1. a

Logic: Amount is a static variable which is not associated with any instance. However, in this specific case “this” keyword should be used inside any of the instance methods, not from a static method like main. So, considering all this will generate compiler error.

2. d

Logic: Both SuperClass and MyClass have the getType method implementation and MyClass basically overrides the behavior of SuperClass implementation. As the instance was created from MyClass (though assigned to SuperClass instance), call to getType method will invoke the MyClass implementation following polymorphic nature (runtime polymorphism).

3. c

Logic: Both SuperClass and MyClass have the getType method implementation and MyClass basically overrides the behavior of SuperClass implementation. As the instance was created from SuperClass and assigned to SuperClass instance, the call to getType method will invoke the SuperClass implementation following polymorphic nature.

4. a

Logic: MyInnerClass is an inner class of outer class MyClass. So, the outer class will be able to access any inner class property by creating the inner class instance. So, the call should be like “**this**.**new** MyInnerClass().value” instead from direct accessing the value property;

5. c

Logic: MyClass creates an instance of its MyInnerClass before invoking the inner class method using that inner class instance which is the right approach. Moreover, the inner class can access outer class variables directly (even private variables) and as a result nothing wrong accessing the “value” property of outer class inside inner class instance methods. So, the program will generate expected output.

6. a

Logic: Static inner class can access static variables and methods of outer class. But in this case the “compute” method inside the static inner class tries to access the member variable of outer class which is the violation of rules, hence will generate compile time error.

7. a

Logic: Inner class can access all the variables of outer class (even private variables). However, any subclass doesn’t have access to its superclass’s private variable which indicates those superclass private variables are not part of subclass implementation. As a result, the inner class of the subclass will not have access to its outer class’s superclass private variable. So, this program will generate compile time errors.

8. d

Logic: As any superclass constructor will be invoked first even when the subclass instance will be created, so in this case will fall into a cyclic loop and will print nothing, rather will generate stack overflow exceptions at some point of time.

9. c

Logic: Considering all the logics and access rights mentioned below, a call to the ‘run’ function will generate the result

i) TheInnerClass is an inner class of TheClass and so will have access to all the properties of TheInnerClass.

ii) TheSubClass inherits from TheClass and so will have all the public and protected level access to its properties and methods.

iii) Moreover, TheSubInner is inherited from TheInnerClass, so will have public and protected level access to its properties and methods, and also an inner class of TheSubClass and so will have access to TheSubInner class private properties.

10. d

Logic: This is basically an anonymous implementation of Manager class inherited from Employee. So, it has access to all the public and protected properties and methods. In addition, having its own implementation of bonus logic and computation of salary. As a result, this program will generate and print manager object information to the console.