Java vs Python OOP Concepts

For execution of this laboratory work I chose two OOP languages,one which is considered more purer OOP - **Java** , and another which is more a mixture of Object Oriented and procedure oriented language - **Python**.

I will start with comparison of the first OOP concept - **Encapsulation**, which means putting together all the instance variables and the methods into a single unit called Class. Encapsulation provides the security that keeps data and methods safe from inadvertent changes. **Java** allows you to apply encapsulation at the level of individual class members. The programmer gets to see what external software sees each field or method by using one of the following access modifiers: `private`, `protected`, `public`. **Python**'s approach to encapsulation is slightly different.Python does **NOT** have a **strong encapsulation**. There's no public, private, protected keyword in python.Anything that starts with two underscores is private to the class, others which start with one underscore is private and everything else is public. However, Python doesn't really enforce encapsulation strongly like Java.

Polymorphism is an important definition in OOP. Absolutely, we can realize polymorphism in Python just like in JAVA.However there are some differences in accessibility of Python and Java.

- Java is a static and strong type definition language.In Java, all variable names (along with their types) must be explicitly declared. Attempting to assign an object of the wrong type to a variable name triggers a type exception.That's what it means to say that Java is a statically typed language.
- While Python is a dynamic and weak type definition language. In Python, you never declare anything. An assignment statement binds a name to an object, and the object can be of any type.That's what it means to say that Python is a dynamically typed language.

Inheritance in **Python** is simple, just **like JAVA**, subclass can invoke attributes and methods in superclass.In my code examples the superclass is Mobile, from which are extended the classes Android and BlackBerry, which have actually the same properties as the parent class have. The only difference between them is that Java doesn't support multiple inheritance,because multiple inheritance can cause ambiguity in few scenarios.