Name:- AKSHAT CHUDASAMA

Roll no:- 13

Class:- FYCS

Subject:- Object Oriented Programming

Assignment No:- 1

Q1. Compare Procedure Orientated Programming (POP) and Object Orientation Programming (OOP) with diagram and features.

Answer:-

Procedure oriented Programming (POP):-

- In procedural programming, program is divided into small parts called functions.
- Procedural programming follows top down approach.
- There is no access specifier in procedural programming.
- Adding new data and function is not easy.
- In procedural programming, overloading is not possible.
- Examples: C, FORTRAN, Pascal, Basic etc.

Object Oriented Programming (OOP):-

- In object oriented programming, program is divided into small parts called objects.
- Object oriented programming follows bottom up approach.
- Object oriented programming have access specifiers like private, public, protected etc.
- Adding new data and function is easy.
- Overloading is possible in object oriented programming.
- Examples: C++, Java, Python, C# etc.

Features of POP:-

- It emphasis on algorithm.
- Function can communicate by global variable.
- Data move freely from one function to another function.
- It has predefined functions.

- It has a portable source code.

Features of OOP:-

- Encapsulation
- Polymorphism
- Inheritance
- Abstraction

Diagram of POP:-

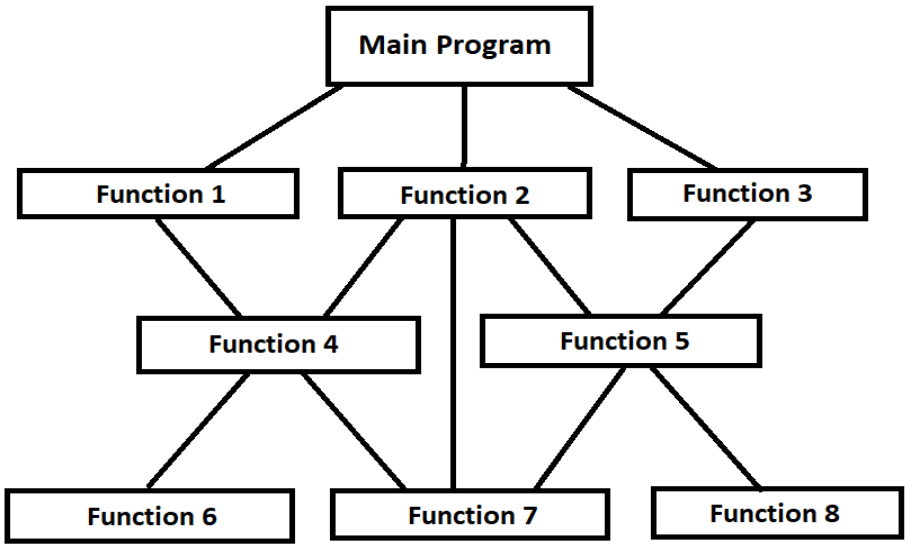
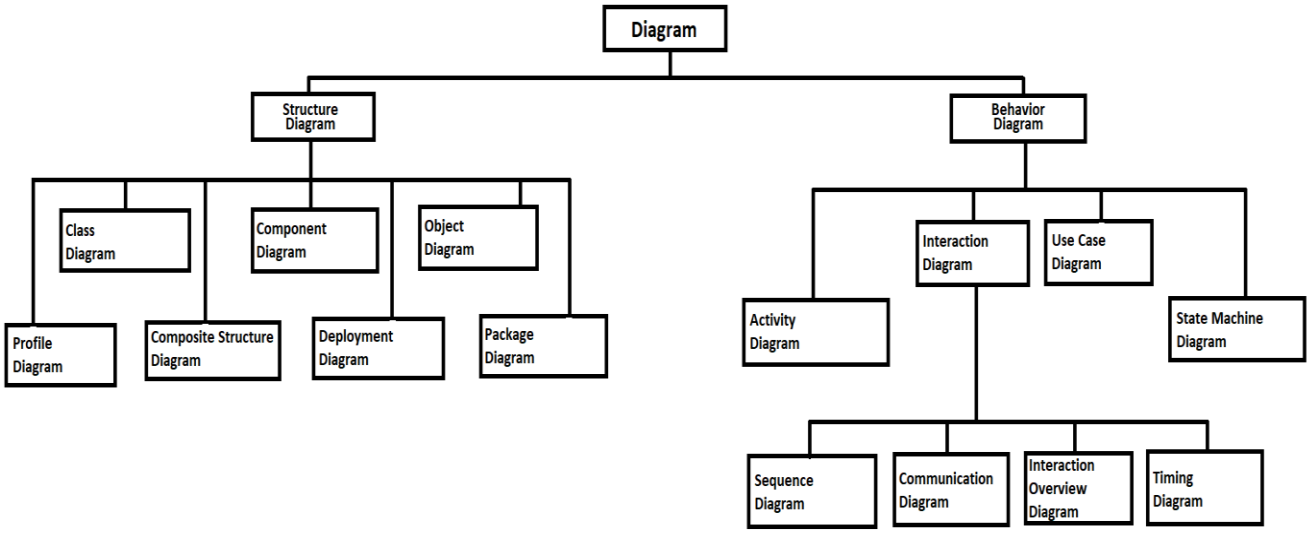


Diagram of OOP:-



Q2. Define OOP.

Answer:-

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic. An object can be defined as a data field that has unique attributes and behavior.

Object-Oriented Programming or OOPs refers to languages that use objects in programming. Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism, etc in programming.

OOP focuses on the objects that developers want to manipulate rather than the logic required to manipulate them. This approach to programming is well-suited for programs that are large, complex and actively updated or maintained.

Q3. Explain benefits of OOP.

Answer:-

1. Encapsulation Enforces Modularity:-

Encapsulation refers to the creation of self-contained modules that bind processing functions to the data. These user-defined data types are called "classes," and one instance of a class is an "object." For example, in a payroll system, a class could be Manager, and Pat and Jan could be two instances (two objects) of the Manager class. Encapsulation ensures good code modularity, which keeps routines separate and less prone to conflict with each other.

2. Inheritance:-

Classes are created in hierarchies, and inheritance allows the structure and methods in one class to be passed down the hierarchy. That means less programming is required when adding functions to complex systems. If a step is added at the bottom of a hierarchy, only the processing and data associated with that unique step needs to be added. Everything else is inherited. The ability to reuse existing objects is considered a major advantage of object technology.

3. Polymorphism:-

Object-oriented programming allows procedures about objects to be created whose exact type is not known until runtime. For example, a screen cursor may change its shape from an arrow to a line depending on the program mode. The routine to move the cursor on screen in response to mouse movement would be written for "cursor," and polymorphism allows that cursor to take on whatever shape is required at runtime. It also allows new shapes to be easily integrated.

Q4. List and explain any five application of OOP.

Answer:-

1. Real-Time Systems:-

Adopting the object-oriented technology in development real-time systems provides adaptation, ease of modifications, and reusability for such systems. Also, object-oriented methods are dominated by the need to deal with manipulating and dealing with the data that typically can be found in most of the information management systems.

2. Simulation and Modeling:-

Simulation with Object Oriented Programming (SW-OOP) is a library of software modules (classes) designed to provide the software constructs necessary for programming general purpose discrete event computer simulations in Pascal. Systems modeling and simulation has been cited by those disciplined in operations research and management science as one of the skills most frequently applied to the study, understanding and improvement of systems in industry

3. Object oriented databases:-

An object-oriented database (OODBMS) or object database management system (ODBMS) is a database that is based on object-oriented programming (OOP). The data is represented and stored in the form of objects. OODBMS are also called object databases or object-oriented database management systems. Object database management systems (ODBMSs) are based on objects in object-oriented programming (OOP). In OOP, an entity is represented as an object and objects are stored in memory. Objects have members such as fields, properties, and methods. Objects also have a life cycle that includes the creation of an object, use of an object, and deletion of an object. OOP has key characteristics, encapsulation, inheritance, and polymorphism. Today, there are many popular OOP languages such as C++, Java, C#, Ruby, Python, JavaScript, and Perl.

4. AI and Expert System:-

The expert systems are the computer applications developed to solve complex problems in a particular domain, at the level of extra-ordinary human intelligence and expertise. It contains domain-specific and high-quality knowledge. Knowledge is required to exhibit intelligence. The success of any ES majorly depends upon the collection of highly accurate and precise knowledge.

5. Neural Networks and parallel programming:-

The field of neural networks is being investigated by many researchers in order to provide solutions to difficult problems in the area of manufacturing systems. Computer simulation of neural networks is an important part of this investigation. This paper applies concepts from an important trend in software engineering research, namely object-oriented programming, to model neural networks.

Q5. What is the difference between objects and class, How they are related?

Answer:-

- Objects:-

Object is an instance of a class. An object in OOPS is nothing but a self-contained component which consists of methods and properties to make a particular type of data useful. For example color name, table, bag. When you send a message to an object, you are asking the object to invoke or execute one of its methods as defined in the class. From a programming point of view, an object in OOPS can include a data structure, a variable, or a function. It has a memory location allocated. Java Objects are designed as class hierarchies. An object in OOPS is a specimen of a class. Software objects are often used to model real-world objects you find in everyday life.

- Class:-

Class are a blueprint or a set of instructions to build a specific type of object. It is a basic concept of Object-Oriented Programming which revolve around the real-life entities. Class determines how an object will behave and what the object will contain. A Class in object oriented programming is a blueprint or prototype that defines the variables and the methods (functions) common to all Java Objects of a certain kind.

- Relation between object and class:-

A class defines the properties and behavior for the objects represented by the abstraction. Abstraction is a property of object oriented programming. It denotes the essential properties and behaviors of an object. It hides code and data. A class thus denotes a category of objects and act as a blueprint for creating such objects. An object exhibits the property and behaviors defined by its class. Generally, an object is an instance of a class.

Q6. What is UML, list various UML diagrams and explain the components of class diagram?

Answer:-

Unified Modeling Language (UML) is a general purpose modeling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering. UML is not a programming language, it is rather a visual language. We use UML diagrams to portray the behavior and structure of a system. UML helps software engineers, businessmen and system architects with modeling, design and analysis.

UML is linked with object oriented design and analysis. UML makes the use of elements and forms associations between them to form diagrams. Diagrams in UML can be broadly classified as:

1. Structural Diagrams:- Capture static aspects or structure of a system. Structural Diagrams include: Component Diagrams, Object Diagrams, Class Diagrams and Deployment Diagrams.
2. Behavior Diagrams:- Capture dynamic aspects or behavior of the system. Behavior diagrams include: Use Case Diagrams, State Diagrams, Activity Diagrams and Interaction Diagrams.

Structural UML diagrams:-

1. Class Diagram
2. Composite Structure Diagram
3. Object diagram
4. Component diagram
5. Deployment diagram
6. Package diagram

Behavior Diagrams:-

1. State Machine Diagrams
2. Activity Diagrams
3. Use case diagrams
4. Communication diagram
5. Timing diagram
6. Sequence diagram
7. Interaction overview diagram

Class Diagram:-

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Q7. What is the use of self parameter in any method in python?

Answer:-

Self represents the instance of the class. By using the “self” keyword we can access the attributes and methods of the class in python. It binds the attributes with the given arguments.

The reason you need to use self is because Python does not use the syntax to refer to instance attributes. Python decided to do methods in a way that makes the instance to which the method belongs be passed automatically, but not received automatically the first parameter of methods is the instance the method is called on.

Example (Code):-

```
class Arithmetic:
    def accept (self):
        print ("Enter first number:")
        self.a = int(input())
        print ("Enter second number:")
        self.b = int(input())

    def sum (self):
        addition = self.a + self.b
        print ("Sum is", addition)

    def sub (self):
        subtraction = self.a - self.b
        print("Subtraction is", subtraction)

    def multi (self):
        multiplication = self.a * self.b
        print ("Multiplication is", multiplication)

    def div (self):
        division = self.a / self.b
        print ("Division is", division)

    def join (self):
        joining = str(self.a) + str(self.b)
        print("Concatination is", joining)
```

```
a = Arithmetic()
a.accept()
a.sum()
a.sub()
a.multi()
a.div()
a.join()
```

Output:-

```
Enter first number:
5
Enter second number:
6
Sum is 11
Subtraction is -1
Multiplication is 30
Division is 0.8333333333333334
Concatination is 56
```