. Waterfall is a sequential methodology that can be collaborative, but activities are usually handled in a more linear fashion.

The project will go through a series of cycles throughout the course of its lifespan, following the agile methodology. The work item's development, review, and feedback, followed by approval – yes or no. If so, carry out and finish the task. If not, make a note of it and make any necessary modifications, as well as track and alter the backlog or prioritising to reflect the new information.

The waterfall methodology simplifies the process of moving tasks through the steps of identifying requirements, designing the implementation, implementing the work item, verifying the implementation and quality assurance, and finally maintaining the feature.

What is a reduced function used for?

The reduce(fun,seq) function applies a certain function to all of the list components indicated in the sequence provided along. This function is defined in the "functools" module.

The first two elements of the sequence are chosen in the first step, and the result is achieved.

The result is then saved again after applying the same function to the previously obtained result and the number just preceding the second element.

This method is repeated until there are no more elements in the container.

The final result is returned to the console and printed.

Example

List = [2,4,6,8,19,]

print("The sum of the list elements is : ", end="")

print(functools.reduce(lambda a, b: a+b, lis))

print("The maximum element of the list is : ", end="")

print(functools.reduce(lambda a, b: a if a > b else b, lis))

What is a merge sort ?

A merge sort is a more complicated sort, yet it is also one of the most efficient.

Divide and conquer is a strategy used in a merge sort. The list is divided in half again and again until all of the elements are separated one by one. The elements are then compared, ordered, and combined in pairs. The process is then repeated until the entire list has been recompiled.

Consider this unsorted list:

2,4,6,8,10,12,14,16

An unsorted list of numbers

The list is split into half:

2 4 6 8 1 10 12 14 16

An unsorted list of numbers that has been split in half

The process repeats:

2 4 6 8 10 12 14 16

An unsorted list of numbers that has been split into four parts

2 4 6 8 10 12 14 16

Until all elements are individually separated

The programme examines each piece individually and compares them in pairs. Each pair is arranged in ascending order.

Following that, the pairs are compared, beginning with the first number in each pair. It is placed in order if the left hand number is less than the right hand number. The comparison then advances to the second number on the left side, and the procedure is repeated. If the right-hand number is less than the left-hand number, the comparison proceeds to the next number on that side.

Generator functions allow you to declare a function that behaves like an iterator, i.e. it can be used in a for loop. What is the use case?

All local variables are erased in a return statement, and the resulting value is given back (returned) to the caller. If the same function is called again at a later time, it will be given a new set of variables. However, if the local variables aren't thrown away when a function is exited, that means that we can pick up where we left off. The yield statement is what distinguishes a function as a generator. As a result, generators provide a simple yet effective method of producing iterators. They're constructed like conventional functions, but if they want to return data, they employ the yield statement. The generator resumes where it left off each time next() is called (it remembers all the data values and which statement was last executed).

def generate\_integers(N):

for i in xrange(N):

yield i

In [1]: gen = generate\_integers(3)

In [2]: gen

<generator object at 0x8117f90>

In [3]: gen.next()

0

In [4]: gen.next()

1

In [5]: gen.next()

Some more use cases for generators include

for loops which need to be paused and resumed at a later date

infinitely looping over an array and having it reset to the beginning once it's done

creating iterables to use in for of loops from non-iterable objects using [Symbol.Iterator]

Generators can also be used to loop through structures when numerous variables must be tracked at the same time. Instead of having numerous pieces of global state, you can just call the generator's.next() method.