**Test Ereditarietà e polimorfismo**

!) Che cosa è l’ereditarietà? spiega in un breve?

L’ ereditarietà è un meccanismo molto utile utilizzabile in Java. Consente di massimizzare il riutilizzo del codice in quanto permette di definire una classe più generale e di definire in seguito classi specializzate che aggiungono nuovi dettagli alla classe più generale. La classe specializzata eredita tutte le proprietà della classe generale ed è necessario solo realizzare le nuove caratteristiche.

2) Cosa si utilizza per definire una sotto classe?

* Extend √
* Implemet
* Abstract

3) Scrivere la sintassi di un costruttore

Il costruttore deve richiamare un costruttore della superclasse usando la parola chiave super() :

super([parametri]);

Esempio:

public NomeClasse(int a, double b, String c) {

super(a, b);

…

4) Cosa succede se la super classe non ha alcun costruttore?

Se la superclasse ha il costruttore con zero parametri, viene invocato. Se invece la superclasse non ha il costruttore allora viene provocato un errore di compilazione.

5) Cosa significa overriding in java?

Overriding significa sovrascrivere/ridefinire un metodo presente nella superclasse utilizzando lo stesso nome e gli stessi parametri. E’ possibile però cambiare il modificatore d’ accesso che può però solo rendere più “permissivo” l’ accesso al metodo.

6) Che cosa è la classe object e cosa fa?

La classe Object è la classe da cui tutte le altre ereditano per default. Non ha attributi e possiede metodi come il toString ed equals oltre a molti altri. Per poter utilizzare però questi metodi è necessario farne l’ Override quindi ridefinirli.

7) Come possiamo disabilitare l’ereditarietà di una classe?

Per disabilitare l’ ereditarietà di una classe è necessario dichiarare la classe con il modificatore final. Così facendo si impedisce la creazione di sottoclassi a partire dalla classe dichiarata final e i metodi (anch’ essi possono essere impostati final) non potranno essere ridefiniti.

8) Che cosa è il polimorfismo?

Il polimorfismo è la possibilità di associare più significati a una entità sfruttando il binding dinamico. Consente di fare riferimento ad oggetti di classi diverse mediante la stessa entità e di svolgere la stessa azione in modi diversi a seconda della particolare istanziazione.

9) Quale delle due funzioni viene chiamata?

Auto a = new Auto();

Modello m = a;

m.annoDiFabricazione(2010);

|  |
| --- |
| Auto |
| annoDiFabricazione |

|  |
| --- |
| Modello |
| annoDiFabricazione |

Viene chiamata la funzione di Auto in quanto la funzione m.annoDiFabbricazione è un override della funzione “originale” che è annoDiFabbricazione di Auto

10) è necessario aggiungere nuove classi per modificare quelle già presenti?

* Si
* No √

motivare il perché.

Aggiungere nuove classi non necessita mai di modificare quelle preesistenti. Ciò è dovuto all’ utilizzo del meccanismo del polimorfismo e dell’ ereditarietà che consente facilmente di ampliare le applicazioni che utilizzano un paradigma Object Oriented