1. **Program**

```
class Hello {
        public static void main(String args[])
        {
          System.out.println("Hello World!");
        }
}
```

2. **Program**

```
class HelloAgain {
        public static void main(String[] args)
        {
        System.out.println("Hello Again");
        }
        }
```

3. **Program**

```
class HelloAgainAndAgain {
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        System.out.println("Hello World!");
        System.out.println("Hello World!");
        System.out.println("Hello World!");
        }
}
```

4. **Program**

```
class  MyClass {
        public static void main(String args[])
        {
        System.out.println(100);
        }
}
```

5. **Program**

```
        class Manager  {
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        System.out.println(100);
                System.out.println(20.909);
                System.out.println(true);
                System.out.println(10 + 20);
                System.out.println(20 == 20);
        }
}
```

6. **Program**

```
class A
{
        public static void main(String[] args)
        {
                int i=0;
                System.out.println(i);
        }
}
```

7. **Program**

```
class B
{
        public static void main(String[] args)
        {
        int i;
        System.out.println(i);
        }
}
```

8. **Program**

```
class C
{
        public static void main(String[] args)
        {
        int i;
        int j;
        j=i;
        System.out.println(j);
        }
}
```

9. **Program**

```
class D
{
        public static void main(String[] args)
        {
        int i,j,k;
        i=j=k=10;
        System.out.println(i);
        System.out.println(j);
        System.out.println(k);
        }
}
```

10. **Program**

```
 class E
{
        public static void main(String[] args)
        {
        int p,q=10,m;
        System.out.println(p);
        System.out.println(q);
        System.out.println(m);
        }
}
```

11. **Program**

```
class F
{
        public static void main(String[] args)
        {
                int i;
                i++;
                System.out.println(i);
        }
}
```

12. **Program**

```
class G
{
        public static void main(String[] args)
        {
                int i;
                System.out.println("done");
        }
}
```

13. **Program**

```
class C  {
        public static void main(String[] args)
        {
                int i = 10;
                System.out.println(i);
                i = 20;
                System.out.println(i);
                i = 50;
                System.out.println(i);
        }
}
```

14. **Program**

```
    class D {
    public static void main(String[] args)
        {
                int i;
                i = 10;
                System.out.println(i);
        }
}
```

15. Program

```
class E  {
        public static void main(String[] args)
        {
                int i;
                System.out.println(i = 10);
                System.out.println(i);
        }
}
```

16. Program

```
class F  {
        public static void main(String[] args)
        {
                int i;
                int j;
                j = i = 0;
                System.out.println(i);
                System.out.println(j);
        }
}
```

17. Program

```
class G {
        public static void main(String[] args)
        {
                int i, j;
                i = 0;
                j = 20;
                System.out.println(i + j);
        }
}
```

18. Program

```
class H {
        public static void main(String[] args)
        {
                int i = 10, j, k = 20;
                j = i + k;
        System.out.println(i);
        System.out.println(j);
        System.out.println(k);
    }
}
```

19. Program

```
class I {
    public static void main(String[] args)
    {
        int k = 0;
        System.out.println(k ++);
        System.out.println(k);
    }
}
```

20. . Program

```
class J {
    public static void main(String[] args)
    {
        int i = 0;
        System.out.println(++ i);
        System.out.println(i);
    }
}
```

21. Program

```
class K {
    public static void main(String[] args)
    {
        int i = 1;
        System.out.println(i ++);
        System.out.println(i);
        i = 1;
        System.out.println(i --);
        System.out.println(i);
        i = 1;
        System.out.println(++ i);
        System.out.println(i);
        i = 1;
        System.out.println(-- i);
        System.out.println(i);
    }
}
```

22. Program

```
class L {
    public static void main(String[] args)
    {
        int i = 0;
        int j = i ++;
        System.out.println(i);
        System.out.println(j);
    }
}
```

23. Program

```
class M {
    public static void main(String[] args)
    {
        int i = 1;
        int j = ++ i;
        System.out.println(i);
        System.out.println(j);
    }
}
```

24. Program

```
class N  {
    public static void main(String[] args)
    {
        int i = 0;
        int j = i ++   +   i;
        System.out.println(i);
        System.out.println(j);
    }
}
```

25. Program

```
class O  {
    public static void main(String[] args)
    {
        int i = 0;
        int j = i++  +  i++  +  i++  + i;
        System.out.println(i);
        System.out.println(j);
    }
}
```

26. Program

```
class P  {
        public static void main(String[] args)
        {
                int i = 0;
                int j = i++  +  ++i  +  i++  +  ++i;
                //     0  +  2  +  2  +  4
                System.out.println(i);
                System.out.println(j);
        }
}
```

27. Program

```
class Q {
        public static void main(String[] args)
        {
                int i = 0;
                i = i++;
                System.out.println(i);
        }
}
```

28. Program

```
class R  {
        public static void main(String[] args)
        {
                int i = 0;
                i = i--;
                System.out.println(i);
        }
}
```

29. Program

```
class S  {
        public static void main(String[] args)
        {
                int i = 0;
                i = ++i;
                System.out.println(i);
        }
}
```

30. Program

```
class T  {
        public static void test(){
                System.out.println("test");
        }
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        }
}
```

31. Program

```
class U {
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        }
        public static void test(){
                System.out.println("test");
        }
}
```

32. Program

```
class V  {
        static void test1(){
        System.out.println("from test1");
        }
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        }
        static void test2(){
        System.out.println("from test2");
        }
}
```

33. Program

```
class W {
        static void test(){
        System.out.println("from test");
        }
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        test();
        }
}
```

34. Program

```
class X  {
        static void test(){
        System.out.println("from test");
        }
        public static void main(String[] args)
        {
        System.out.println("------");
        test();
        test();
        test();
        System.out.println("------");
        }
}
```

35. Program

```
class A  {
        public static void main(String[] args)
        {
        System.out.println("done");
        }
        static void test(){
        int i;
        System.out.println(i = 0);
        }
}
```

36. Program

```
class B  {
        public static void main(String[] args)
        {
        System.out.println("Hello World!");
        test(10);
        }
        static void test(int i) {
        System.out.println(i);
        }
}
```

37. Program

```
class C  {
        public static void main(String[] args)
        {
        int i = 10;
        System.out.println("i value:" + i);
System.out.println(i + " is value of i");
        }
}
```

38. Program

```
class D  {
        public static void main(String[] args)
        {
        int i;
        int j;
        i = 10;
        j = 20;
        System.out.println("i value:" + i + ", j value is:" + j);
        }
}
```

39. Program

```
class E  {
        public static void main(String[] args)
        {
        int i = 10;
        test(10, 20);
        System.out.println("done");
        }
        static void test(int k) {
        System.out.println("from test:" + k);
        }
}
```

40. Program

```
class F  {
        public static void main(String[] args)
        {
        int i = 20;
        test(i);
        System.out.println("done:" + i);
        }
        static void test(int k) {
                k = 10;
        }
}
```

41. Program

```
class G  {
    public static void main(String[] args)
    {
    int i = 10;
    System.out.println("main1:" + i);
    test(i);
    System.out.println("main2:" + i);
    }
    static void test(int i)  {
    i = 20;
    }
}
```

42. Program

```
class H  {
    static int test()   {
    return 10;
    }
    public static void main(String[] args)
    {
    int i = 10;
    System.out.println("1:" + i);
    i = i + test();
    System.out.println("2:" + i);
    System.out.println("3:" + test());
    }
}
```

43. Program

```
class I   {
    public static void main(String[] args)
    {
        int i = 10;
        System.out.println(test(i));
        System.out.println(i);
    }
    static int test(int i) {
        return i++;
    }
}
```

44. Program

```
class J   {
    public static void main(String[] args)
    {
        int i = 0;
        int j = i++  +  test(i)  +  i;
        System.out.println(i);
        System.out.println(j);
    }
    static int test(int i){
        return i++;
    }
}
```

45. Program

```
class K   {
    public static void main(String[] args)
    {
        int i = 0;
        int j = i++  +  test1(i++)  +  i;
        System.out.println(i);
        System.out.println(j);
    }
    static int test1(int i) {
        return ++i;
    }
}
```

46. Program

```
class L {
    public static void main(String[] args)
    {
    int i = 0;
    int j = i++  +  test1(i++)  +   test2(++i);
    System.out.println(i);
    System.out.println(j);
    }
    static int test1(int i) {
        return ++i;
    }
    static int test2(int i) {
        return i++;
    }
}
```

47. Program

```
class A {
    static int i;
    public static void main(String[] args){
        System.out.println(i);
    }
}
```

48. Program

```
class B {
    static int i;
    static void test() {
    System.out.println("from test:" + i);
    }
    public static void main(String[] args)
{
    System.out.println("from main:" + i);
    }
}
```

49. Program

```
class C {
    static int i = 10;
    public static void main(String[] args)
    {
        System.out.println(i);
    }
}
```

50. Program

```
class D  {
    static void test() {
    System.out.println("from test:" + i);
    }
    static int i;
    public static void main(String[] args){
    System.out.println("from main:" + i);
        i = 10;
        test();
    }
}
```

51. Program

```
class E  {
    static int i = 10, j;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
    }
}
```

52. Program

```
class F  {
    static int i, j = 10;
    static int k;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
        System.out.println(k);
        System.out.println(m);
        System.out.println(n);
        System.out.println(p);
    }
    static int m, n = 20, p;
}
```

53. Program

```
class G  {
    static int i;
    public static void main(String[] args)
    {
        System.out.println(i);
        i = 10;
        System.out.println(i);
        int i = 20;
        System.out.println(i);
        i = 30;
        System.out.println(i);
    }
}
```

54. Program

```
class H  {
    static int i;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(H.i);
        i = 10;
        System.out.println(i);
        System.out.println(H.i);
        H.i = 20;
        System.out.println(i);
        System.out.println(H.i);
    }
}
```

55. Program

```
class X  {
    static int i;
    public static void main(String[] args)
    {
        int i = 10;
        System.out.println(i);
        System.out.println(X.i);
        i = 20;
        X.i = 30;
        System.out.println(X.i);
        System.out.println(i);
    }
}
```

56. Program

```
class Y  {
    static int i = 10;
    static double j = 20.7;
}
```

57. Program

```
class A {
    static int i = 10;
    static int j = i;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
    }
}
```

58. Program

```
class B  {
    static int i;
    static int j = i;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
    }
}
```

59. Program

```
class C {
    static int i = 10;
    static int j = 20 + i;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
    }
}
```

60. Program

```
class D {
    static int i = j;
    static int j = 10;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
    }
}
```

61. Program

```
class E {
    static int i = 10;
    static int j = i;
    static int k = m;
    static int m = j;
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
        System.out.println(k);
        System.out.println(m);
    }
}
```

62. Program

```
class F  {
    static void test() {
        System.out.println(i);
    }
    static int i = 10;
    public static void main(String[] args)
    {
        System.out.println(i);
        test();
    }
}
```

63. Program

```
class G {
    static int i = test();
    static int test()   {
        return 10;
    }
    public static void main(String[] args)
    {
        System.out.println(i);
    }
}
```

64. Program

```
class H {
    static int i = test();
    static int j = 10;
    static int test()   {
        return j;
    }
    public static void main(String[] args)
    {
        System.out.println(i);
        System.out.println(j);
    }
}
```

65. Program

```
class I   {
    static int x = 10;
    static int y = test();
    static int z = 20;
    static int test()   {
        //return z;
        return x;
    }
    public static void main(String[] args)
    {
        System.out.println(x);
        System.out.println(y);
        System.out.println(z);
    }
}
```

66. Program

```
class J {
    static int x = test();
    static int test()   {
    System.out.println("A:" + x);
    return 10;
    }
    public static void main(String[] args)
    {
        System.out.println("B:" + x);
    }
}
```

67. Program

```
class L {
    static int i;
    static void test(){
    System.out.println("from test");
    }
}
```

68. Program

```
class M  {
    static int x = test();
    static int test()   {
    System.out.println("from test");
    return 10;
    }
}
```

69. Program

```
class N {
    static    {
        System.out.println("SIB");
    }
}
```

70. Program

```
class A {
    public static void main(String[] args)
    {
    System.out.println("Hello World!");
    }
    Static {
        System.out.println("SIB");
    }
}
```

71. Program

```
class B {
    static {
        System.out.println("SIB1");
    }
    public static void main(String[] args)
    {
    System.out.println("Hello World!");
    }
    Static {
        System.out.println("SIB2");
    }
}
```

72. Program

```
class C {
    static {
        main(null);
    }
    public static void main(String[] args)
    {
    System.out.println("Hello World!");
    }
}
```

73. Program

```
class D {
    static    {
        System.out.println(1);
        main(null);
        System.out.println(3);
    }
    public static void main(String[] args)
    {
    System.out.println("from main");
    }
}
```

74. Program

```
class E {
    static int i = test();
    static int test()   {
    main(null);
    return 20;
    }
    public static void main(String[] args)
    {
    System.out.println("main:" + i);
    }
}
```

75. Program

```
class F {
    static int i = test1();
    static int test1() {
        System.out.println(1);//1
        test2();//2
        System.out.println(2);//8
        return 10;//9
    }
    static void test2() {
        System.out.println(3);//3
        main(null);//4
        System.out.println(4);//7
    }
    static    {
    System.out.println(5 + ":" + i);//10
    i = 20;//11
    main(null);//12
    System.out.println(6 + ":" + i);//15
    }
    public static void main(String[] args)
    {
    System.out.println(7 + ":" + i);//5, 13,
16
```

```
        i = 30;//6, 14, 17
        }
}
```

76. Program

```
class G {
        static int i = 10;
        static    {
                System.out.println(i);
        }
        public static void main(String[] args)
        {
                System.out.println(i);
        }
}
```

77. Program

```
class H {
        static int i = 20;
        static    {
                i = 10;
        }
        public static void main(String[] args)
        {
                System.out.println(i);
        }
}
```

78. Program

```
class I {
        static    {
                x = 10;
        }
        static int x = 20;
        public static void main(String[] args)
        {
                System.out.println(x);
        }
}
```

79. Program

```
class A  {
        static int i;
        static void test() {
        System.out.println("from test:" + i);
        System.out.println("from test:" + A.i);
        }
}
class B  {
        public static void main(String[] args)
        {
        System.out.println("from main:" + A.i);
        }
}
```

80. Program

```
class C  {
        static int i;
        static void test() {
        System.out.println("from test:" + i);
        }
}
class D  {
        static int i = 10;
        public static void main(String[] args)
        {
        int i = 20;
        System.out.println("from main:" + i);
        System.out.println("from main:" + D.i);
        System.out.println("from main:" + C.i);
        C.test();
        }
}
```

81. Program

```
class E  {
        public static void main(String[] args)
        {
        System.out.println("from E.main");
        }
        static    {
        System.out.println("from E.SIB");
        }
}
class F  {
        public static void main(String[] args)
        {
        System.out.println("from F.main");
        }
        static  {
        System.out.println("from F.SIB");
        }
}
```

82. Program

```
class G  {
        static  {
                System.out.println("G-SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("G-main");
        }
}
class H  {
        static  {
                System.out.println("H-SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("H-main-begin");
        G.main(args);
        System.out.println("H-main-end");
        }
}
```

83. Program

```
class I {
        static void test() {
        System.out.println("from test");
        }
        static  {
                System.out.println("I-SIB");
        }
}
class J  {
        public static void main(String[] args)
        {
        System.out.println("---111---");
        I.test();
        System.out.println("---222---");
        I.test();
        System.out.println("---333---");
        I.test();
        System.out.println("---444---");
        }
        static  {
                System.out.println("J-SIB");
        }
}
```

84. Program

```
class K {
        static int i = 10;
        static  {
                System.out.println("K-SIB");
        }
        static void test(){
                System.out.println("K-test");
        }
}
class L {
        static  {
                System.out.println("L-SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("---aaa---");
        K.test();
        System.out.println("---bbb---");
        K.test();
        System.out.println("---ccc---");
        System.out.println(K.i);
        System.out.println("---ddd---");
        }
}
```

85. Program

```
class B {
        static int i;
        static int j = 10;
        static  {
                System.out.println("B-SIB");
```

```
	}
	static void test() {
	System.out.println("from test");
	}
	public static void main(String[] args)
	{
	System.out.println("main begin");
	test();
	System.out.println("main end");
	}
}
```

86. Program

```
class C {
	static int i;
	static	{
		i = 10;
		System.out.println("C-SIB");
	}
	static void test() {
		System.out.println("C-test");
	}
}
class D {
	static	{
		System.out.println("D-SIB");
	}
	public static void main(String[] args)
	{
		System.out.println(1);
		C.test();
		System.out.println(2);
		System.out.println(C.i);
	}
}
```

87. Program

```
class A {
	int i;
	public static void main(String[] args)
	{
		A a1 = new A();
		System.out.println(a1.i);
	}
}
```

88. Program

```
class B {
	void test() {
	System.out.println("from test");
	}
	public static void main(String[] args)
	{
	B b1 = new B();
	b1.test();
	System.out.println("Hello World!");
	}
}
```

89. Program

```
class C {
	int i;
	static void test() {
		C c1 = new C();
		System.out.println(c1.i);
	}
}
```

90. Program

```
class D {
	void test1()  {
	}
	static void test2() {
		D rv = new D();
		rv.test1();
	}
}
```

91. Program

Given:

```
class Foo {
static void alpha() { /* more code here */ }
void beta() { /* more code here */ }
}
```

Which two are true? (Choose two.)

A. Foo.beta () is a valid invocation of beta ().

B. Foo.alpha () is a valid invocation of alpha ().

C. Method beta () can directly call method alpha ().

D. Method alpha () can directly call method beta ().

Answer: BC

92. Program

```
class E {
        int i;
        static  {
                E e1 = new E();
                System.out.println(e1.i);
        }
}
```

93. Program

```
class F {
        int i;
        void test1()  {
        }
        public static void main(String[] args)
        {
                F f1 = new F();
                f1.i = 10;
                f1.test1();
        }
}
```

94. Program

```
class G {
        void test1()  {
        }
        static    {
                G obj = new G();
                obj.test1();
        }
}
```

95. Program

```
class H {
        int i;
        public static void main(String[] args)
        {
                H obj = new H();
                System.out.println(obj.i);
                obj.i = 10;
                System.out.println(obj.i);
        }
}
```

96. Program

```
class I {
        int x;
        int y = 10;
        public static void main(String[] args)
        {
                I obj = new I();
                System.out.println(obj.x);
                System.out.println(obj.y);
                obj.x = 20;
                obj.y = 40;
                System.out.println(obj.x);
                System.out.println(obj.y);
        }
}
```

97. Program

Given:

```
public class Pass {
public static void main(String [1 args) {
int x= 5;
Pass p = new Pass();
p.doStuff(x);
System.out.print(” main x = “+ x);
}

void doStuff(int x) {
        22. System.out.print(" doStuff x = "+
        x++);
        23. }
        24. }
```

What is the result?

A. Compilation fails.

B. An exception is thrown at runtime.

C. doStuffx = 6 main x = 6

D. doStuffx = 5 main x = 5

E. doStuffx = 5 main x = 6

F. doStuffx = 6 main x = 5

Answer: D

98. Program

```
class J {
        int x, y = 10, z;
        public static void main(String[] args)
        {
                J obj1 = new J();
                System.out.println(obj1.x);
                System.out.println(obj1.y);
                System.out.println(obj1.z);
        }
}
```

99. Program

```
class K {
        int i;
        public static void main(String[] args)
        {
                K k1 = new K();
                k1.i = 10;
                K k2 = new K();
                k2.i = 20;
                System.out.println(k1.i);
                System.out.println(k2.i);
        }
}
```

100. Program

```
class A {
        int i;
        public static void main(String[] args)
        {
                A a1 = new A();
                System.out.println(a1.i);
A a2 = new A();
                System.out.println(a2.i);
A a3 = new A();
                System.out.println(a3.i);
                a1.i = 10;
                a2.i = 20;
                a3.i = 30;
                System.out.println(a1.i);
                System.out.println(a2.i);
                System.out.println(a3.i);
        }
}
```

101. Program

```
class B {
        int i;
        double d;
        boolean b;
        public static void main(String[] args)
        {
                B b1 = new B();
                System.out.println(b1.i);
                System.out.println(b1.d);
                System.out.println(b1.b);
        }
}
```

102. Program

```
class C {
        int i;
        int j = 10;
        C()  {
                i = 20;
        }
        public static void main(String[] args)
        {
                C c1 = new C();
                System.out.println(c1.i);
                System.out.println(c1.j);
        }
}
```

103. Program

```
class D {
        int i, j;
        D() {
                i = 20;
                j = 40;
        }
        public static void main(String[] args)
        {
        D d1 = new D();
        System.out.println("i=" + d1.i + ",j=" +
d1.j);
        }
}
```

104. Program

```
class E {
        E() {
                System.out.println("E()");
        }
        public static void main(String[] args)
        {
                E e1 = new E();
                System.out.println("----");
                E e2 = new E();
        }
}
```

105. Program

```
class F {
        public static void main(String[] args)
        {
                F f1 = new F();
                System.out.println("----");
                F f2 = new F();
        }
}
```

106. Program

```
class G {
        G() {
                System.out.println("G()");
        }
        G(int i) {
                System.out.println("G(int)");
        }
        public static void main(String[] args)
        {
                G g1 = new G();
                System.out.println("----");
                G g2 = new G(20);
                System.out.println("----");
                G g3 = new G();
                System.out.println("----");
        }
}
```

107. Program

```
class H {
        H(int i) {
                System.out.println("H(int)");
        }
        public static void main(String[] args)
        {
                H h1 = new H(20);
                System.out.println("----");
                H h2 = new H(10);
                System.out.println("----");
                //H h3 = new H();
                System.out.println("----");
        }
}
```

108.

Program

```
class I {
        public static void main(String[] args)
        {
                I i1 = new I();
                System.out.println("------");
                I i2 = new I();
                System.out.println("------");
        }
}
```

109. Program

```
class J {
        J(int x)  {
        System.out.println("X(int x)");
        }
        J(byte y) {
        System.out.println("X(int y)");
        }
        public static void main(String[] args)
        {
                J obj = new J(90);
                System.out.println("done");
        }
}
```

110. **Program**

```
class K {
        K(int i, int j)  {
        System.out.println("int, int");
        }
        K(double i, int j) {
        System.out.println("double, int");
        }
        public static void main(String[] args)
        {
                K obj1 = new K(10, 20);
                System.out.println("-----");
                K obj2 = new K(10.9, 20);
                System.out.println("-----");
        }
}
```

111. Program

```
class L {
        L()  {
        System.out.println("L()");
        }
        L(int i)  {
        System.out.println("L(int)");
        }
        L(int i, int j)  {
        System.out.println("L(int, int)");
        }
        public static void main(String[] args)
        {
                L obj1 = new L();
                System.out.println("-----");
                L obj2 = new L(10);
                System.out.println("-----");
                L obj3 = new L(10, 20);
                System.out.println("-----");
        }
}
```

112. Program

```
class M {
        M() {
                System.out.println("M()");
        }
        M(int i) {
                this();
                System.out.println("M(int)");
        }
        public static void main(String[] args)
        {
                M m1 = new M();
                System.out.println("-----");
                M m2 = new M(10);
                System.out.println("-----");
        }
}
```

113. Program

```
class N {
        N()  {
                this(10);
                System.out.println("N()");
        }
        N(int i)  {
                System.out.println("N(int)");
        }
        public static void main(String[] args)
        {
                N n1 = new N();
                System.out.println("-----");
                N n2 = new N(20);
                System.out.println("-----");
        }
}
```

114. Program

```
class O  {
        O() {
                this(2, 5);
                System.out.println("O()");
        }
        O(int i)  {
                this();
                System.out.println("O(int)");
        }
O(int i, int j) {
        System.out.println("O(int, int)");
        }
        public static void main(String[] args)
        {
                O o1 = new O();
                System.out.println("-----");
                O o2 = new O(10);
                System.out.println("-----");
                O o3 = new O(10, 30);
                System.out.println("-----");
        }
}
```

115. Program

```
class P {
        P() {
                System.out.println("P()");
        }
        P(int i)  {
                this();
                System.out.println("P(int)");
        }
        P(char c1) {
        this();
        System.out.println("P(char)");
        }
        P(boolean b1)   {
        this('a');
        System.out.println("P(boolean)");
        }
        P(double d)  {
        this(10);
        System.out.println("P(double)");
        }
        public static void main(String[] args)
        {
                P p1 = new P();
                System.out.println("-----");
                P p2 = new P('a');
                System.out.println("-----");
                P p3 = new P(10);
                System.out.println("-----");
                P p4 = new P(10.9);
                System.out.println("-----");
                P p5 = new P(false);
                System.out.println("-----");
        }
}
```

116. Program

```
class Q {
        Q(int i)  {
                this();
                System.out.println("Q()");
        }
        Q()  {
                this(20);
                System.out.println("Q(int)");
        }
}
```

117. Program

```
class R {
        R()  {
                this(10);
                System.out.println("R()");
        }
        R(int i)  {
                System.out.println("R(int)");
        }
}
```

118. Program

```
class S {
        S()  {
                System.out.println("S()");
        }
{
                System.out.println("IIB");
        }

        public static void main(String[] args)
        {
                S s1 = new S();
                System.out.println("------");
                S s2 = new S();
                System.out.println("------");
        }
}
```

119. Program

```
class T {
        T()  {
                System.out.println("T()");
        }
{
                System.out.println("IIB");
        }
T(int i) {
                System.out.println("T(int)");
        }
public static void main(String[] args)       {
                T t1 = new T();
                System.out.println("------");
                T t2 = new T(20);
                System.out.println("------");
        }
}
```

120. Program

```
class A {
        A()  {
                System.out.println("A()");
        }

        {
                System.out.println("IIB1");
        }
{
                System.out.println("IIB2");
        }
        public static void main(String[] args)
        {
                A a1 = new A();
                System.out.println("-----");
                A a2 = new A();
                System.out.println("-----");
        }
}
```

121. Program

```
class B {
        {
                System.out.println("IIB1");
        }

        B() {
                System.out.println("B()");
        }

        {
                System.out.println("IIB2");
        }

        B(int i)  {
                System.out.println("B(int)");
        }
        public static void main(String[] args)
        {
                B b1 = new B();
                System.out.println("------");
                B b2 = new B(90);
        }
}
```

122. Program

```
class C {
        {
                System.out.println("IIB1");
        }
        public static void main(String[] args)
        {
                C c1 = new C();
                System.out.println("-----");
                C c2 = new C();
                System.out.println("-----");
        }
}
```

123. Program

```
class D {
        static
        {
                System.out.println("SIB");
        }

        {
                System.out.println("IIB");
        }

        public static void main(String[] args)
        {
                D d1 = new D();
                System.out.println("-----");
                D d2 = new D();
                System.out.println("-----");
        }
}
```

124. Program

```
class E {
        static    {
                System.out.println("SIB1");
        }

        {
                System.out.println("IIB1");
        }

        static    {
                System.out.println("SIB2");
        }
        E()       {
                System.out.println("E()");
        }
        public static void main(String[] args)
        {
        System.out.println("main begin");
        E e1 = new E();
        System.out.println("-------");
        E e2 = new E();
        System.out.println("main end");
        }

        {
                System.out.println("IIB2");
        }
}
```

125. Program

```
class F {
        F()  {
                System.out.println("F()");
        }
```

```
	{
		System.out.println("IIB");
	}

	F(int i)  {
		this();
		System.out.println("F(int)");
	}
	public static void main(String[] args)
	{
		F f1 = new F();
		System.out.println("-----");
		F f2 = new F(20);
		System.out.println("-----");
	}
}
```

126. Program

Given:

```
class One {
public One() {     System.out.print(1); }
}
class Two extends One {
public Two() { System.out.print(2); }
}
class Three extends Two {
public Three() {   System.out.print(3); }
}
public class Numbers{
public static void main( String[] args) {
new Three(); }
}
```

What is the result when this code is executed?

A. 1

B. 3

C. 123

D. 321

E. The code rims with no output.

Answer: C

127. Program

```
class G {
	{
		System.out.println("G-IIB");
	}

	G(int i)  {
		System.out.println("G(int)");
	}

	{
		System.out.println("G-IIB2");
	}

	G()  {
		this(10);
		System.out.println("G()");
	}
	public static void main(String[] args)
	{
		G g1 = new G();
		System.out.println("----");
		G g2 = new G(20);
		System.out.println("----");
	}
}
```

128. Program

Given:

```
public class Hello {
String title;
int value;
public Hello() {
title += " World";
}
public Hello(int value) {
this.value = value;
title = "Hello";
Hello();
}
}
```

and:

```
Hello c = new Hello(5);
System.out.println(c.title);
```

What is the result?

A. Hello

B. Hello World
C. Compilation fails.
D. Hello World 5
E. The code runs with no output.
F. An exception is thrown at runtime.

Answer: C

129. Program

```
class I {
        {
                System.out.println("IIB1");
        }

        static  {
                System.out.println("SIB1");
        }

        I()  {
                System.out.println("I()");
        }

        {
                System.out.println("IIB2");
        }

        static  {
                System.out.println("SIB2");
        }
        I(double d)  {
        this();
        System.out.println("I(double)");
        }
        public static void main(String[] args)
        {
        System.out.println("main-begin");
        I obj1 = new I();
        System.out.println("-----");
        I obj2 = new I(90.8);
        System.out.println("main-end");
        }
}
```

130. . Program

```
class J  {
        {
                System.out.println("J-IIB1");
        }

        J()  {
                System.out.println("J()");
        }

        {
                System.out.println("J-IIB2");
        }
}
class K  {
        {
                System.out.println("K-IIB1");
        }

        K() {
                System.out.println("K()");
        }

        {
                System.out.println("K-IIB2");
        }

        K(int i)  {
                System.out.println("K(int)");
        }

        static  {
                System.out.println("K-SIB");
        }

        public static void main(String[] args)
        {
        System.out.println("main begin");
        K k1 = new K();
        System.out.println("-----");
```

```
		J j1 = new J();
		System.out.println("-----");
		K k2 = new K(20);
		System.out.println("main end");
		}
}
```

131. Program

```
class L {
		static	{
			System.out.println("L-SIB");
		}
		L()	{
			System.out.println("L()");
		}

		{
			System.out.println("L-IIB");
		}
}
class M {
		static	{
			System.out.println("M-SIB");
		}
		M()	{
			System.out.println("M()");
		}

		{
			System.out.println("M-IIB");
		}

		M(int i) {
			L obj = new L();
			System.out.println("M(int)");
		}
		public static void main(String[] args)
		{
		System.out.println("main begin");
		M m1 = new M();
		System.out.println("-----");
		M m2 = new M(90);
		System.out.println("-----");
		System.out.println("main end");
		}
}
```

132.

Program

```
class A {
		int i;
}
class B extends A {
		int j;
		public static void main(String[] args)
		{
			B b1 = new B();
			System.out.println(b1.i);
			System.out.println(b1.j);
		}
}
```

133. Program

```
class C {
		int i;
		void test()  {
			System.out.println("test");
		}
}
class D extends C  {
		int j;
}
class E  {
		public static void main(String[] args)
		{
			D d1 = new D();
			d1.i = 10;
			d1.j = 20;
			d1.test();
			System.out.println(d1.i);
			System.out.println(d1.j);
		}
}
```

134. Program

```
class F  {
    int i;
    static int j;
    void test1()  {
        System.out.println("test1");
    }
    static void test2() {
        System.out.println("test2");
    }
}
class G extends F   {
    int k;
    static void test3() {
        System.out.println("test3");
    }
    public static void main(String[] args)
    {
        F.j = 10;
        F.test2();
        G.test2();
        G.test3();
        System.out.println(G.j);
        F f1 = new F();
        f1.i = 20;
        f1.test1();
        G g1 = new G();
        g1.i = 30;
        g1.k = 50;
        g1.test1();
    }
}
```

135. Program

```
class H {
    int x;
}
class I extends H  {
    int y;
}
class J extends I  {
    int z;
}
class Manager  {
    public static void main(String[] args)
    {
        H h1 = new H();
        I i1 = new I();
        J j1 = new J();
        System.out.println(h1.x);
        System.out.println(i1.x);
        System.out.println(j1.x);
        System.out.println(i1.y);
        System.out.println(j1.y);
        System.out.println(j1.z);
    }
}
```

136. Program

```
class K  {
    K()  {
        System.out.println("K()");
    }
}
class L extends K  {
    L()  {
        System.out.println("L()");
    }
}
class Manager1  {
    public static void main(String[] args)
    {
        K k1 = new K();
        System.out.println("-----");
        L l1 = new L();
        System.out.println("-----");
    }
}
```

137. Program

```
class M extends Object  {
        M()  {
                super();
                System.out.println("M()");
        }
}
class N extends M  {
        N() {
                super();
                System.out.println("N()");
        }
}
class O extends N  {
        O()  {
                super();
                System.out.println("O()");
        }
}
class Manager2  {
        public static void main(String[] args)
        {
        M m1 = new M();
        System.out.println("-----");
        N n1 = new N();
        System.out.println("-----");
    o1 = new O();
        System.out.println("-----");
        }
}
```

138. Program

Click the Exhibit button.

```
1. public class Employee {
2. String name;
3. double baseSalary;
4. Employee(String name, double baseSalary) {
5. this.name = name;
6. this.baseSalary = baseSalary;
7. }
8. }
And:
1. public class Salesperson extends Employee {
2. double commission;
3. public Salesperson(String name, double
baseSalary,
4. double commission) {
5. // insert code here
6. }
7. }
```

Which code, inserted at line 7, completes the Salesperson constructor?

A. this.commission = commission;

B. superb();
commission = commission;

C. this.commission = commission;
superb();

D. super(name, baseSalary);
this.commission = commission;

E. super();
this.commission = commission;

F. this.commission = commission;
super(name, baseSalary);

Answer: D

139. Program

```
11.class Mammal{
12.
13. class Raccoon extends Mammal{
14. Mammal m=new Mammal();
15.}
16.
17. class BabyRaccoon extends Mammal{}
```

Which four statements are true?(choose four)

A. Raccoon is-a Mammal
B. Raccoon has-a Mammal
C. BabyRaccoon is-a Mammal
D. BabyRaccoon is-a Raccoon
E. BabyRaccoon has-a Mammal
F. BabyRaccoon is-a BayRaccoon

Ans : A, B ,C, F

140. Program

```
class P {
        P()  {
                System.out.println("P()");
        }
}
class Q extends P  {
        Q()  {
```

```
        System.out.println("Q()");
    }
}
class R extends Q  {
    R()  {
        System.out.println("R()");
    }
}
class S extends R  {
    S()  {
        System.out.println("S()");
    }
}
class T extends S {
    T()  {
        System.out.println("T()");
    }
    public static void main(String[] args)
    {
        P p1 = new P();
        System.out.println("------");
        Q q1 = new Q();
        System.out.println("------");
        R r1 = new R();
        System.out.println("------");
        S s1 = new S();
        System.out.println("------");
        T t1 = new T();
    }
}
```

141. Program

```
class A {
    A()  {
        System.out.println("A()");
    }
    A(int i)  {
        System.out.println("A(int)");
    }
}
class B extends A  {
    B() {
        System.out.println("B()");
    }
    B(int i)  {
        System.out.println("B(int)");
    }
}
class Manager  {
    public static void main(String[] args)
    {
        A a1 = new A();
        System.out.println("-----");
        A a2 = new A(20);
        System.out.println("-----");
        B b1 = new B();
        System.out.println("-----");
        B b2 = new B(30);
        System.out.println("-----");
    }
}
```

142. Program

```
class C  {
    C(int i)  {
        System.out.println("C(int)");
    }
    C()  {
        System.out.println("C()");
    }
}
class D extends C  {
    D(int i)  {
        System.out.println("D(int)");
    }
}
class Manager1 {
    public static void main(String[] args)
    {
        C c1 = new C(90);
        System.out.println("-----");
        D d1 = new D(80);
        }
}
```

143. Program

Click the Exhibit button.

```
11. class Person {
12. String name = “No name’;
13. public Person(String nm) { name = nm;
}
14. }
15.
16. class Employee extends Person {
17. String emplD = “0000”;
18. public Employee(String id) { empID =
id; }
19. }
20.
21. public class EmployeeTest {
22. public static void main(String[] args) {
23. Employee e = new Employee(”4321”);
24. System.out.println(e.empID);
25. }
26. }
```

What is the result?

A. 4321

B. 0000

C. An exception is thrown at runtime.

D. Compilation fails because of an error in line 18.

Answer: D

144. Program

```
class E  {
        E()  {
                System.out.println("E()");
        }
}
class F extends E  {
        F(int i)  {
                System.out.println("F(int)");
        }
}
class Manager2   {
        public static void main(String[] args)
        {
                E e1 = new E();
                System.out.println("------");
                F f1 = new F(90);
        }
}
```

145. Program

```
class G {
        G(int i)  {
                System.out.println("G(int)");
        }
}
class H extends G  {
        H()  {
                super(10);
                System.out.println("H()");
        }
}
class Manager3  {
        public static void main(String[] args)
        {
                G g1 = new G(90);
                System.out.println("-----");
                H h1 = new H();
        }
}
```

146. Program

```
1. class X{
2.X(){System.out.println(1);}
3. X(int x){
4. this();Sysyem.out.println(2);
5.}
6.}
7. public class Y extends X{
8.Y(){super(6);System.out.println(3)  ;}
9. Y(in x){
10. this();System.out.println(4);
11.}
12. public static void main(String args[])
        {
                new Y(5);
        }
13.}
```

What is the result?

A. 13
B. 134
C. 1234
D. 2134
E. 2143
F. 4321

Ans :C

147. Program

```
class I {
        I(double j) {
        System.out.println("I(double)");
        }
}
class J extends I {
        J() {
                super(9.9);
                System.out.println("J()");
        }
        J(double d) {
        super(d);
        System.out.println("J(double)");
        }
}
class Manager4 {
        public static void main(String[] args)
        {
                I i1 = new I(9.9);
                System.out.println("-----");
                J j1 = new J();
                System.out.println("-----");
                J j2 = new J(3.4);
        }
}
```

148. Program

Given:

```
public class Plant {
private String name;
public Plant(String name) { this.name = name; }
public String getName() { return name; }
}
public class Tree extends Plant {
public void growFruit() { }
public void dropLeaves() { }
}
```

Which is true?

A. The code will compile without changes.

B. The code will compile if public Tree() { Plant(); } is added to the Tree class.

C. The code will compile if public Plant() { Tree(); } is added to the Plant class.

D. The code will compile if public Plant() { this(”fern”); } is added to the Plant class.

E. The code will compile if public Plant() { Plant(”fern”); } is added to the Plant class.

Answer: D

149. Program

```
class K {
        K() {
                System.out.println("K()");
        }

        {
                System.out.println("K-IIB");
        }
}
class L extends K {
        L() {
                System.out.println("L()");
        }

        {
                System.out.println("L-IIB");
        }
}
class Manager5 {
        public static void main(String[] args)
        {
                K k1 = new K();
                System.out.println("-----");
                L l1 = new L();
                System.out.println("-----");
        }
}
```

150. Program

```
class M {
        M() {
                System.out.println("M()");
        }

        {
                System.out.println("M-IIB1");
        }

        {
                System.out.println("M-IIB2");
        }
}
class N extends M  {
        {
                System.out.println("N-IIB1");
        }

        N() {
                System.out.println("N()");
        }

        {
                System.out.println("N-IIB2");
        }
}
class Manager6 {
        public static void main(String[] args)
        {
                M m1 = new M();
                System.out.println("------");
                N n1 = new N();
        }
}
```

151. Program

Given:

```
1. class ClassA {
2. public int numberOfinstances;
3. protected ClassA(int numberOfinstances) {
4. this.numberOflnstances =
numberOfinstances;
5. }
6. }
7. public class ExtendedA extends ClassA {
8. private ExtendedA(int numberOfinstances)
        {
9.       super(numberOflnstances);
10.      }
11. public static void main(String[] args) {
12. ExtendedA ext = new ExtendedA(420);
13. System.out.print(ext.numberOflnstances);
14. }
15. }
```

Which is true?

A. 420 is the output.

B. An exception is thrown at runtime.

C. All constructors must be declared public.

D. Constructors CANNOT use the private modifier.

E. Constructors CANNOT use the protected modifier.

Answer: A

152. Program

```
class O {
        {
                System.out.println("O-IIB");
        }

        O() {
                System.out.println("O()");
        }
}
class P extends O  {
        P() {
                System.out.println("P()");
        }

        {
```

```
		System.out.println("P-IIB");
	}

	P(int i)  {
		this();
		System.out.println("P(int)");
	}
}
class Manager7 {
	public static void main(String[] args)
	{
		O o1 = new O();
		System.out.println("------");
		P p1 = new P();
		System.out.println("------");
		P p2 = new P(20);
		System.out.println("------");
	}
}
```

153. Program

Given

```
5. class Atom{
6.Atom(){System.out.println("atom)   }
7.}
8. class Rock extends Atom{
9.Rock(String type)
{System.out.println(type);}
10.}
11.public class Mountain extends
Rock{
12.Mountain(){
13. super("granite");
14. new Rock("granite");
15.}
16. public static void main (String
args[])
	{
	new Mountain();
	}
17.}
```

What is the result?

A. Compilation fails
B. atom granite
C. granite granite
D. atom granite granite
E. An exception is thrown at runtime
F. Atom granite atom granite

Ans : F

154. Program

```
class A {
	static	{
		System.out.println("A-SIB");
	}
	A()  {
		System.out.println("A()");
	}

	{
		System.out.println("A-IIB");
	}
}
class B extends A  {
	static	{
		System.out.println("B-SIB");
	}

	{
		System.out.println("B-IIB");
	}

	B()  {
		System.out.println("B()");
	}
}
class C extends B  {
	C()  {
		System.out.println("C()");
	}

	{
		System.out.println("C-IIB");
	}

	static	{
```

```
                System.out.println("C-SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("main begin");
        A a1 = new A();
        System.out.println("------");
        B b1 = new B();
        System.out.println("------");
        C c1 = new C();
        System.out.println("main end");
        }
}
```

155. Program

```
class D  {
        D()  {
                System.out.println("D()");
        }
        static    {
                System.out.println("D-SIB");
        }

        {
                System.out.println("D-IIB");
        }
}
class E extends D  {
        E()  {
                System.out.println("E()");
        }

        {
                System.out.println("E-IIB");
        }

        static    {
                System.out.println("E-SIB");
        }
}
class Manager  {
        public static void main(String[] args)
        {
        System.out.println("main begin");
        D d1 = new D();
        System.out.println("-----");
        E e1 = new E();
        System.out.println("main end");
        }
        static    {
        System.out.println("SIB-Manager");
        }
}
class Manager1   {
        public static void main(String[] args)
        {
        System.out.println("main begin");
        E e1 = new E();
        System.out.println("-----");
        D d1 = new D();
        System.out.println("main end");
        }
        static    {
        System.out.println("SIB-Manager1");
        }
}
class Manager2 extends D {
        public static void main(String[] args)
        {
        System.out.println("main begin");
        E e1 = new E();
        System.out.println("-----");
        D d1 = new D();
        System.out.println("main end");
        }
        static    {
        System.out.println("SIB-Manager2");
        }
}
class Manager3 extends E {
        public static void main(String[] args)
        {
```

```
		System.out.println("main begin");
		E e1 = new E();
		System.out.println("-----");
		D d1 = new D();
		System.out.println("main end");
		}
		static	{
		System.out.println("SIB-Manager3");
		}
}
```

**156.	Program**

```
class Manager4 {
		static	{
		System.out.println("Manager4-SIB");
		}
		public static void main(String[] args)
		{
		System.out.println("Manager4-main");
		}
}
class Manager5 extends  Manager4  {
		static	{
		System.out.println("Manager5-SIB");
		}
}
class Manager6  {
		static	{
		System.out.println("Manager6-SIB");
		}
		public static void main(String[] args)
		{
		System.out.println("Manager6-main-
b");
		Manager4.main(args);
		System.out.println("Manager6-main-
e");
		}
}
class Manager7  {
		static	{
		System.out.println("Manager7-SIB");
		}
		public static void main(String[] args)
		{
		System.out.println("Manager7-main-
b");
		Manager5.main(args);
		System.out.println("-----");
		Manager4.main(args);
		System.out.println("Manager7-main-
e");
		}
}
```

**157.	Program**

```
class F {
		static	{
			System.out.println("F-SIB");
		}
		public static void main(String[] args)
		{
		System.out.println("F-main");
		}
}
class G extends F {
		static	{
		System.out.println("G-SIB");
		}
		public static void main(String[] args)
		{
		System.out.println("G-main");
		}
}
class H extends G  {
		static	{
			System.out.println("H-SIB");
		}
}
class I  {
		static	{
			System.out.println("I-SIB");
```

```
	}
	public static void main(String[] args)
	{
		System.out.println("I-main-begin");
		G.main(args);
		System.out.println("I-main-end");
	}
}
class J {
	static	{
		System.out.println("J-SIB");
	}
	public static void main(String[] args)
	{
	System.out.println("J-main-begin");
	H.main(args);//G.main(args);
	System.out.println("J-main-end");
	}
}
```

158. Program

```
class K {
	static	{
		System.out.println("K-SIB");
	}
	static void test() {
		System.out.println("K-test");
	}
}
class L extends K {
	static	{
		System.out.println("L-SIB");
	}
}
class M {
	public static void main(String[] args)
	{
		System.out.println("--11---");
		L.test();//K.test();
		System.out.println("--22---");
	}
}
```

159. Program

```
class A {
	public static void main(String[] args)
	{
	System.out.println("from first A");
	}
}
class A {
	public static void main(String[] args)
	{
	System.out.println("from second A");
	}
}
```

160. Program

```
class B
{
	static int i = 10;
}

-----------------------------
// Other .java file
class B {
	static int i = 20;
}
class C {
	public static void main(String[] args)
	{
	System.out.println("main:" + B.i);
	}
}
```

161. Program

```
package pack1;
public class D
{
	public static int i = 10;
}
--- - - - - - - - -  - - - - - - - - - - -
// Other . file

package pack2;
class D {
        static int i = 20;
}
class E {
        public static void main(String[] args)
        {
        System.out.println(D.i);
        System.out.println(pack1.D.i);
        }
}
```

162. Program

```
package pack1;
public class A
{
        public static void test1()
        {
        System.out.println("from test1");
        }
}
- - - - - - - - - - - - - - - - - - - - - - - - - - -
package pack1;
class Manager {
        public static void main(String[] args)
        {
        System.out.println("main begin");
        A.test1();
        System.out.println("main end");
        }
}
```

163. Program

```
package com;
import com.lara.B;
import com.rst.C;
public class A {
        public static void main(String[] args)
        {
                B.test2();
                C.test3();
        }
        public static void test1() {
                System.out.println("test1");
        }
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
package com.lara;
import com.A;
import com.rst.C;
public class B {
        public static void main(String[] args)
        {
                A.test1();
                C.test3();
        }
        public static void test2() {
                System.out.println("test2");
        }
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - - -
package com.rst;
import com.A;
import com.lara.B;
public class C {
        public static void main(String[] args)
        {
                A.test1();
                B.test2();
        }
        public static void test3() {
                System.out.println("test3");
        }
}
```

164. Program

```
package pack1;
class A  {
        private static int i;
        private static void test1() {
        System.out.println("from test1:" + i);
        }
        static void test2() {
        System.out.println("from test2:" + i);
        }
        public static void main(String[] args)
        {
        System.out.println("from A.main:" + i);
        test1();
        test2();
        }
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
package  pack1;
class B  {
        public static void main(String[] args)
        {
                //A.i = 20;
                //A.test1();
                A.test2();
        }
}
```

165. Program

```
package pack1;
class C  {
        private int i;
        int j;
}
class D extends C {
        void test()  {
                //i = 10;
                j = 20;
        }
}
```

166. Program

```
package pack1;
class E {
        private E()  {
        }
        public static void main(String[] args)
        {
                E e1 = new E();
        }
}
class F  {
        public static void main(String[] args)
        {
                E e1 = new E();
        }
}
```

167. Program

Click the Exhibit button.

```
1. public class A {
2.
3. private int counter = 0;
4.
5. public static int getInstanceCount() {
6. return counter;
7. }
8.
9. public A() {
10. counter++;
11. }
12.
13. }
```

Given this code from Class B:

```
25.A a1 =new A();
26. A a2 =new A();
27. A a3 =new A();
28. System.out.printIn(A.getInstanceCount() );
```

What is the result?

A. Compilation of class A fails.

B. Line 28 prints the value 3 to System.out.

C. Line 28 prints the value 1 to System.out.

D. A runtime error occurs when line 25 executes.

E. Compilation fails because of an error on line 28.

Answer: A

168. Program

```
class G {
        private G()  {

        }
        G(int i)  {

        }
}
class H extends G {
        H()  {
                super(10);
        }
}
```

169. Program

```
package pack1;
class I {
        private static I obj = null;
        private I() {
        System.out.println("obj created");
        }
        static    {
                obj = new I();
        }
        public static I getObject()          {
                return obj;
        }
}
class Manager  {
        public static void main(String[] args)
        {
                //I i1 = new I();
                I i2 = I.getObject();
                I i3 = I.getObject();
                I i4 = I.getObject();
                System.out.println("done");
        }
}
```

170. Program

```
package pack1;
class J {
        int x;
}
- - - -- - - - - - - - - - - - - - - - - - - - - - - -  - -
package pack1;
class K {
        public static void main(String[] args)
        {
                J obj = new J();
                System.out.println(obj.x);
        }
}
```

171. Program

Given:

1. package test;
2.
3. class Target {
4. public String name = “hello”;
5. }

What can directly access and change the value of the variable name?

A. any class
B. only the Target class
C. any class in the test package
D. any class that extends Target

Answer: C

172. Program

```
package pack1;
public class L {
        int x;
}
- - - - - - - - - - - - - - -  - - - - - - -  - - - - -
package pack2;
class M {
        public static void main(String[] args)
        {
                L obj = new L();
                System.out.println(obj.x);
        }
}
```

173. Program

Given the following Directory structure big project

```
├── source
│   ├── Utils.java
├── classes
```

And the following command line invocation

Javac –d classes source/Utils.java

Assume that current directory is bigProject, what is the result?

A. If the compile is successfully, Utils.class is added to the source directory.
B. The compiler returns an invalid flag error.
C. If the compile is successful Utils.class is added to the classes directory.
D. If the compile is successful Utils.class is added to the bigProgect directory.

Ans :C

174. Program

```
Package  e pack1;
public class L {
        int x;
}
- - - - - - - - - - - - - - - - - - - - - - - - - - -
package pack1;
class N extends L {
        public static void main(String[] args)
        {
                N n1 = new N();
                System.out.println(n1.x);
        }
}
```

175. Program

```
package pack2;
//import pack1.J;
import pack1.L;
class A  {
        public static void main(String[] args)
        {
                //J obj1 = new J();
                //obj1.x = 10;
                L obj2 = new L();
                //obj2.x = 10;
                System.out.println("done");
        }
}
 - - - - - - - - - - - - - - - - - - - - - - - - -
package pack2;
import pack1.L;
class B extends L {
        public static void main(String[] args)
        {
                B b1 = new B();
                //System.out.println(b1.x);
        }
}
```

176. Program

```
package pack1;
public class P {
        int x;
        protected int y;
        public int z;
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
package pack1;
class Q {
        public static void main(String[] args)
        {
                P p1 = new P();
                System.out.println(p1.x);
                System.out.println(p1.y);
                System.out.println(p1.z);
        }
}
```

177. Program

```
package pack1;
public class P {
        int x;
        protected int y;
        public int z;
}
```

```
- - - - - - - - - - - - - - - - - - - - - - - - - - - -
package pack1;
class R extends P {
        public static void main(String[] args)
        {
                P p1 = new P();
                p1.x = 10;
                p1.y = 10;
                p1.z = 10;

                R r1 = new R();
                r1.x = 2;
                r1.y = 2;
                r1.z = 2;
                System.out.println("done");
        }
}
```

178. Program

```
    package pack1;
    class S {

        public static void main(String[] args)
        {
                P p1 = new P();
                p1.x = 10;
                p1.y = 10;
                p1.z = 10;

                R r1 = new R();
                r1.x = 2;
                r1.y = 2;
                r1.z = 2;
                System.out.println("done");
        }
}
```

179. Program

```
package pack1;
public class P {
        int x;
        protected int y;
        public int z;
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - -
package pack2;
import pack1.P;
class C  {
        public static void main(String[] args)
        {
                P p1 = new P();
                //System.out.println(p1.x);
                //System.out.println(p1.y);
                System.out.println(p1.z);
        }
}
```

180. Program

```
package pack1;
public class P {
        int x;
        protected int y;
        public int z;
}
- - - - - - - - - - - - - - - - - - - - - - - - - - - -
package pack2;
import pack1.P;
class D extends P {
        public static void main(String[] args)
        {
                P p1 = new P();
                //p1.x = 1;
                //p1.y = 1;
                p1.z = 1;

                D d1 = new D();
                //d1.x = 2;
                d1.y = 2;
                d1.z = 2;
                System.out.println("done");
        }
}
```

181. Program

Given a file GrizzlyBear.java:

```
1. package animals.mammals;
2.
3. public class GrizzlyBear extends Bear {
4. void hunt() {
5. Salmon s = findSalmon();
6. s.consume();
7. }
8. }
```

and another file, Salmon.java:

```
1. package animals.fish;
2.
3. public class Salmon extends Fish {
4. void consume() { /* do stuff */ }
5. }
```

Assume both classes are defined in the correct directories for theft packages, and that the Mammal class correctly defines the findSalmon() method. Which two changes allow this code to compile correctly? (Choose two.)

A. add public to the start of line 4 in Salmon.java
B. add public to the start of line 4 in GrizzlyBear.java
C. add import animals.mammals.; at line 2 in Salmon.java*
D. add import animals.fish.; at line 2 in GrizzlyBear.java*
E. add import animals.fish.Salmon.; at line 2 in GrizzlyBear.java*
F. add import animals.mammals.GrizzlyBear.;at line 2 in Salmon.java*

Answer: AD

182. Program

Given:

```
1. class Super {
2. private int a;
3. protected Super(int a) { this.a = a; }
4. }
.....
11. class Sub extends Super {
12. public Sub(int a) { super(a); }
13. public Sub() { this.a= 5; }
14. }
```

Which two, independently, will allow Sub to compile? (Choose two.)

A. Change line 2 to:
public int a;
B. Change line 2 to:
protected int a;
C. Change line 13 to:
public Sub() { this(5); }
D. Change line 13 to:
public Sub() { super(5); }
E. Change line 13 to:
public Sub() { super(a); }

Answer: CD

183. Program

```
package pack2;
class E {
        public static void main(String[] args)
        {
                D d1 = new D();
                //System.out.println(d1.y);
                System.out.println(d1.z);
        }
}
```

184. Program

```
package pack2;
class F extends D {
        public static void main(String[] args)
        {
                D d1 = new D();
                //System.out.println(d1.y);
                System.out.println(d1.z);
                F f1 = new F();
                System.out.println(f1.y);
                System.out.println(f1.z);
        }
}
```

185. Program

```
abstract class A {
        abstract void test1();
        void test2()  {
        }
}
```

186. Program

```
abstract class B {
        abstract void test1();
        abstract void test2();
        abstract void test3();
}
```

187. Program

```
abstract class C {
        abstract void test1();
        void test2() {
                System.out.println("test2");
        }
}
class D  {
        public static void main(String[] args)
        {
                //C c1 = new C();
                //c1.test2();
                C c2 = null;
                System.out.println("done");
        }
}
```

188. Program

```
abstract class E {
        abstract void test1();
        void test2()  {
                System.out.println("test2");
        }
}
class F extends E {
        void test1()  {
                System.out.println("test1");
        }
        public static void main(String[] args)
        {
                F f1 = new F();
                f1.test1();
                f1.test2();
                System.out.println("done");
        }
}
```

189. Program

```
abstract class G  {
        void test1()  {
                System.out.println("test1");
        }
        abstract void test2();
        void test3()  {
                System.out.println("test3");
        }
        abstract void test4();
}
class H extends G  {
        void test2()  {
                System.out.println("test2");
        }
        void test4()  {
                System.out.println("test4");
        }
}
class I {
        public static void main(String[] args)
        {
                H obj = new H();
                obj.test1();
                obj.test2();
                obj.test3();
                obj.test4();
                System.out.println("done");
        }
}
```

190. Program

```
abstract class J {
        abstract void test1();
        abstract int test2();
        abstract int test3(double d);
        void test4()  {
                System.out.println("test4");
        }
}
abstract class K extends J  {
        void test1()  {
                System.out.println("test1");
        }
}
class L extends K  {
        int test2()  {
                System.out.println("test2");
                return 10;
        }
        int test3(double d) {
                System.out.println("test3");
                return 20;
        }
}
class M {
        public static void main(String[] args)
        {
                L obj = new L();
                obj.test1();
                obj.test2();
                obj.test3(2.3);
                obj.test4();
                System.out.println("done");
        }
}
```

191. Program

```
abstract class N {
        void test1()  {
                System.out.println("test1");
        }
}
class O extends N {
}
class P {
        public static void main(String[] args)
        {
                //N n1 = new N();
                //n1.test1();
                O o1 = new O();
                o1.test1();
                System.out.println("done");
        }
}
```

192. Program

```
abstract class Q  {
        Q()  {
                System.out.println("Q()");
        }
        abstract void test1();
        abstract void test2();
        void test3()  {
                System.out.println("test3");
        }
}
class R extends Q  {
        R()  {
                System.out.println("R()");
        }
        void test1()  {
                System.out.println("test1");
        }
        void test2()  {
                System.out.println("test2");
        }
}
class S {
        public static void main(String[] args)
        {
                R r1 = new R();
                System.out.println("-----");
                r1.test1();
                r1.test2();
                r1.test3();
        }
}
```

193. Program

```
abstract class T {
        abstract void test1();
        abstract void test2();
        abstract void test3();
        abstract void test4();
}
```

194. Program

```
interface A {

}
```

195. Program

```
interface B {
}
class C  {
}
abstract class D {
}
```

196. Program

```
abstract interface E  {
        abstract void test1();
        void test2();
        public void test3();
}
```

197. Program

```
class F implements E   {
        public void test1()        {
                System.out.println("test1");
        }
        public void test2()        {
                System.out.println("test2");
        }
        public void test3()        {
                System.out.println("test3");
        }
        public static void main(String[] args)
        {
                F f1 = new F();
                f1.test1();
                f1.test2();
                f1.test3();
                System.out.println("done");
        }
}
```

198. Program

Given:

11. public interface Status {
12. /* insert code here */ int MY_VALUE = 10;
13. }

Which three are valid on line 12? (Choose three.)

A. final
B. static
C. native
D. public
E. private
F. abstract
G. protected

Answer: ABD

199. Program

Which four are true? (Choose four.)

A. Has-a relationships should never be encapsulated.
B. Has-a relationships should be implemented using inheritance.
C. Has-a relationships can be implemented using instance variables.
D. Is-a relationships can be implemented using the extends keyword.
E. Is-a relationships can be implemented using the implements
keyword.
F. The relationship between Movie and Actress is an example of an is-a
relationship.
G. An array or a collection can be used to implement a one-to-many

has-a relationship.

Answer: CDEG

200. Program

Which two are true about has-a and is-a relationships? (Choose two.)

A. Inheritance represents an is-a relationship.
B. Inheritance represents a has-a relationship.
C. Interfaces must be used when creating a has-a relationship.
D. Instance variables can be used when creating a has-a relationship.

Answer: AD

201. Program

Given:

10. interface Jumper { public void jump(); }
......
20. class Animal {}
......
30. class Dog extends Animal {
31. Tail tail;
32. }
......
40. class Beagle extends Dog implements Jumper {
41. public void jump() { }
42. }
....
50. class Cat implements Jumper {
51. public void jump() { }
52. }
Which three are true? (Choose three.)

A. Cat is-a Animal
B. Cat is-a Jumper
C. Dog is-a Animal
D. Dog is-a Jumper
E. Cat has-a Animal
F. Beagle has-a Tail
G. Beagle has-a Jumper

Answer: BCF

202. Program

Given:

1. public interface A {
2. String DEFAULT_GREETING = “Hello World”;
3. public void method1();
4. }

A programmer wants to create an interface called B that has A as its parent. Which interface declaration is correct?

A. public interface B extends A { }
B. public interface B implements A {}
C. public interface B instanceOf A {}
D. public interface B inheritsFrom A { }

Answer: A

203. Program

```
interface G  {
          void test1();
          int test2();
          void test3(int i);
}
class H implements G  {
          public void test1() {
          System.out.println("from test1");
          }
          public int test2() {
          System.out.println("from test2");
                    return 10;
          }
          public void test3(int i) {
          System.out.println("from test3");
          }
}
class Manager  {
          public static void main(String[] args)
          {
                    H obj = new H();
                    obj.test1();
                    obj.test2();
                    obj.test3(20);
                    System.out.println("done");
          }
}
```

204. Program

```
interface I {
        void test1();
}
interface J {
        void test2();
}
class K implements I, J {
        public void test1() {
                System.out.println("test1");
        }
        public void test2() {
                System.out.println("test2");
        }
}
class Manager1 {
        public static void main(String[] args)
        {
                K obj = new K();
                obj.test1();
                obj.test2();
                System.out.println("done");
        }
}
```

205. Program

Given:

```
1. interface DoStuff2 {
2. float getRange(int low, int high); }
3.
4. interface DoMore {
5. float getAvg(int a, int b, int c); }
6.
7. abstract class DoAbstract implements DoStuff2, DoMore { }
8.
9. class DoStuff implements DoStuff2 {
10. public float getRange(int x, int y) { return 3.14f; } }
11.
12. interface DoAll extends DoMore {
13. float getAvg(int a, int b, int c, int d); }
```

What is the result?

A. The file will compile without error.

B. Compilation fails. Only line 7 contains an error.

C. Compilation fails. Only line 12 contains an error.

D. Compilation fails. Only line 13 contains an error.

E. Compilation fails. Only lines 7 and 12 contain errors.

F. Compilation fails. Only lines 7 and 13 contain errors.

G. Compilation fails. Lines 7, 12, and 13 contain errors.

Answer: A

206. Program

Given

```
11.public interface A {public void m1();}
12.
13. class B implements A{}
14. class C implements A{pubic void m1(){}}
15. class D implements A{public void m1(int x){}}
16. abstract class E implements A{}
17. abstract class F implements A{public void m1(){}}
18. abstract class G implements A{public void m1(int x){}}
```

What is the result?

A. Compilation success

B. Exactly one class does not compile

C. Exactly two class does not compile

D. Exactly four class does not compile

E. Exactly three class does not compile

Ans : C

207. Program

```
interface L {
        void test1();
}
class M {
        void test2() {
                System.out.println("test2");
        }
}
class N extends M implements L  {
        public void test1() {
                System.out.println("test1");
        }
}
class O  {
        public static void main(String[] args)
        {
                M m1 = new M();
                m1.test2();
                N n1 = new N();
                n1.test1();
                n1.test2();
                System.out.println("done");
        }
}
```

208. Program

```
interface P {
        void test1();
}
interface Q {
        void test2();
}
class R {
        public void test1()        {
                System.out.println("test1");
        }
}
class S extends R implements P, Q  {
        public void test2()        {
                System.out.println("test2");
        }
}
class Manager2 {
        public static void main(String[] args)
        {
                S s1 = new S();
                s1.test1();
                s1.test2();
                System.out.println("done");
        }
}
```

209. Program

```
interface T  {
        void test1();
}
interface U  {
        void test2();
}

interface V extends T, U  {
        void test3();
}
class W implements V  {
        public void test1()        {
                System.out.println("test1");
        }
        public void test2()        {
                System.out.println("test2");
        }
        public void test3()        {
                System.out.println("test3");
        }
}
class Manager3 {
        public static void main(String[] args)
        {
                W obj = new W();
                obj.test1();
                obj.test2();
                obj.test3();
                System.out.println("done");
        }
}
```

210. Program

Given:

```
10. interface Data { public void load(); }
11. abstract class Info { public abstract void load(); }
```

Which class correctly uses the Data interface and Info class?

A. public class Employee extends Info implements Data {
public void load() { /*do something*/ }
}

B. public class Employee implements Info extends Data {
public void load() { /*do something*/ }
}

C. public class Employee extends Info implements Data {
public void load() { /*do something */ }
public void Info.load() { /*do something*/ }
}

D. public class Employee implements Info extends Data {
public void Data.load() { /*d something */ }
public void load() { /*do something */ }
}

E. public class Employee implements Info extends Data {
public void load() { /*do something */ }
public void Info.load(){ /*do something*/
}
}

F. public class Employee extends Info implements Data{
public void Data.load() { /*do something*/ }
public void Info.load() { /*do something*/ }
}

Answer: A

211. Program

```
abstract class X {
	static void test1()		{
		System.out.println("test1");
	}
	static int i = 10;
	void test2()  {
		System.out.println("test2");
	}
}
interface Y {
	static int j = 20;
}
class Z {
	public static void main(String[] args)
	{
		X.test1();
		System.out.println(X.i);
		System.out.println(Y.j);
	}
}
```

212. Program

Given:

11. public abstract class Shape {
12. private int x;
13. private int y;
14. public abstract void draw();
15. public void setAnchor(int x, int y) {
16. this.x = x;
17. this.y = y;
18. }
19. }

Which two classes use the Shape class correctly? (Choose two.)

A. public class Circle implements Shape {
private int radius;
}

B. public abstract class Circle extends Shape {
private int radius;
}

C. public class Circle extends Shape {
private int radius;
public void draw();
}

D. public abstract class Circle implements Shape {
private int radius;
public void draw();
}

E. public class Circle extends Shape {
private int radius;
public void draw() {/* code here */}
}

F. public abstract class Circle implements Shape {

```
private int radius;
public void draw() { / code here */ }
}
```

Answer: BE

213. Program

```
class A {
        void test1(int i) {
        System.out.println("test1(int)");
        }
        int test1()  {
                System.out.println("test1()");
                return 10;
        }
        public void test1(double d) {
        System.out.println("test1(double)");
        }
}
class B  {
        public static void main(String args[])
        {
                A a1 = new A();
                a1.test1(10);
                a1.test1();
                a1.test1(20.9);
        }
}
```

214. Program

```
class D {
        static void test() {
        System.out.println("Hello World!");
        }
        public int test() {
                return 20;
        }
}
```

215. Program

Given:

```
1. public class A {
2. public void doit() {
3. }
4. public String doit() {
5. return "a";
6. }
7. public double doit(int x) {
8. return 1.0;
9. }
10.}
```

What is the result?

A. An exception is thrown at runtime.

B. Compilation fails because of an error in line 7.

C. Compilation fails because of an error in line 4.

D. Compilation succeeds and no runtime errors with class A occur.

Answer: C

216. Program

```
abstract class E {
        abstract void test();
        void test() {
                int i = 10;
        }
}
```

217. Program

```
abstract class F  {
        abstract void test();
        abstract void test(int i);
        void test(int i, int j) {
        System.out.println("test(int, int)");
        }
}
class G extends F {
        void test() {
                System.out.println("test()");
        }
        void test(int i) {
        System.out.println("test(int)");
        }
        public static void main(String[] args)
```

```
    {
        G g1 = new G();
        g1.test();
        g1.test(10);
        g1.test(1, 20);
        System.out.println("done");
    }
}
```

218. Program

```
class H {
    void test1() {
        System.out.println("test1()");
    }
void test2() {
        System.out.println("test2()");
    }
}
class I extends H {
    void test2() {
    System.out.println("modified test2()");
    }
    public static void main(String[] args)
    {
        I obj = new I();
        obj.test1();
        obj.test2();
        System.out.println("done");
    }
}
```

219. Program

```
abstract class J {
    abstract void test(int i);
}
class K extends J {
    void test() {
        System.out.println("test()");
    }
    /*
    void test(int i)   {
        System.out.println("test(int)");
    }
    */
}
```

220. Program

```
class L  {
    void test1() {
        System.out.println("test1()");
    }
}
class M extends L  {
    int test1() {
        System.out.println("test1()");
        return 10;
    }
}
```

221. Program

Given:

```
10. class One {
11. void foo() {}
12. }
13. class Two extends One {
14. //insert method here
15. }
```

Which three methods, inserted individually at line 14, will correctly complete class Two? (Choose three.)

A. int foo() { / more code here */ }*

B. void foo() { / more code here */ }*

C. public void foo() { / more code here */ }*

D. private void foo() { / more code here */ }*

E. protected void foo() { / more code here */ }*

Answer: BCE

222. Program

```
abstract class N {
    abstract int test1();
}
class O extends N {
        void test1() {
        }
}
```

223. Program

Given:

```
10. abstract public class Employee {
11. protected abstract double getSalesAmount();
12. public double getCommision() {
13. return getSalesAmount() * 0.15;
14. }
15. }
16. class Sales extends Employee {
17. // insert method here
18. }
```

Which two methods, inserted independently at line 17, correctly complete the Sales class? (Choose two.)

A. double getSalesAmount() { return 1230.45; }

B. public double getSalesAmount() { return 1230.45; }

C. private double getSalesAmount() { return 1230.45; }

D. protected double getSalesAmount() { return 1230.45; }

Answer: BD

224. Program

Given:

```
10. public class Money {
11. private String country, name;
12. public String getCountry() { return country;
}
13.}
and:
24. class Yen extends Money {
25. public String getCountry() { return
super.country; }
26. }
27.
28. class Euro extends Money {
29. public String getCountry(String timeZone) {
30. return super.getCountry();
31. }
32. }
```

Which two are correct? (Choose two.)

A. Yen returns correct values.

B. Euro returns correct values.

C. An exception is thrown at runtime.

D. Yen and Euro both return correct values.

E. Compilation fails because of an error at line 25.

F. Compilation fails because of an error at line 30.

Answer: BE

225. Program

Given

```
    21. class Money{
    22.private String country =”Canada”;
    23.public Sting getC(){return country;}
    24.}
    25. class Yen extends Money{
    26.public Sting getC(){return
super.country;}

    27.}
      28. public class Euro extends Money{
       29. public String getC(int x){return
                 super.getC();}
    30. public static void main(String
                 args[]){
    31.System.out.println(new
Yen().getC()+””+new Euro().getC());
    32.}
    33.}
```

What is the Result?

A. Canada.
B. Null Canada.
C. Canada Null.
D. Canada Canada.
E. Compilation fails due to an error on line 26.
F. Compilation fails due to an error on line 29.

Ans: E

226. Program

Click the Exhibit button.

```
1. public class SimpleCalc {
2. public int value;
3. public void calculate() { value += 7; }
4. }
And:
1. public class MultiCalc extends SimpleCalc {
```

```
2. public void calculate() { value -= 3; }
3. public void calculate(int multiplier) {
4. calculate();
5. super.calculate();
6. value *=multiplier;
7. }
8. public static void main(String[] args) {
9. MultiCalc calculator = new MultiCalc();
10. calculator.calculate(2);
11. System.out.println(”Value is: “+ calculator.value);
12. }
13. }
```

What is the result?

A. Value is: 8

B. Compilation fails.

C. Value is: 12

D. Value is: -12

E. The code runs with no output.

F. An exception is thrown at runtime.

Answer: A

227. Program

Given:

```
1. public class Blip {
2. protected int blipvert(int x) { return 0; }
3. }
4. class Vert extends Blip {
5. // insert code here
6. }
```

Which five methods, inserted independently at line 5, will compile? (Choose five.)

A. public int blipvert(int x) { return 0; }

B. private int blipvert(int x) { return 0; }

C. private int blipvert(long x) { return 0; }

D. protected long blipvert(int x) { return 0; }

E. protected int blipvert(long x) { return 0; }

F. protected long blipvert(long x) { return 0; }

G. protected long blipvert(int x, int y) { return 0; }

Answer: ACEFG

228. Program

Given

```
2. public class Hi{
3. void m1(){}
4. protected void m2{}
5.}
6. class Lois extends Hi{
7. //insert code here
8.}
```

Which four code fragments, inserted independently at line 7,compile?(Choose four)

A. public void m1(){}

B. protected void m1(){}

C. private void m1(){}

D. void m2(){}

E. public void m2(){}

F. protected void m2(){}

G. private void m2(){}

Ans: A, B, E, F

229. Program

Click the Exhibit button.

```
1. public class GoTest {
2. public static void main(String[] args) {
3. Sente a = new Sente(); a.go();
4. Goban b = new Goban(); b.go();
5. Stone c = new Stone(); c.go();
6. }
7. }
8.
9. class Sente implements Go {
10. public void go() { System.out.println(”go in Sente.”); }
11. }
12.
13. class Goban extends Sente {
14. public void go() { System.out.println(”go in Goban”); }
15. }
16.
17. class Stone extends Goban implements Go { }
18.
19. interface Go { public void go(); }
```

What is the result?

A. go in Goban
   go in Sente
   go in Sente

B. go in Sente
   go in Sente
   go in Goban

C. go in Sente
    go in Goban
    go in Goban
D. go in Goban
    go in Goban
    go in Sente
E. Compilation fails because of an error in line 17.

Answer: C

230. **Program**

Given:

```
10. public class SuperCaic {
11. protected static int multiply(int a, int b) {
return a * b; }
12. }
and:
20. public class SubCalc extends SuperCalc {
21. public static int multiply(int a, int b) {
22. int c = super.multiply(a, b);
23. return c;
24. }
25. }
and:
30. SubCalc sc = new SubCalc();
31. System.out.println(sc.multiply(3,4));
32. System.out.println(SubCalc.multiply(2,2));
```

What is the result?

A. 12 4
B. The code runs with no output.
C. An exception is thrown at runtime.
D. Compilation fails because of an error in line 21.
E. Compilation fails because of an error in line 22.
F. Compilation fails because of an error in line 31.

Answer: E

231. Program

```
class P {
}
class Q extends P {
}
class R extends Q {
}
class S {
        Q test() {
                return null;
        }
}
class T extends S {
        Q test() {
                return null;
        }
}
```

232. Program

Given:

```
10. class One {
11. public One foo() { return this; }
12. }
13. class Two extends One {
14. public One foo() { return this; }
15. }
16. class Three extends Two {
17. // insert method here
18. }
```

Which two methods, inserted individually, correctly complete the Three class? (Choose two.)

A. public void foo() { }
B. public int foo() { return 3; }
C. public Two foo() { return this; }
D. public One foo() { return this; }
E. public Object foo() { return this; }

Answer: CD

233. Program

Click the Exhibit button.

```
1. public interface A {
2. public void doSomething(String thing);
3. }
1. public class AImpl implements A {
2. public void doSomething(String msg) { }
3. }
1. public class B {
2. public A doit() {
3. // more code here
4. }
5.
6. public String execute() {
```

```
7. // more code here
8. }
9. }
1. public class C extends B {
2. public AImpl doit() {
3. // more code here
4. }
5.
6. public Object execute() {
7. // more code here
8. }
9. }
```

Which statement is true about the classes and interfaces in the
exhibit?

A. Compilation will succeed for all classes and interfaces.

B. Compilation of class C will fail because of an error in line 2.

C. Compilation of class C will fail because of an error in line 6.

D. Compilation of class AImpl will fail because of an error in line 2.

Answer: C

234. Program

```
abstract class U {
        public abstract void test1();
}
class V extends U {
        void test1() {

        }
}
```

235. Program

Given:

```
1. class SuperClass {
2. public A getA() {
3. return new A();
4. }
5. }
6. class SubClass extends SuperClass {
7. public B getA() {
8. return new B();
9. }
10. }
```

Which is true?

A. Compilation will succeed if A extends B.

B. Compilation will succeed if B extends A.

C. Compilation will always fail because of an error in line 7.

D. Compilation will always fail because of an error in line 8.

Answer: B

236. Program

```
class A {
        public static void main(String[] args)
        {
                int i = 10;
                double d = i;
                System.out.println("done");
        }
}
```

237. Program

```
class B {
        public static void main(String[] args)
        {
                byte b = 10;
                int i = b;
                double d = i;
                float f = b;
                long l = i;
                System.out.println("done");
        }
}
```

238. Program

```
class C {
        public static void main(String[] args)
        {
                int i = 10;
                test(i);
        }
        static void test(double d) {
        System.out.println("test(double)");
```

```
    }
}
```

239. Program

```
class D {
    public static void main(String[] args)
    {
        int i = test();
        System.out.println("done");
    }
    static byte test() {
        return 10;
    }
}
```

240. Program

```
class E {
    public static void main(String[] args)
    {
        int i = 10;
        double d = (double) i;
        System.out.println("done");
    }
}
```

241. Program

```
class F {
    static long test() {
        int i = 10;
        return i;
    }
    public static void main(String[] args)
    {
        double d = test();
        System.out.println("done");
    }
}
```

242. Program

```
class G {
    static float test1(short s) {
        return test2(s);
    }
    static long test2(int i) {
        return i;
    }
    public static void main(String[] args)
    {
        byte b = 10;
        double d = test1(b);
        System.out.println("done");
    }
}
```

243. Program

```
class H {
    public static void main(String[] args)
    {
        double d = 10.9;
        int i = (int)d;
        System.out.println(d);
        System.out.println(i);
    }
}
```

244. Program

```
class I {
    public static void main(String[] args)
    {
        long x = 10;
        test((int)x);
        System.out.println("done");
    }
    static void test(int i) {

    }
}
```

245. Program

```
class J {
    public static void main(String[] args)
    {
        byte a = 10;
        short b = 20;
        int c = 10;
        long d = 20;
        float f = 20;
        double g = 20.09;
        a = b; //1
        b = a; //2
        d = c; //3
        c = d; //4
        g = f; //5
        f = g; //6
        System.out.println("done");
    }
}
```

246. Program

```
class K {
    static int test(long x) {
        return (int)x;
    }
    public static void main(String[] args)
    {
        int i = 10;
        byte b = (byte)test(i);
        System.out.println("done");
    }
}
```

247. Program

```
class L {
    public static void main(String[] args)
    {
        double d = 20;
        int i = (byte) d;
        System.out.println("done");
    }
}
```

248. Program

```
class M {
    public static void main(String[] args)
    {
    double d = 20.9;
    int i = (short)(int)(byte)(long) d;
    System.out.println("done");
    }
}
```

- -

```
class A
{
}
class B extends A
{
}
class C extends B
{
}
class D extends C
{
}
class E extends D
{
}
class F extends E
{
}
```

General examples in diff   .java files

- -

249. Program

```
class Manager1 {
    public static void main(String[] args)
    {
        A a1 = new A();
        B b1 = new B();
        C c1 = new C();
        D d1 = new D();
        E e1 = new E();
        F f1 = new F();
        System.out.println("done");
    }
}
```

250. Program

```
class Manager2 {
    public static void main(String[] args)
    {
        A a1 = new A();
        A a2 = a1;
        B b1 = new B();
        B b2 = b1;
        C c1 = null;
        C c2 = c1;
        D d1, d2;
        d1 = d2 = new D();
        E e1, e2 = new E();
        e1 = e2;
        F f1, f2 = null;
        f1 = f2;
        System.out.println("done");
    }
}
```

251. Program

```
class Manager3 {
    public static void main(String[] args)
    {
        A a1 = new B();
        B b1 = new D();
        C c1 = new E();
        D d1 = new F();
        System.out.println("done");
    }
}
```

252. Program

```
class Manager4 {
    public static void main(String[] args)
    {
        B b1 = null;
        C c1 = null;
        D d1 = new D();
        E e1 = null;
        F f1 = new F();
        b1 = f1;
        c1 = e1;
        e1 = f1;
        b1 = d1;
        d1 = e1;
        System.out.println("done");
    }
}
```

253. Program

```
class Manager5 {
    public static void main(String[] args)
    {
        E e1 = new E();
        test(e1);
    }
    static void test(C c1) {
        System.out.println("done");
    }
}
```

254. Program

```
class Manager6  {
    static D test()    {
        F f1 = new F();
        return f1;
    }
    public static void main(String[] args)
    {
        B b1 = test();
        System.out.println("done");
    }
}
```

255. Program

```
class Manager7 {
    static B test1(E e1) {
        return test2(e1);
    }
    static C test2(D d1)  {
```

```
		return d1;
	}
	public static void main(String[] args)
	{
		F f1 = new F();
		A a1 = test1(f1);
		System.out.println("done");
	}
}
```

256. Program

Given:

```
11. public abstract class Shape {
12. int x;
13. int y;
14. public abstract void draw();
15. public void setAnchor(int x, int y) {
16. this.x = x;
17. this.y = y;
18. }
19. }
```

and a class Circle that extends and fully implements the Shape class.

Which is correct?

A. Shape s = new Shape();
s.setAnchor(10,10);
s.draw();

B. Circle c = new Shape();
c.setAnchor(10,10);
c.draw();

C. Shape s = new Circle();
s.setAnchor(10,10);
s.draw();

D. Shape s = new Circle();
s->setAnchor(10,10);
s->draw();

E. Circle c = new Circle();
c.Shape.setAnchor(10,10);
c.Shape.draw();

Answer: C

257. Program

```
class Manager8 {
	public static void main(String[] args)
	{
		A a1 = new C();
		B b1 = (B) a1;
		System.out.println("done");
	}
}
```

258. Program

```
class Manager9 {
	public static void main(String[] args)
	{
		D d1 = new F();
		E e1 = (E) d1;
		System.out.println("done");
	}
}
```

259. Program

```
class Manager10 {
	public static void main(String[] args)
	{
		E e1 = new F();
		test((F)e1);
		System.out.println("done");
	}
	static void test(F f1) {
		System.out.println("test(F)");
	}
}
```

260. Program

```
class Manager11 {
	public static void main(String[] args)
	{
		A a1 = new A();
		B b1 = (B)a1;
		System.out.println("done");
	}
}
```

261. Program

```
class Manager12 {
        public static void main(String[] args)
        {
                A a1 = new B();
                B b1 = (B) a1;
                System.out.println("done");
        }
}
```

262. Program

Given:

1. interface A { public void aMethod(); }
2. interface B { public void bMethod(); }
3. interface C extends A,B { public void cMethod(); }
4. class D implements B {
5. public void bMethod() { }
6. }
7. class E extends D implements C {
8. public void aMethod() { }
9. public void bMethod() { }
10. public void cMethod() { }
11. }

What is the result?

A. Compilation fails because of an error in line 3.
B. Compilation fails because of an error in line 7.
C. Compilation fails because of an error in line 9.
D. If you define D e = new E(), then e.bMethod() invokes the version of bMethod() defined in Line 5.
E. If you define D e = (D)(new E()), then e.bMethod() invokes the version of bMethod() defined in Line 5.
F. If you define D e = (D)(new E()), then e.bMethod() invokes the version of bMethod() defined in Line 9.

Answer: F

263. Program

```
class Manager13 {
        public static void main(String[] args)
        {
                B b1 = new D();
                C c1 = (C) b1;
                D d1 = (D) c1;
                System.out.println("done");
        }
}
```

264. Program

Given

11. class Alpha{
12.publicvoidfoo(){System.out.println(“ Afoo”);}
13.}
14. public class Beta extends Alpha{
15.publicvoidfoo(){System.out.println(“ Bfoo”);}
16. public static void main (String args[])
{
17. Alpha a=new Beta();
18. Beta b=(Beta)a;
19. a.foo();
20. b.foo();
}

What is the result?

A. Afoo Afoo
B. Afoo Bfoo
C. Bfoo Afoo
D. Bfoo Bfoo
E. CompilationFails
F. An exception is thrown at runtime

Ans: D

265. Program

Given:

1. class TestA {
2. public void start() { System.out.println(”TestA”); }
3. }
4. public class TestB extends TestA {
5. public void start() { System.out.println(”TestB”); }
6. public static void main(String[] args) {
7. ((TestA)new TestB()).start();
8. }
9. }

What is the result?

A. TestA
B. TestB
C. Compilation fails.
D. An exception is thrown at runtime.

Answer: B

266. Program

```
class Manager14 {
        public static void main(String[] args)
        {
                B b1 = new E();
                C c1 = (C) b1;
                D d1 = (D) b1;
                E e1 = (E) b1;
                F f1 = (F) b1;
                System.out.println("done");
        }
}
```

267. **Program**

Given:

```
11. class ClassA {}
12. class ClassB extends ClassA {}
13. class ClassC extends ClassA {}
and:
21. ClassA p0 = new ClassA();
22. ClassB p1 = new ClassB();
23. ClassC p2 = new ClassC();
24. ClassA p3 = new ClassB();
25. ClassA p4 = new ClassC();
```

Which three are valid? (Choose three.)
A. p0 = p1;
B. p1 =p2;
C. p2 = p4;
D. p2 = (ClassC)p1;
E. p1 = (ClassB)p3;
F. p2 = (ClassC)p4;

Answer: AEF

268. **Program**

Given:

```
11. class Animal { public String noise()
{ return "peep"; } }
12. class Dog extends Animal {
13. public String noise() { return "bark";
}
14. }
15. class Cat extends Animal {
16. public String noise() { return
"meow"; }
17. }
.....
30. Animal animal = new Dog();
31. Cat cat = (Cat)animal;
32. System.out.printIn(cat.noise());
```

What is the result?
A. peep
B. bark
C. meow
D. Compilation fails.
E. An exception is thrown at runtime.

Answer: E

269. **Program**

Given:

```
10. interface Foo {}
11. class Alpha implements Foo { }
12. class Beta extends Alpha {}
13. class Delta extends Beta {
14. public static void main( String[]
args) {
15. Beta x = new Beta();
16. // insert code here
17. }
18. }
```

Which code, inserted at line 16, will cause a java.lang.ClassCastException?
A. Alpha a = x;
B. Foo f= (Delta)x;
C. Foo f= (Alpha)x;
D. Beta b = (Beta)(Alpha)x;

Answer: B

270. **Program**

```
class Manager15 {
        public static void main(String[] args)
        {
        A a1 = new B();
        System.out.println(a1 instanceof A);
```

```
		System.out.println(a1 instanceof B);
		System.out.println(a1 instanceof C);
		System.out.println(a1 instanceof D);
		}
}
```

271. Program

```
class Manager16 {
		public static void main(String[] args)
		{
		C c1 = new D();
		System.out.println(c1 instanceof A);
		System.out.println(c1 instanceof B);
		System.out.println(c1 instanceof C);
		System.out.println(c1 instanceof D);
		System.out.println(c1 instanceof E);
		System.out.println(c1 instanceof F);
		}
}
```

272. Program

Given:
11. class Cup { }
12. class PoisonCup extends Cup { }
21. public void takeCup(Cup c) {
22. if(c instanceof PoisonCup) {
23. System.out.println(”Inconceivable!”);
24. } else if(c instanceof Cup) {
25. System.out.println(”Dizzying intellect!”);
26. } else {
27. System.exit(0);
28. }
29. }
And the execution of the statements:
Cup cup = new PoisonCup();
takeCup(cup);
What is the output?

A. Inconceivable!
B. Dizzying intellect!
C. The code runs with no output.
D. An exception is thrown at runtime.
E. Compilation fails because of an error in line 22.

273. Program

```
class Manager17 {
		public static void main(String[] args)
		{
				A a1 = new C();
				if(a1 instanceof A)	{
						A a2 = (A) a1;
						System.out.println(1);
				}
				if(a1 instanceof B)	{
						B b1 = (B) a1;
						System.out.println(2);
				}
				if(a1 instanceof C)	{
						C c1 = (C) a1;
						System.out.println(3);
				}
				if(a1 instanceof D)	{
						D d1 = (D) a1;
						System.out.println(4);
				}
				if(a1 instanceof E)	{
						E e1 = (E) a1;
						System.out.println(5);
				}
				if(a1 instanceof F)	{
						F f1 = (F) a1;
						System.out.println(6);
				}
				System.out.println("done");
		}
}
```

274. Program

```
class A {
		int i;
		void test1()  {
				System.out.println("A-test1");
		}
}
class B extends A {
		int j;
```

```
        void test2() {
        System.out.println("B-test2");
        }
}
class Manager1 {
        public static void main(String[] args)
        {
                A a1 = new A();
                a1.test1();
                a1.i = 10;
                System.out.println(a1.i);
                B b1 = new B();
                b1.test1();
                b1.i = 10;
                b1.test2();
                b1.j = 20;
                System.out.println(b1.i);
                System.out.println(b1.j);
        }
}
```

275. Program

```
class Manager2 {
        public static void main(String[] args)
        {
                A a1 = new B();
                a1.i = 10;
                a1.test1();
                //a1.j = 20;
                //a1.test2();
                System.out.println("done");
        }
}
```

276. Program

```
class Manager3 {
        public static void main(String[] args)
        {
                A a1 = new B();
                a1.i = 10;
                a1.test1();
                B b1 = (B) a1;
                b1.j = 20;
                b1.test2();
                b1.i = 30;
                b1.test1();
                System.out.println("done");
        }
}
```

277. Program

```
class C {
        static void test1() {
        System.out.println("C-test1");
        }
}
class D extends C {
        static void test1() {
        System.out.println("D-test1");
        }
}
class Manager4 {
        static void method(C c1) {
                c1.test1();
        }
        public static void main(String[] args)
        {
                C obj1 = new C();
                D obj2 = new D();
                method(obj1);
                method(obj2);
        }
}
```

278. Program

Given:

```
10. abstract class A {
11. abstract void al();
12. void a2() { }
13. }
14. class B extends A {
15. void a1() { }
16. void a2() { }
```

17. }
18. class C extends B { void c1() { } }
and:
A x = new B(); C y = new C(); A z = new C();

Which four are valid examples of polymorphic method calls? (Choose four.)

A. x.a2();
B. z.a2();
C. z.c1();
D. z.a1();
E. y.c1();
F. x.a1();

Answer: ABDF

279. Program

```
class Manager4  {
        static    {
                System.out.println("Manager4-
SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("Manager4-main");
        }
}
class Manager5 extends  Manager4 {
        static    {
        System.out.println("Manager5-SIB");
        }
}
class Manager6 {
        static    {
        System.out.println("Manager6-SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("Manager6-main-
b");
        Manager4.main(args);
        System.out.println("Manager6-main-
e");
        }
}
class Manager7 {
        static    {
        System.out.println("Manager7-SIB");
        }
        public static void main(String[] args)
        {
        System.out.println("Manager7-main-
b");
        Manager5.main(args);
        System.out.println("-----");
        Manager4.main(args);
        System.out.println("Manager7-main-
e");
        }
}
```

280. Program

Given:
10. interface A { public int getValue() }
11. class B implements A {
12. public int getValue() { return 1; }
13. }
14. class C extends B {
15. // insert code here
16. }

Which three code fragments, inserted individually at line 15, make use of polymorphism? (Choose three.)

A. public void add(C c) { c.getValue(); }
B. public void add(B b) { b.getValue(); }
C. public void add(A a) { a.getValue(); }
D. public void add(A a, B b) { a.getValue(); }
E. public void add(C c1, C c2) { c1.getValue(); }

Answer: BCD

281. Program

Given:
20. public class CreditCard {
21.
22. private String cardlD;
23. private Integer limit;
24. public String ownerName;

25.
26. public void setCardlnformation(String cardlD,
27. String ownerName,
28. Integer limit) {
29. this.cardlD = cardlD;
30. this.ownerName = ownerName;
31. this.limit = limit;
32. }
33. }
Which is true?
A. The class is fully encapsulated.
B. The code demonstrates polymorphism.
C. The ownerName variable breaks encapsulation.
D. The cardlD and limit variables break polymorphism.
E. The setCardlnformation method breaks encapsulation.

Answer: C

282. Program

Which two are true? (Choose two.)
A. An encapsulated, public class promotes re-use.
B. Classes that share the same interface are always tightly
encapsulated.
C. An encapsulated class allows subclasses to overload methods, but
does NOT allow overriding methods.
D. An encapsulated class allows a programmer to change an
implementation without affecting outside code.

Answer: AD

283. Program

```
class A  {
        public static void main(String[] args)
        {
                int i = 0;
                final int j = 1;
                System.out.println(i);
                System.out.println(j);
                i = 10;
                //j = 1;
                System.out.println(i);
                System.out.println(j);
        }
}
```

284. Program

```
class B {
        public static void main(String[] args)
        {
                final int i = 10;
                //i++;
                System.out.println(i);
        }
}
```

285. Program

```
class C {
        public static void main(String[] args)
        {
                final int[] x = new int[2];
                x[0] = 10;
                x[1] = 20;
                System.out.println(x[0]);
                System.out.println(x[1]);
        }
}
```

286. Program

```
class D {
        public static void main(String[] args)
        {
                final int[] x = new int[3];
                x = new int[3];
                System.out.println("done");
        }
}
```

287. Program

```
class E {
        int i;
```

```
	public static void main(String[] args)
	{
		final E e1 = new E();
		e1.i = 20;
		System.out.println("done");
	}
}
```

288. Program

```
class F {
	int i;
	public static void main(String[] args)
	{
		final F f1 = new F();
		//f1 = new F();
		//f1 = null;
		System.out.println("done");
	}
}
```

289. Program

```
class G {
	public static void main(final String[]
args) {
	System.out.println("Hello World!");
	//args = null;
	}
}
```

290. Program

```
class  H{
	final int i = 10;
}
class Manager {
	public static void main(String[] args)
	{
		H obj = new H();
		System.out.println(obj.i);
		obj.i = 10;
		System.out.println(obj.i);
	}
}
```

291. Program

```
Class  I{
	final int x = 10;
	void test() {
		x = 10;
	}
}
```

292. Program

```
class  J {
	final int i;
}
class Manager1 {
	public static void main(String[] args)
	{
		J obj = new J();
		System.out.println(obj.i);
	}
}
```

293. Program

```
class  K {
	final int i;
}
```

294. Program

```
class  L{
	final int i;
	L() {
		i = 10;
	}
}
```

295. Program

```
class M {
	final int i = 10;
	M() {
		 i = 10;
	}
}
```

296. Program

```
class  N {
        final int i;
        N() {
                i = 20;
        }
        N(int j)  {

        }
}
```

297. Program

```
class O {
        final int i;
        O() {
                i = 10;
        }
        O(int i)  {
                i = 20;
        }
}
```

298. Program

```
class  P {
        final int i;
        P() {
                i = 10;
        }
        P(int i)  {
                this.i = i;
        }
}
```

299. Program

```
class  Q{
        final int i;
        Q() {
                i = 10;
        }
        Q(String s1) {
                this();
        }
}
```

300. Program

```
class   R{
        final int i;


{
                i = 0;
        }
}
```

301. Program

```
class  S{
        final int i;
        S() {
                i = 0;
        }

        {
                i = 0;
        }
}
```

302. Program

```
class  T{
        final int i;

        {
                i = 0;
        }

        {
                i = 0;
        }
}
```

303. Program

```
class U
```

```
{
        U(int i, final int j)
        {
                i = 20;
                j = 20;
        }
}
```

304. Program

```
class  V {
        static final int i = 10;
        public static void main(String[] args)
        {
                System.out.println(i);
                i = 20;
                System.out.println(i);
        }
}
```

305. Program

```
class W{
        final static int i;
}
```

306. Program

```
class  X{
        static final int i;
        static    {
                i = 0;
        }
}
```

307. Program

```
class  Y{
        static final int i = 0;
        static    {
                i = 0;
        }
}
```

308. Program

```
class  Z{
        final static int i;
        static    {
                i = 0;
        }
        static    {
                i = 0;
        }
}
```

309. Program

```
class C {
        void test1() {

        }
        final void test2() {

        }
}
class D extends C {
        void test1() {

        }
        void test2() {


        }
```

310. Program

Given:

1. class Pizza {
2. java.util.ArrayList toppings;
3. public final void addTopping(String topping) {
4. toppings.add(topping);
5. }
6. }
7. public class PepperoniPizza extends Pizza {
8. public void addTopping(String topping) {
9. System.out.println(”Cannot add Toppings”);
10. }
11. public static void main(String[] args) {
12. Pizza pizza = new PepperoniPizza();
13. pizza.addTopping(”Mushrooms”);
14. }
15. }

What is the result?
A. Compilation fails.
B. Cannot add Toppings
C. The code runs with no output.
D. A NullPointerException is thrown in Line4.

Answer: A

311. Program

```
final class F  {
	//some members
}
class G extends F
{
}
```

312. Program

```
abstract class E {
	final abstract void test1();
}
```

313. Program

```
abstract final classH
{
}
```

314. Program

```
abstract class I {
	public static void main(String[] args)
	{
	System.out.println("Hello World!");
	}
}
```

315. Program

```
abstract class J {
	static abstract void test1();
}
```

316. Program

```
interface A {
	int i;
}
```

317. Program

```
interface B {
	String s1 = null;
	int j = 0;
}
```

318. Program

```
class A
{
	int i;
	static int j;
	public static void main(String[] args)
	{
		System.out.println(j);
		System.out.println(A.j);

		A a1 = new A();
		System.out.println(a1.i);
		System.out.println(a1.j);
	}
}
```

319. Program

```
class B
{
	int i;
	void test1()
	{
		System.out.println("test1:" + i);
		i = 10;
		test2();
	}
	void test2()
	{
		System.out.println("test2:" + i);
		i = 20;
		test3();
	}
	void test3()
	{
		System.out.println("test3:" + i);
		i = 30;
	}
	public static void main(String[] args)
	{
		B b1 = new B();
	System.out.println("main1:" + b1.i);
		b1.i = 5;
		b1.test1();
	System.out.println("main2:" + b1.i);
	}
}
```

320. Program

Given:

```
interface DeclareStuff{
public static final int EASY = 3;
void doStuff(int t); }
public class TestDeclare implements DeclareStuff {
public static void main(String [] args) {
int x=5;
new TestDeclare().doStuff(++x);
}
void doStuff(int s) {
s += EASY + ++s;
System.out.println(”s “ + s);
}
}
```

What is the result?

A. s 14
B. s 16
C. s 10
D. Compilation fails.
E. An exception is thrown at runtime.

Answer: D

321. Program

```
class G
{
	int i;
	G obj;
	public static void main(String[] args)
	{
		G g1 = new G();
		System.out.println(g1.i);
		System.out.println(g1.obj);
		g1.obj = new G();
		g1.obj.i = 10;
		g1.i = 20;
		System.out.println(g1.i);
		System.out.println(g1.obj.i);
	}
}
```

322. Program

Given:

```
public class Base {
public static final String FOO = “foo”;
public static void main(String[] args) {
Base b = new Base();
Sub s = new Sub();
System.out.print(Base.FOO);
System.out.print(Sub.FOO);
System.out.print(b.FOO);
System.out.print(s.FOO);
	System.out.print(((Base)s).FOO);
	} }
	class Sub extends Base {public static final String FOO=bar;}
```

What is the result?

A. foofoofoofoofoo
B. foobarfoobarbar
C. foobarfoofoofoo
D. foobarfoobarfoo
E. barbarbarbarbar
F. foofoofoobarbar
G. foofoofoobarfoo

Answer: D

323. Program

```
class C
{
	int i;
}
class D
{
	C c1;
public static void main(String[] args)
	{
		D d1 = new D();
		System.out.println(d1.c1);
	}
}
class E
{
	C c1 = new C();

	public static void main(String[] args)
```

```
        {
            E e1 = new E();
            System.out.println(e1.c1);
            System.out.println(e1.c1.i);
        }
    }
    class F
    {
        C c1 = null;
        public static void main(String[] args)
        {
            F f1 = new F();
            f1.c1 = new C();
            System.out.println(f1.c1);
            System.out.println(f1.c1.i);
        }
    }
```

324. **Program**

```
    class P
    {
        int i = 1;
    }
    class Q extends P
    {
        int i = 2;
    }
    class R extends Q
    {
        int i = 3;
    }
    class Manager1
    {
        public static void main(String[] args)
        {
            P p1 = new P();
            System.out.println(p1.i);

            P p2 = new Q();
            System.out.println(p2.i);

            Q q1 = new Q();
            System.out.println(q1.i);

            R r1 = new R();
            System.out.println(r1.i);

            P p3 = r1;
            System.out.println(p3.i);
            System.out.println(((Q)p3).i);
        }
    }
```

325. **Program**

Given:
10. public class Foo {
11. public int a;
12. public Foo() { a = 3; }
13. public void addFive() { a += 5; }
14. }
and:
20. public class Bar extends Foo {
21. public int a;
22. public Bar() { a = 8; }
23. public void addFive() { this.a +=5; }
24. }
invoked with:
30. Foo foo = new Bar();
31. foo.addFive();
32. System.out.println(”Value: “+ foo.a);
What is the result?
A. Value: 3
B. Value: 8
C. Value: 13
D. Compilation fails.
E. The code runs with no output.
F. An exception is thrown at runtime.

Answer: A

326. **Program**

```
class A
{
void test1()
{
System.out.println("test1:" + this);
        test2();  // this.test2();
}
void test2()
{
System.out.println("test2:" + this);
        test3();  // this.test3();
}
```

```
    void test3()
    {
    System.out.println("test3:" + this);
    }
    public static void main(String[] args)
    {
        A a1 = new A();
    System.out.println("main:" + a1);
        a1.test1();
    }
}

class B
{
    int i;

    void test1()
    {
        System.out.println("test1:" + i);
        i = 10;
        test2();
    }
    void test2()
    {
        System.out.println("test2:" + i);
        i = 20;
        test3();
    }
    void test3()
    {
        System.out.println("test3:" + i);
        i = 30;
    }
    public static void main(String[] args)
    {
        B b1 = new B();
    System.out.println("main1:" + b1.i);
        b1.i = 5;
        b1.test1();
    System.out.println("main2:" + b1.i);
    }
}
```

327. **Program**

```
class C
{
    int i;
}
class D
{
    C c1;
    public static void main(String[] args)
    {
        D d1 = new D();
        System.out.println(d1.c1);
    }
}
class E
{
    C c1 = new C();
    public static void main(String[] args)
    {
        E e1 = new E();
        System.out.println(e1.c1);
        System.out.println(e1.c1.i);
    }
}
class F
{
    C c1 = null;
    public static void main(String[] args)
    {
        F f1 = new F();
        f1.c1 = new C();
        System.out.println(f1.c1);
        System.out.println(f1.c1.i);
    }
}
```

328. **Program**

```
class G
{
    int i;
    G obj;
    public static void main(String[] args)
    {
        G g1 = new G();
        System.out.println(g1.i);
        System.out.println(g1.obj);
        g1.obj = new G();
        g1.obj.i = 10;
        g1.i = 20;
        System.out.println(g1.i);
        System.out.println(g1.obj.i);
    }
}
```

329. Program

```
class H
{
        int i;
        H obj;
        public static void main(String[] args)
        {
                H h1 = new H();
                System.out.println(h1.i);
                System.out.println(h1.obj.i);
        }
}
```

330. Program

```
class P
{
        int i = 1;
}
class Q extends P
{
        int i = 2;
}
class R extends Q
{
        int i = 3;
}
class Manager1
{
        public static void main(String[] args)
        {
                P p1 = new P();
                System.out.println(p1.i);
                P p2 = new Q();
                System.out.println(p2.i);
                Q q1 = new Q();
                System.out.println(q1.i);
                R r1 = new R();
                System.out.println(r1.i);
                P p3 = r1;
                System.out.println(p3.i);
                System.out.println(((Q)p3).i);
        }
}
```