Subclass inherit all (automatically) the instance variables and methods, but the subclass can add new instance variable and methods.
Intance variable is not overriden because they don't need to be.
common superclass
(1)look for obkects that have common attrbutes and behaviors
(2)Design the class that represent the common state and behaviors
(3)Decede if a superclass needs behaviors that are specific to that particular subclass type
(4)Use abstract more
When you call a method on an object reference, you're calling the most specific version of the method for that object type.
"Lowest" on the inheritance tree.
OneFromSubclass is OneFromSuperclass, and can do all the things OneFromSuperclass can do
Access level controls who sees what
(public protected default private)
When a subless inherit a viriable, it is asif the subless defined the member itself.

DO use inheritance when one class is a more specific type of a superclass. Example: Willow *is a* more specific type of Tree, so Willow *extends* Tree makes sense.

DO consider inheritance when you have behavior (implemented code) that should be shared among multiple classes of the same general type.

DO NOT use inheritance just so that you can reuse code from another class, if the relationship between the superclass and subclass violate either of the above two rules.

DO NOT use inheritance if the subclass and superclass do not pass the IS-A test. Always ask yourself if the subclass IS-A more specific type of the superclass. Example: Tea IS-A Beverage makes sense. Beverage IS-A Tea does not.

- A subclass *extends* a superclass.
- A subclass inherits all *public* instance variables and methods of the superclass, but does not inherit the *private* instance variables and methods of the superclass.
- Inherited methods *can* be overridden; instance variables *cannot* be overridden (although they can be *redefined* in the subclass, but that's not the same thing, and there's almost never a need to do it.)
- Use the IS-A test to verify that your inheritance hierarchy is valid. If X *extends* Y, then X *IS-A* Y must make sense.
- The IS-A relationship works in only one direction. A Hippo is an Animal, but not all Animals are Hippos.
- When a method is overridden in a subclass, and that method is invoked on an instance of the subclass, the overridden version of the method is called. (*The lowest one wins.*)
- If class B extends A, and C extends B, class B IS-A class A, and class C IS-A class B, and class C also IS-A class A.

A: One of the key benefits of inheritance is to minimise the amount of duplicate code in an application by sharing common code amongst several subclasses.
Inheritance can also make application code more flexible to change because classes that inherit from a common superclass can be used interchangeably.