Assignment 4.5

1. Can we achieve run time polymorphism through data members?

* No. Runtime polymorphism is achieved through overriding METHODS, not the data members.

1. Can you declare a class as private?

* You can do this. This is typically done for inner or nested classes and not a class with a main method, however.

1. What is the difference between Abstraction and Encapsulation?

* They are both similar in the fact that they are both “hiding” something, however their intentions are where they differ. With abstraction you are creating a method without a body to be implemented later when it is needed across other classes. With encapsulation you are hiding the content within your variables by making them private but are often still keeping your getters and setters as public so that other developers would still be able to

1. Method Overloading rules? Can we overload the super class method in a sub class?

* When overloading a method you must have the same method name but different method parameters. The return type can be different as well as the access modifiers. A subclass is definitely able to use its super class’s overloaded method. For example:
* Public class Animal {
* String noise;
* Int legs;
* Public Animal(String noise, int legs) {
* This.noise = noise;
* This.legs = legs;
* }
* \*getters and setters\*
* Public class Dog extends Animal {
* String fur;
* Public Dog(String fur) {
* Super(“bark”, 4)
* This.fur=fur;
* }
* A brief example of using a super classes constructor in a sub class.

1. Method Overriding rules:

* The method in the child class must have the same signature as the method in the parent class. The method in the child class must have at least the same or more accessibility (access modifiers) than the parent class, and the return type must be the same.

1. A. it will fail to compile. The test class needs to be in its own class.
2. The **protected** access modifier allows for access in the same package AS WELL AS a subclass
3. A
4. D
5. A and C