Question 1:- Define Object Oriented Programming Language?

**ANSWER:-**

Object-oriented programming language (OOPL) is a high-level programming language based on the object-oriented programming (OOP) model. Refers to a programming methodology based on objects, instead of just functions and procedures. These objects are organized into classes, which allow individual objects to be grouped together.

An "object" in an OOP language refers to a specific type, or "instance," of a class. Each object has a structure similar to other objects in the class, but can be assigned individual characteristics. An object can also call functions, or methods, specific to that object. For example, the [source code](https://techterms.com/definition/sourcecode) of a video game may include a class that defines the structure of characters in the game. Individual characters may be defined as objects, which allows them to have different appearances, skills, and abilities. They may also perform different tasks in the game, which are run using each object's specific methods.

Object-oriented programming makes it easier for programmers to structure and organize [software](https://techterms.com/definition/software) programs. Because individual objects can be modified without affecting other aspects of the program, it is also easier to update and change programs written in object-oriented languages. As software programs have grown larger over the years, OOP has made developing these large programs more manageable.

Most modern programming languages are object-oriented or support the OOP model to an extent. Popular OOPLs include Java, C++, Python and SmallTalk.

Question 2:-List down the Benefits of OOP?

**ANSWER:-**

1. Re-usability:-

It means reusing some facilities rather than building it again and again. This is done with the use of a class. We can use it ‘n’ number of times as per our need.

2. Data Redundancy:-

This is a condition created at the place of data storage (you can say Databases) where the same piece of data is held in two separate places. So the data redundancy is one of the greatest advantages of OOP. If a user wants a similar functionality in multiple classes he/she can go ahead by writing common class definitions for the similar functionalities and inherit them.

3. Code Maintenance:-

This feature is more of a necessity for any programming languages, it helps users from doing re-work in many ways. It is always easy and time-saving to maintain and modify the existing codes with incorporating new changes into it.

4. Security:-

With the use of data hiding and abstraction mechanism, we are filtering out limited data to exposure which means we are maintaining security and providing necessary data to view.

5. Design Benefits**:-**

If you are practicing on OOPs the design benefit a user will get is in terms of designing and fixing things easily and eliminating the risks (if any). Here the Object Oriented Programs forces the designers to have a longer and extensive design phase, which results in better designs and fewer flaws. After a time when the program has reached some critical limits, it is easier to program all the non-OOP’s one separately.

6. Better Productivity**:-**

with the above-mentioned facts of using the application definitely enhances its users overall productivity. This leads to more work done, finish a better program, having more inbuilt features and easier to read, write and maintain. An OOP programmer cans stitch new software objects to make completely new programs. A good number of libraries with useful functions in abundance make it possible.

7. Easy Troubleshooting**:-**

Lets witness some common issues or problems any developers face in their work.

Is this the problem in the widget file?

Is the problem is in the WhaleFlumper?

Will I have to trudge through that ‘sewage.c’ file?

Commenting on all these issues related to code.

So, many a time it happens that something has gone wrong which later becomes so brainstorming for the developers to look where the error is. Relax! Working with OOP language you will know where to look for. This is the advantage of [using encapsulation](https://www.educba.com/abstraction-vs-encapsulation/) in OOP; all the objects are self-constrained. With this modality behavior, the IT teams get a lot of work benefits as they are now capable to work on multiple projects simultaneously with an advantage that there is no possibility of code duplicity.

8. Polymorphism Flexibility**:-**

Let’s see a scenario to better explain this behavior.

You behave in a different way if the place or surrounding gets change. A person will behave like a customer if he is in a market, the same person will behave like a student if he is in a school and as a son/daughter if put in a house. Here we can see that the same person showing different behavior every time the surroundings are changed. This means polymorphism is flexibility and helps developers in a number of ways.

* It’s simplicity Extensibility

9. Problems Solving**:-**

Decomposing a complex problem into smaller chunks or discrete components is a good practice. OOP is specialized in this behavior, as it breaks [down your software code](https://www.educba.com/hardware-vs-software/) into bite-sized – one object at a time. In doing this the broken components can be reused in solutions to different other problems (both less and more complex) or either they can be replaced by the future modules which relate to the same interface with implementations details.

Question 3:-Differentiate between function and method?

**ANSWER:-**

|  |  |
| --- | --- |
| A **function** is a piece of code that is called by name. It can be passed data to operate on (i.e. the parameters) and can optionally return data (the return value). All data that is passed to a function is explicitly passed. | A **method** is a piece of code that is called by a name that is associated with an object. In most respects it is identical to a function except for two key differences:   1. A method is implicitly passed the object on which it was called. 2. A method is able to operate on data that is contained within the class (remembering that an object is an instance of a class - the class is the definition, the object is an instance of that data). |

Question 4:-Define the following terms:

1. Class 2. Object 3. Attribute 4. Behavior

**ANSWER:-**

**CLASS**  A blueprint created by a programmer for an object. This defines a set of attributes that will characterize any object that is instantiated from this class.

**OBJECT** An instance of a class. This is the realized version of the class, where the class is manifested in the program.

**ATTRIBUTES** Instance attributes are owned by the specific instances of a class. This means for two different instances the instance attributes are usually different.   
  
We can also define attributes at the class level. Class attributes are attributes which are owned by the class itself. They will be shared by all the instances of the class. Therefore they have the same value for every instance. We define class attributes outside of all the methods, usually they are placed at the top, right below the class header.

**BEHAVIOR** In software, a *behavior* is how a feature operates within a well-defined scenario of inputs, actions, and outcomes. Products can exhibit countless behaviors, such as:

* Submitting forms on a website
* Searching for desired results
* Saving a document
* Making REST API calls
* Running command-line interface commands