Test A

1) "local" → new Astring object created in test().

2) "a new value" → aString value in test() passed to method1(), new string object of astring created and the new value given which gets printed.

3) "Local" → as ~~method~~ no value is returned and no reference to the value of aString is referenced, the aString value refered

Test B

1) "initial" → this. keyword is used to refer instance variable of a class.

2) "a new value" → same as Test A 2) but the value passed is "initial.

3) "initial" → as method1 doesn't ~~assi~~ change the value of this.aInt the value printed is "initial".

Test C

1 "Local" → Reference made to aString object in method test().

2 "new value" → changes the value of the instance variable of class from "initial" to "new Value" and prints it.

3. "Local" → Reference made to aString object in method tes().

Test D

1) "new value" → Test C-2) changed the value of instance variable and hence this.aString refers to the ~~ne~~ new value.

2) "new value" → we pass the same value of aString to method2 and it assigns the same and prints the given literal.

3) "new value" → as this keyword is used the reference is made again to the new value assigned in method2 .

Pest E

1) "Local" → Pill now no changes have been made to aString in test(). so value remains same.

2) "yet a new value" → method 3 creates a new String with a new value and that value is assinged to aString and printed.

8) "Local" → As no referene or return is made from any methods the value remains local.

Pest F

1) "new value" → the value of field variable aString was changed in Test D-2) and we print the same using this ~~[illegible]~~ keyword.

~~2) "new value" →~~

2) "yet a new value" → eventhough this.aString is passed we change ~~the~~ value of aString locally to the new value and print it

3) "new value" → As all the changes made were local the value of the class instance variable doesnt change.

Test G :

1) "local" → Refference to the local variable of test()

2) "new value" → The parameter passed in method4 ~~take~~ is assigned the value of ~~the~~ the instance variable and printed locally

3) "local" → as the above operation was local and no return or reference to method4 is made value of aString remains local to test()

Test H

1) "new value" → we directly print the class instance variable value

2) "new value" → we pass the ~~ve~~ ~~class~~ instance variable of the class to method4 where it locally assigns aString the value of thisaString which is the same the output remains same

3) "newvalue" → We refer the instance variable which still has the same value.