1. Что будет напечатано в результате выполнения следующего кода?  
   def map = [3:1, 1:2, 6:3]   
   s = map.values().sum()   
   print map.subMap([s])
2. Укажите все правильные способы использования closure:

* (1..10).each({ println it})
* (1..10).each { println it}
* (1..10).each ( println it)

1. Даны следующие объявления:

def a = **null**

def b = 'something'

Каким будет значение выражения: a.equals(b)

* true
* false
* Возникнет ошибка компиляции
* Возникнет NullPointerException во время выполнения
* null

1. Что напечатает следующий скрипт:  
   def a = [1, 2] as Set; a += [1, 2]  
   def b = [1, 2] as Set; b << [1, 2]  
   print a.size()  
   println b.size()

* 22
* 32
* Код содержит синтаксическую ошибку
* 33
* 23

1. Какие из равенств выполняются, если x == [1,1,1] и y == new HashSet(x).toList()?

* y == x.unique()
* x.size() == 3\*y.size()
* x == y
* y == [1]

1. Что будет выведено в консоль?   
   def a = [ a: "String", b: 5]   
   println a.class.name

* java.util.LinkedHashMap
* java.lang.String
* java.util.TreeMap
* Получим NullPointerException

7. Каков результат выполнения следующего кода?

def doubler = {x -> **return** 2\*x}

**if** (doubler(4) == doubler.call(4)) {

print ([1,2,3].collect(doubler)) }

* Ничего не напечатается
* Напечатается [1,2,3]
* Напечатается [2,4,6]
* Напечатается [2,2,3]

8. Что можно сказать о соотношении между   
 x.findAll{null != it} и x.grep{it}, если x == [1, null, 1]?

* Их размеры отличаются
* Они равны между собой
* Их размеры равны между собой
* Они не сравнимы между собой

9. Какой тип будет в результате вычисления выражения 2\*\*33 ?

* Double
* Integer
* Long
* BigInteger

10. Что напечатает следующий скрипт:  
1: def l = "1234"   
2: if (l ==~ "1.\*4")   
3: println l

* 14
* 1234
* В строке 2 содержится ошибка
* Ничего из вышеперечисленного

11. Что произойдет при компиляции и запуске следующего кода?

String condition = **new** Boolean("faLse").toString();

**if** (condition as Boolean) {

System.out.print("true");

} **else** {

System.out.print("false");

}

* true
* false
* Ошибка компиляции
* Ошибка выполнения

12. Что напечатает следующий фрагмент кода:

p = 3

**switch** (p) {

**default**:

print "default"

**case** 1:

print "1"

**break**;

**case** 2:

print p

**break**;

}

* 1
* 2
* default
* Ничего из вышеперечисленного

13. Каким будет результат выражения: '0' ? ([] ? 'a' : 'b') : ([:] ? 'c' : 'd')

* a
* b
* c
* d
* Выражение некорректно составлено

14. Дан следующий фрагмент кода:

def pat = "(ba)+"

def mat = ~pat

def s = "baca"

**switch**(s) {

// программный код

}

Укажите все подходящие варианты конструкции case.

* case new java.util.regex.Pattern(pat, 0) : break;
* case "baca" : break;
* case mat : break;
* case ~pat : break;

15. Каким будет результат выполнения скрипта:

def a = '1'

def d = a as double

print d

* Возникнет исключение ClassCastException
* 1
* 49
* Возникнет исключение GroovyCastException
* 1.0
* 49.0

16. Дан фрагмент Java кода:

String msg = "Hello ";

msg += firstName == **null** ? "World" : firstName;

Как правильно переписать его на Groovy?

* msg = "Hello "   
  msg = firstName == **null** ? "World" : firstName
* msg = "Hello "   
  msg += firstName ? "World" : firstName
* msg = "Hello "   
  msg += firstName ? "World" : **null**
* msg = "Hello "   
  msg += firstName ?: "World"
* Ни один из вариантов не является правильным

17. Что будет выведено на консоль при выполнении следующего скрипта

**class** A {

**int var**

**void** setVar(**int** var) {

**this**.**var** = 1;

}

}

**def** A = **new** A()

A.var = 123

println A.var

A.setVar(123)  
println A.var

* 123
* 1
* 0
* ошибка компиляции или при выполнении скрипта

18. Какие элементы будут храниться в списке table после выполнения следующего кода:

**def** table =[‘a’,’b’,’c’,’d’,’e’]

**def** b =[‘f’, ’g’, ’h’]

table[1..2]=b

* [‘a’,‘f’, ’g’, ’h’,’d’,’e’]
* [‘a’,‘f’, ’g’,‘d’,’e’]
* [‘a’,‘f’, ’g’, ’h’, ‘c’,’d’,’e’]
* ошибка компиляции
* ничего из перечисленного выше

19. Что будет выведено в консоль, если   
 println a=**this**.**class**.methods.name.grep(~**/print.\*/**).size()

* 6
* 2
* ошибка компиляции
* 1
* 5

20. Что будет результатом такого сравнения: map.a?.x==map.a?.x?.c?   
если def map=[a:[b:[c:1]]]

* true
* false

21. Напишите результат вычисления выражения :   
 **def** i = [1, 2, 3, 4, 5].inject(0) **{** x, it **->** x += it **}**

println i

RESULT = 15

22. Каким будет результат выполнения скрипта:

def a = '1.0'

double d = a

print d

* Возникнет исключение ClassCastException
* 1
* 49
* Возникнет исключение GroovyCastException
* 1.0
* 49.0

23. Каким будет результат выполнения скрипта:

def a = '1.0'

def d = a as double

print d

* Возникнет исключение ClassCastException
* 1
* 49
* Возникнет исключение GroovyCastException
* 1.0
* 49.0

24. Каков результат выполнения следующего кода?

print (0..<10).size()

* Напечатает 10
* Напечатает 9
* Напечатает число меньшее 10
* Возникнет NullPointerException

25. Каков результат выполнения следующего кода?

**def** range=(0..<10)

println range.size()

* Напечатает 10
* Напечатает 9
* Напечатает число меньшее 10
* Возникнет NullPointerException

26. Что будет напечатано в результате выполнения следующего скрипта:

**class** MyBean {

String name

def getName(){

**return** "Michael"

}

def setName(a){

println "Hi"

**return** **this**.**name** = a

}

}

me = **new** MyBean()

println me.name

me.name = "John"

println me.@name

* null  
  John
* null  
  Michael
* Michael  
  John
* Michael  
  Hi  
  John
* Michael  
  Hi  
  Michael

27. Что будет напечатано в результате выполнения следующего скрипта:

**class** MyBean {

String name

def getName(){

**return** "Michael"

}

def setName(a){

println "Hi"

**return** **this**.name = a

}

}

me = **new** MyBean()

println me.name

me.name = "John"

println me.name

* null  
  John
* null  
  Michael
* Michael  
  John
* Michael  
  Hi  
  John
* Michael  
  Hi  
  Michael

28. Имеется код. Что будет результатом его выполнения?

**class** Vendor {

String name, product

Vendor (name, product) {

**this**.name = name

**this**.product = product

}

}

String vendorRepr(Vendor v) {

print v.name + ' ' + v.product

}

**println** vendorRepr(['Canoo', 'ULC'] as Vendor) == vendorRepr(**new** Vendor('Canoo', 'ULC'))

* возникнет ошибка во время выполнения кода
* Canoo ULC true
* false
* Canoo ULC Canoo ULC true

29. Каков результат выполнения следующего кода?

def map = [a:1, b:2, c:3]

**if** (map['c'] - map.b) {

print map.get('a', 0) + map.get('d', 4)

}

* Возникнет ошибка компиляции в первой строке
* Возникнет ошибка компиляции во второй строке
* Напечатает 4
* Напечатает 5

30. Каков результат выполнения следующего кода?

store = ''

1.downto(-2) { num ->

store += num + ' '

}

print store

* Напечатает -2
* Напечатает 1 0 -1
* Напечатает 1 0 -1 -2
* Напечатает 0

31. Как можно создать экземпляр следующего класса:

**public** **class** MyBean{

**int** a

String b

**boolean** c

**static** **void** main(args) {

def var = **new** MyBean(a:10);

println var.a

println var.getA()

}

}

Выберите все подходящие варианты.

* def v = new MyBean(a:10'', c:false)
* def v = new MyBean(10, , true)
* def v = new MyBean(c:true, b:'', a:5);
* def parMap = [a:10, b:'', c:false]  
  def v = new MyBean(parMap);
* def var5 = new MyBean();
* def parMap = [10, '', false]  
  def var4 = new MyBean(parMap);

32. Какие из высказываний справедливы по отношению к приведенному коду:

matcher = 'a:1 b:2 c:3' =~ /(\S+):(\S+)/

**if** (matcher.hasGroup()) {

print matcher[0]

}

* На консоль ничего не выведется
* Напечатает строку a:1
* Возникнет ошибка компиляции в строке №3
* Напечатает строку [a:1, a, 1]

33. Существует следующий код:

**class** SampleClass {

def value = 1

def getOtherValue() { value \* 2 }

}

obj = **new** SampleClass()

Укажите все корректные способы обращения к полю value.

* obj.properties['value']
* obj.value
* obj['value']
* obj.properties.value