1. The Kotlin code is compiled to:

   a) Java source code
   b) Java byte code

2. Does the following code compile? If not, what's the error?

```
var string = 1
string = "a"
```

3. Does the following code compile? If not, what's the error?

```
val languages = listOf("Java")
languages.add("Kotlin")
```

4. What will be printed? ____________________

```
println(listOf('a', 'b', 'c').joinToString(
        separator = "", prefix = "(", postfix = ")"))
```

5. What will be printed (a, b or c)?

```
fun foo(): String {
    println("Calculating foo...")
    return "foo"
}

fun main(args: Array<String>) {
    println("First ${foo()}, second ${foo()}")
}
```

```
a) Calculating foo...
   First foo, second foo

b) Calculating foo...
   Calculating foo...
   First foo, second foo

c) First Calculating foo...
   foo, second Calculating foo...
   foo
```

6. What will be printed?

```
println("Kotlin" in "Java".."Scala")           ____________________

println("Kotlin" in setOf("Java", "Scala"))    ____________________
```

7. Rewrite the following Java code into Kotlin:

```
for (char c = '0'; c < '9'; c++) {
    System.out.print(c);
}
```

____

____

____

8. How many methods does the class Person have from Java's point of view?

```
class Person(val name: String, var age: Int) ________________
```

9. What will be printed? (The second declaration adds `data` modifier)

```
class Evaluation1(val positions: Int, val letters: Int)

>>> val e1 = Evaluation1(1, 3)
>>> val e2 = Evaluation1(1, 3)
>>> println(e1 == e2)            ____________________
```

```
data class Evaluation2(val positions: Int, val letters: Int)

>>> val e1 = Evaluation2(1, 3)
>>> val e2 = Evaluation2(1, 3)
>>> println(e1 == e2)            ____________________
```

10. Find the correspondence between modifiers and their meaning:

	Modifier			**Meaning**
a	`final` (by default)		1	overrides a member in a superclass or interface
b	`open`		2	must be overridden (can't have an implementation)
c	`abstract`		3	cannot be overridden
d	`override` (mandatory)		4	can be overridden

11. Fill the table with the values:
everywhere, in a module, in a file, in a class, in subclasses

Modifier	**Class member**	**Top-level declaration**
`public` (by default)	visible ?	visible ?
`internal`	visible ?	visible ?
`protected`	visible ?	———
`private`	visible ?	visible ?

12. Which class (nested or inner) stores a reference to an outer class? ____________________

In Java	**In Kotlin**	**Class declared within another class**
`static class A`	`class A` (by default)	nested class
`class A` (by default)	`inner class A`	inner class

13. Top-level function declared in the file `Util.kt` is compiled to:

a) a regular non-`final` method of the class UtilKt
b) a `final` method of the class UtilKt
c) a `static` method of the class UtilKt

14. How many times the phrase "Calculating the answer..." will be printed?

```
val foo1 = run {
    println("Calculating the answer...")
    42
}

val foo2: Int
    get() {
        println("Calculating the answer...")
        return 42
    }

fun main(args: Array<String>) {
    println("$foo1 $foo1 $foo2 $foo2")
}
```

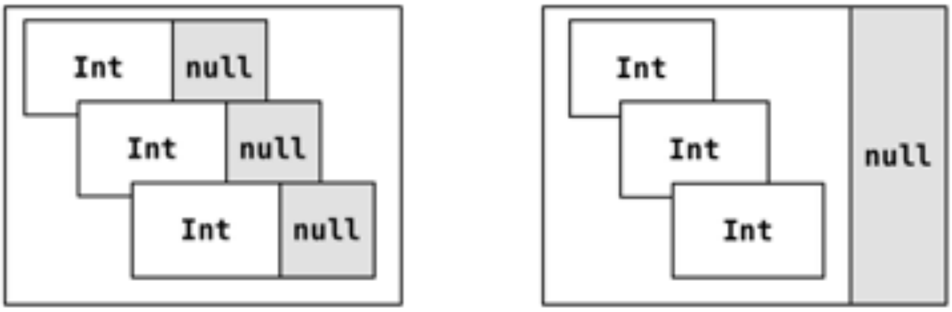
15. Which line(s) won't compile?

```
 1  class Name(val value: String?)

 2  fun isFoo1(n: Name) = n.value == "foo"
 3  fun isFoo2(n: Name?) = n.value == "foo"
 4  fun isFoo3(n: Name?) = n != null && n.value == "foo"
 5  fun isFoo4(n: Name?) = n?.value == "foo"

 6  fun main(args: Array<String>) {
 7      isFoo1(null)
 8      isFoo2(null)
 9      isFoo3(null)
10      isFoo4(null)
11  }
```

16. Add question marks when necessary to make the code compile.



List<Int?>   List<Int>?

```
fun foo(list1: List<Int?>, list2: List<Int>?) {
    list1 . size
    list2 . size

    val i: Int  = list1 . get(0)
    val j: Int  = list2 . get(0)
}
```

17. Implement an extension function `isNullOrEmpty` on the type `String?`. It should return `true`, if the string is empty or `null`.

```
val s1: String? = null
val s2: String? = ""
```

```
    println(s1.isNullOrEmpty() && s2.isNullOrEmpty())   // true
```

____

____

____

18. Type cast **as** throws `ClassCastException`, if the cast is unsuccessful. Safe cast **as?** returns **null**, if the cast is unsuccessful. In which of the following cases the exception will be thrown?

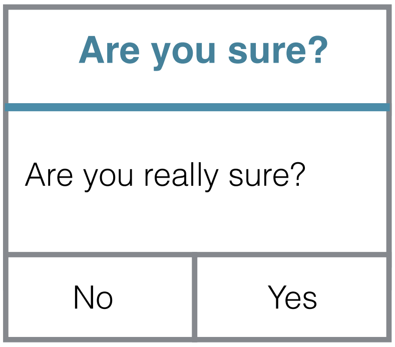
```
val i = null
println(i as Int)
println(i as Int?)
println(i as? Int)
```

19. Declare the **i** variable to make the first line throw an exception and the second one print `null`.

```
_________________________
println(i as Int?)
println(i as? Int)
```

20. In the following Kotlin code `td` - is a) function declaration; b) function invocation; c) special built-in syntactic construct?

```
tr {
    td {
        text("Product")
    }
    td {
        text("Popularity")
    }
}
```



Are you sure?

Are you really sure?

21. In the following Kotlin code `color` - is a) new variable declaration; b) argument name; c) argument value?

```
tr (color = "yellow") {
    td {
        text("Product")
    }
    td {
        text("Popularity")
    }
}
```

22. How many arguments does the function `alert` take? _______

How many lambdas are declared below? _______

```
alert(title = "Are you sure?",
      message = "Are you really sure?") {
```

```
    positiveButton("Yes") { /* process */ }
    negativeButton("No") { }
}.show()
```

23. Implement the function `apply`. It should call an argument of a functional type on the receiver, and return the receiver as a result.

```
public inline fun <T> T.apply(block: T.() -> Unit): T {

    __________________________________________________

    __________________________________________________
}
```

The example of the `apply` usage:

```
val sb = StringBuilder().apply {
    for (i in 0..9) {
        append(i)
    }
}
println(sb) // 0123456789
```