1.
    A. → double num1, int num2 = 0;

    Since double and int are different types we cannot initiate them together.

2. D → The code does not compile
   As we see in this code chair variable was not initialized.
   So it does not compile.
3. B → It defaults to null
4. B → In java variables cannot start with numeric.
5. B → Class names should be nouns, in mixed case with the first letter of each internal word capitalized.
6. C → Since in java primitive types do not call toString() method first one cannot compil
7. Over java 1.7.3 compiler we can compile all of them. But under this version it is not possible
8. C → Integer is derivative of Object class that contains pritimive types.
9. C → The code does not compile. Because there is no type like integer. We should create it before compiling.
10. C → When we use <<new>> keyword it start to instantiates new object that places in
11. D p4. Double is 8 bytes but float 4 bytes. So it is not possible to convert double to float
12. C → Char, byte, float, double
13. D → all of 'em are correct.
14. B → int and double are different primitive types. We should seperate them.
15. B → only static initializers can be done without initialize Bowling object.
16. A → node. Since we did not initialize any of them.
17. C → finalize method run as a once time
18. C → Long is primitive type
19.
20. C → if we would like to create float it should be 3.14f;
21. B → we cannot give the variable name as a primitive types
22. B → bar is a global variable. We cannot call any local variable.
23. C → java classes cannot start with numeric
24. **B → Double instance can be converted primitive type**
25. B → It defaults to null
26. C → double 0.0
27. C → you can convert a wrapper class object to primitive
28. C → the code does not compile. Because we cannot call methods with primitives.
29. D → new TennisBall(): when use new keyword it first call constructer.
30. C → I and III
31. C→ char : Character and int: Integer
32. A → they can be set to null
33. A → Primitive types begin with a lowercase letter
34. B → Call System.gc()
35. **C → fruit3 = fruit1 then we dont need to fruit1; fruit2 = fruit3; then we dont need fruit3.**
36. B → double, Double
37. B → constructor
38. C → we cannot assing null to primitive
39. C → Blank !: an instance method only, Blank 2: an instane or static method
40. B → underscores can define between numerics
41. A → short 2 bytes, int 4 bytes, long 8 bytes
42. A → cat.name

43. Play-play-
44. A→ p.beakLnght = b;
45. ..
46. D → Three
47. C → public TennisBall() {}
48. ..
49. B → Call the constructor of the wrapper class. We know that we cannot call method via primitive types
50. C → aab: first the constructer is called and second also constructer is called for new object And last method is called