***Assignment # 6***

**Question 1: Define Object Oriented Programming Language.**

**Object-oriented Programming**

Object-oriented Programming (OOP) is a [programming paradigm](http://en.wikipedia.org/wiki/Programming_paradigm) which provides a means of structuring programs so that properties and behaviors are bundled into individual *objects*. For instance, an object could represent a person with a name property, age, address, etc., with behaviors like walking, talking, breathing, and running. Or an email with properties like recipient list, subject, body, etc., and behaviors like adding attachments and sending.

In other words, object-oriented programming is an approach for modeling concrete, real-world things like cars as well as relations between things like companies and employees, students and teachers, etc. OOP models real-world entities as software objects, which have some data associated with them and can perform certain functions.

Moreover, object-oriented programming, is a well-adopted programming style that uses interacting objects to model and solve complex programming tasks. Two examples of popular object-oriented programming languages are Java and C++. Some other well-known object-oriented programming languages include Objective C, Perl, Python, Javascript, Simula, Modula, Ada, Smalltalk, and the Common Lisp Object Standard.

**Question 2: List down the Benefits of OOP.**

**Benefits of OOP**

Following are the few benefits of OOP:

1. **Improved software-development productivity:** Object-oriented programming is modular, as it provides separation of duties in object-based program development. It is also extensible, as objects can be extended to include new attributes and behaviors. Objects can also be reused within an across applications. Because of these three factors – modularity, extensibility, and reusability – object-oriented programming provides improved software-development productivity over traditional procedure-based programming techniques.
2. **Improved software maintainability:** For the reasons mentioned above, object-oriented software is also easier to maintain. Since the design is modular, part of the system can be updated in case of issues without a need to make large-scale changes.
3. **Faster development:** Reuse enables faster development. Object-oriented programming languages come with rich libraries of objects, and code developed during projects is also reusable in future projects.
4. **Lower cost of development:** The reuse of software also lowers the cost of development. Typically, more effort is put into the object-oriented analysis and design, which lowers the overall cost of development.
5. **Higher-quality software:** Faster development of software and lower cost of development allows more time and resources to be used in the verification of the software. Although quality is dependent upon the experience of the teams, object-oriented programming tends to result in higher-quality software.

Question 3: Differentiate between function and method?

|  |  |  |
| --- | --- | --- |
| **Definition** | **FUNCTION** | **METHOD** |
| 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). | A **method** is a piece of code that is called by a name that is associated with an object. |
| 1. | Functions do not have any reference variables. | Methods are called by reference variables. |
| 2. | All data that is passed to a function is explicitly passed. | It is implicitly passed the object for which it was called. |
| 3. | It does not have access controlling i.e. Function(other than static functions) declared and defined anywhere in the code | It has access controlling i.e. Method should be declared and defined in the class only. |
| 4 | Function applies to both object oriented and non-object oriented language such as procedural language (e.g. C) or scripting language (e.g. Javascript) | Method is only applicable to object oriented programming language such as C++, C#, Java etc. |

**Question 4: Define the following terms:**

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

A class describes the contents of the objects that belong to it, it describes an aggregate of data fields (called instance variables), and defines the operations (called methods).

Classes are used to create new user-defined data structures that contain arbitrary information about something. In the case of an animal, we could create an Animal() class to track properties about the Animal like the name and age.

It’s important to note that a class just provides structure—it’s a blueprint for how something should be defined, but it doesn’t actually provide any real content itself. The Animal() class may specify that the name and age are necessary for defining an animal, but it will not actually state what a specific animal’s name or age is.

It may help to think of a class as an idea for how something should be defined.

1. **Object**

An object is an element (or instance) of a class; objects have the behaviors of their class. The object is the actual component of programs, while the class specifies how instances are created and how they behave.

While the class is the blueprint, an Object (instance) is a copy of the class with actual values, literally an object belonging to a specific class. It’s not an idea anymore; it’s an actual animal, like a dog named Roger who’s eight years old.

For example, a class is like a form or questionnaire. It defines the needed information. After filling out the form, a specific copy is an instance of the class; it contains actual information relevant to specific information. Hence, multiple copies can be created for many different instances, but without the form as a guide, what information is required cannot be known. Thus, before creating individual instances of an object, its specification is needed by defining a class.

1. **Attributes**

Attributes are the individual things that differentiate one object from another and determine the appearance, state, or other qualities of that object. Let's create a theoretical class called Motorcycle. A motorcycle class might include the following attributes and have these typical values:

* Color: red, green, silver, brown
* Style: cruiser, sport bike, standard
* Make: Honda, BMW, Bultaco

Attributes of an object can also include information about its state; for example, you could have features for engine condition (off or on) or current gear selected.

Attributes are defined in classes by variables. Those variables' types and names are defined in the class, and each object can have its own values for those variables. Because each instance of a class can have different values for its variables, these variables are often called instance variables.

1. **Behavior**

A class's behavior determines how an instance of that class operates; for example, how it will "react" if asked to do something by another class or object or if its internal state changes. Behavior is the only way objects can do anything to themselves or have anything done to them. For example, to go back to the theoretical Motorcycle class, here are some behaviors that the Motorcycle class might have:

* Start the engine
* Stop the engine
* Speed up
* Change gear
* Stall

To define an object's behavior, you create methods, a set of Java statements that accomplish some task. Methods look and behave just like functions in other languages but are defined and accessible solely inside a class. Java does not have functions defined outside classes (as C++ does).

**Question 5: Write a code in python in which create a class named it Car which have 5 attributes such like (model, color and name etc.) and 3 methods. And create 5 object instance from that class.**

(Python note book attached)