Тесты

Категория - Java Core

Уровень сложности - Минимальный

№1 Укажите допустимый синтаксис комментария.

1. /* Комментарий */
2. # Комментарий
3. /* Комментарий
4. // Комментарий

№2 Какие типы данных не существуют в Java?

1. int
2. float
3. string
4. unknown
5. Double

№3 Можно ли создать программу (приложение) на Java, не используя среду разработки (IDE)?

1. Да
2. Нет, так как необходимо скомпилировать исходный код в байт-код

№4 Какое расширение имеют файлы с исходным кодом Java?

1. .javac
2. .java
3. .class
4. .classpath

№5 Может ли файл содержать более одного класса Java?

1. Да, но только если один внешний класс имеет модификатор доступа public
2. Да, если все внешние классы будут иметь модификатор доступа private

3. Нет

№6 Какое имя переменной является синтаксически недопустимым?

1. 53someVariable
2. _someVariable
3. some-variable
4. somevariable
5. someVariable
6. Somevariable53
7. _53someVariable

№7 От какого класса наследуются все создаваемые классы в Java?

1. Классы наследуется от типа указанного после ключевого слова extends, если тип не указан, значит класс не является наследником
2. Object
3. Class

№8 Импорт какого пакета в Java происходит автоматически?

1. Все пакеты нужно явно указывать
2. java.util
3. java.lang
4. java.text

№9 Укажите какой тип данных, из представленных, занимает наибольшее место в памяти.

1. int
2. byte
3. double
4. float
5. никакой, так как все примитивные типы занимают одинаковый объём в памяти

№10 Какой арифметический оператор в Java не существует?

1. --

2. %

3. **

4. ++

№11 Какого оператора сравнения в Java не существует

1. !=

2. ==

3. <>

4. ===

5. >=

№12 Какой тип преобразования не произойдёт автоматически?

1. byte в short

2. int в long

3. char в short

4. long в float

№13 Какой тип данных не поддерживает оператор switch?

1. int

2. String

3. char

4. long

№14 Какое выражение создания переменной и массива является синтаксически некорректным?

1. int array[] = new int[0];

2. int[] array = new int[1];

3. int array = new int[1];

4. int[] array = new int[] {1, 2};

5. int[] array = {1, 2};

№15 Выбирите верное утверждение. Добавление ключевого слова final к полю класса означает, что

1. поле класса может быть инициализировано только один раз

2. поле должно инициализироваться при объявлении, в конструкторе или в инициализаторе экземпляра

3. поле по умолчанию становится закрытым (аналогично использованию модификатора доступа private)

№16 Что не влияет на перегрузку метода?

1. Количество параметров
2. Тип возвращаемого значения
3. Модификаторы доступа
4. Типы параметров

№17 Значение ключевого слова void?

1. Метод не может переопределяться в классе наследнике
2. Метод ничего не возвращает
3. Метод не может перегружаться

№18 Назначение ключевого слова extends?

1. Используется, чтобы указать класс, интерфейс, от которого происходит наследование
2. Добавляется к методу класса, указывая, что метод может наследоваться
3. Используется в обобщениях (generics) для наложение ограничений типа

№19 Какое ключевое слово не относится к модификаторам доступа?

1. public
2. package
3. private
4. protected

№20 Как вызвать конструктор класса?

```
class SomeClass {
}
```

1. construct()

2. constructor()

3. SomeClass()

4. someClass()

№21 Все методы интерфейса по умолчанию являются

1. public и abstract

2. protected и abstract

3. public и super

№22 Какие типы не относятся к примитивным?

1. int

2. Byte

3. char

4. short

5. Object

6. String

№23 Выбирите только целочисленные типы.

1. int

2. String

3. double

4. char

5. short

6. long

№24 Можно ли создать свой примитивный тип данных?

1. Нет

2. Да

№25 Укажите отличия переменных примитивных и ссылочных типов.

1. Переменные примитивных типов не могут являться полями в классах, в отличии от ссылочных.

2. Переменные примитивных типов хранят значение, а переменные ссылочных типов

ссылку на объект.

3. Значение переменных примитивных типов нельзя вернуть из метода, а ссылочных можно.

№26 Какое значение по умолчанию будет присвоено для поля counter?

```
public class Some {
    private Short counter;
}
```

1. 0
2. undefined
3. null
4. Произойдет ошибка компиляции, так как private поля нужно обязательно инициализировать.

№27 Что будет выведено в результате выполнения кода?

```
public class Main {
    public static void main(String[] args) {
        Some some1 = new Some();
        Some some2 = some1;
        System.out.println(some1 == some1);
    }
}
```

1. false
2. true

№28 Какой результат будет выведен в результатет выполнения кода?

```
public class Main {
    public static void main(String[] args) {
        int number = 1;
        boolean isSome = (boolean) number;
```

```
        System.out.println(isSome);
    }
}
```

1. 1
2. true
3. Код не скомпилируется, так как существует ошибка привидения типа int к boolean.
4. null

№29 Что будет выведно, в результате выполнения кода?

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        int b = 2;

        if ((a++ > 0) | (b++ > 0)) {
            b += 4;
        }

        System.out.println(b);
    }
}
```

1. 7
2. 6
3. 2

№30 Какой из типов относится к беззнаковым?

1. int
2. char
3. long
4. double

№31 Какая запись вызовет ошибку во время компиляции?

1. int a = 12;
2. int a = 014;
3. int a = 1_2;
4. int a = 0b1100;
5. Все записи корректны, ошибки не будет.

№32 Какое значение будет выведено, при выполнении кода?

```
public class Main {
    public static void main(String[] args) {
        int a = 10;
        some(a++);
    }

    public static void some(int a) {
        System.out.println(a);
    }
}
```

1. 11
2. 12
3. 10

№33 Какое значение будет выведено, при выполнении кода?

```
public class Main {
    public static void main(String[] args) {
        byte a = 127;
        System.out.println(++a);
    }
}
```

1. 128
2. -128
3. 127

4. Произойдёт ошибка, так как тип byte не может хранить число более 127.

№34 Какое значение будет выведено, при выполнении кода?

```
public class Main {
   public static void main(String[] args) {
      int a = 1;
      int b = 2;
      System.out.println(a | b);
   }
}
```

1. 2
2. false
3. 3
4. true

№35 Возможны ли арифметические операции с типом char?

1. Да.
2. Нет.

№36 Что будет храниться в переменной ch в результате выполнения char ch = 'a' + 1 ?

1. a1
2. 2
3. Произойдёт ошибка, так как к символу нельзя прибавить число.
4. b

№37 Какая кодировка используется для хранения символа в памяти?

1. Windows-1250
2. UTF-8
3. UTF-16
4. ASCII

№38 Что будет выведено в результате выполнения кода?

```
public class Main {
    public static void main(String[] args) {
        double a = 0.1 * 10;
        double b = 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1 + 0.1;
        System.out.println(a == b);
    }
}
```

1. true
2. false

№39 Что нужно сделать, если требуется хранить целочисленное число, которое не помещается в тип long?

1. Использовать класс Math.
2. Использовать класс BigInteger.
3. К сожалению java не позволяет хранить числа больших размеров, и при выборе языка это нужно учитывать. Поэтому нужно использовать другой язык программирования.

№40 Скомпилируется ли код?

```
public void someFunc() {
    char ch = 'f';
    short s = ch;
}
```

1. Да.
2. Нет.

№41 Скомпилируется ли код?

```
public void someFunc() {
    short s = 'f';
}
```

1. Да.

2. Нет.

№42 Скомпилируется ли код?

Integer a = 20;

1. Нет, так как число 20 не является объектом, и относится к примитивному типу.

2. Да, так как произойдёт автоупаковка числа в объект типа Integer.

№43 Скомпилируется ли код?

```
public static void main(String[] args) {
    Integer a = 23;
    Integer b = 23;
    int c = a + b;
}
```

1. Нет, так как операция сложения неприминима к ссылочным типам.

2. Нет, так как результат сложения должен быть присвоен переменной ссылочного типа Integer.

3. Да.

№44 Укажите верный способ преобразования строки в число?

1. Integer.parseInt("34");

2. (Integer) "34";

3. "34" + 0;

№45 Укажите допустимый вариант преобразования числа в строку?

1. (String) 45;

2. Такое невозможно.

3. Integer.toString(45);

4. 45 + "";

№46 Что произойдёт, если в классе прописать String text;?

```
public class SomeClass {
    String text;
}
```

1. Создаётся объект типа String, ссылка на который будет хранится в переменной text.
2. Создаётся переменная text типа String, под которую выделяется память.
3. Ничего не произойдёт, пока не будет создан объект с помощью ключевого слова new.

№47 Назначение оператора new?

1. Создание нового объекта.
2. Объявление новой переменной.
3. Объявление нового класса.

№48 Как вызвать деструктор класса?

1. Вызвав метод finalize().
2. Такой возможности нет.
3. Вызвав метод destroy().

№49 Как и когда происходит вызов сборщика мусора?

1. Программист, когда считает нужным, может запустить сборщик мусора.
2. Сборщик мусора запускает JVM (Java Virtual Machine), при наличии определённых условий.
3. Сборщиком мусора управляет операционная система, и запускает, когда свободной памяти останется не более 10%.
4. У современных компьютеров всегда достаточно памяти, и необходимости в сборщике нет. Сборщик является пережитком прошлого.

№50 Каким образом можно изменить размер массива?

1. Размер автоматически увеличится, если явно указать индекс элемента. Например так array[3] = 4, для массива new int[2].
2. Нужно использовать специальное свойство length, которому присвоить значение новой длины массива. Например так array.length = 3.
3. Изменить размер нельзя.

№51 Какое значение будет выведено?

```
public static void main(String[] args) {
    int[] array = new int[10];
    System.out.println(array[2]);
}
```

1. 0
2. 2
3. 1
4. Программа не скомпилируется, так как массив не инициализирован.

№52 Как получить значение первого элемента массива?

1. array[1]
2. array[0]
3. array[]

№53 Что произойдёт, если обратиться к несуществующему элементу массива (индекс выходит за пределы длины массива), например int[5]?

1. Вернётся значение равное 0.
2. Программа не скомпилируется.
3. Произойдёт исключение ArrayIndexOutOfBoundsException.

№54 Сколько битов в памяти занимает тип short?

1. 32
2. 8
3. 16
4. 64

№55 Что обозначает понятие Varargs?

1. Означает, что одним из параметров метода является массив.
2. Означает, что метод может принимать переменное количество параметров, обозначается с помощью троеточия, например int... args.
3. Означает, что метод принимает более двух параметров.

№56 Как правильно сравнить два массива по содержанию?

1. boolean result = (array1 == array2);
2. boolean result = array1.equals(array2);
3. boolean result = Arrays.equals(array1, array2);

№57 Как правильно преобразовать массив в строку (вместе со всем содержанием)?

1. String str = Arrays.toString(array);
2. String str = array.toString();
3. System.out.println(array);

№58 Каким может быть условное выражение в операторе if (выражение)?

1. Только выражение, возвращающее значение boolean.
2. Выражение, возвращающее boolean, int, long.
3. Любое выражение.

№59 Обязательно ли в конструкции if ... else, наличие оператора else?

1. Да.
2. Нет.

№60 Что будет выведено в результате выполнения кода?

```
public static void main(String[] args) {
    int a = 10;
    System.out.println((a == 10) ? 10 : 0);
}
```

1. 0
2. Код не скомпилируется, так как содержится синтаксическая ошибка.
3. 10
4. true

№61 Что будет выведено в результате выполнения кода?

```
public static void main(String[] args) {
    int a = 10;
    switch (a) {
        case 10: System.out.println(10);
        case 1: System.out.println(1); break;
        case 2: System.out.println(2);
    }
}
```

1. 10
2. Ошибка компиляции, так как отсутствует оператор break, в каждой секции case.
3. 10
1

№62 Что будет выведено в результате выполнения кода?

```
public static void main(String[] args) {
    int a = 10;
    switch (a) {
        case 1: System.out.println(1);
        case 2: System.out.println(2); break;
        case 10: System.out.println(10);
    }
}
```

1. 1
2
2. 1
2
10
3. 10
4. Ошибка компиляции, так как отсутствует оператор break, в каждой секции case.

№63 Для чего используется ключевое слово while?

1. Указывает, что метод не возвращает значения.

2. Такого слова в Java нет.

3. Создание цикла.

№64 Является ли следующая запись синтаксически корректной?

for (;;) {

}

1. Да.

2. Нет.

№65 Назначение оператора continue?

1. Прерывает текущую итерацию в цикле, и передаёт управление коду, следующему после цикла.

2. Прерывает текущую итерацию в цикле, и начинает следующую.

3. Возвращает значение из метода.

№66 Как вернуть из метода значение 5?

1. return 5;

2. continue 5;

3. break 5;

4. Все варианты верные.

№67 Что произойдёт, если в void методе указан оператор return, без возвращаемого значения?

1. Код не скомпилируется, так как метод ничего не возвращает, и не может содержать оператор return.

2. Из метода вернётся значение null.

3. Код после return не будет выполнен, и произойдёт немедленный выход из метода.

№68 Скомпилируется ли код?

do {

} while (true)

1. Да.

2. Нет.

№69 Назначение ключевого слова package?

1. Используется для установления принадлежности класса пакету.

2. Используется в качестве модификатора доступа класса, поля, метода и так далее.

3. Такого ключевого слова не существует.

№70 Что означает понятие "пакет по умолчанию"?

1. Это понятие относится к пакету java.lang, который можно явно не импортировать.

2. Означает, что перед определением класса отсутствует директива package с именем пакета, которому класс принадлежит.

3. Означает, что при отсутствии модификатора доступа к классу, методу и так далее, модификтор доступа по умолчанию является package-private.

№71 Назначение ключевого слова import?

1. Используется для импортирования полей и методов из класса родителя при наследовании.

2. Используется для реализации интерфейсов.

3. Используется для указания полного имени класса из другого пакета, чтобы впоследствии ссылаться на импортированные классы по их короктим именам.

Правильные ответы

Вопросы	Ответы
1	1, 4
2	3, 4
3	1
4	2
5	1
6	1, 3
7	2
8	3
9	3
10	3
11	3, 4
12	3
13	4
14	3
15	1, 2
16	2, 3
17	2
18	1, 3
19	2
20	3
21	1
22	2, 5, 6
23	1, 5, 6
24	1
25	2
26	3
27	2
28	3
29	1
30	2

31	5
32	3
33	2
34	3
35	1
36	4
37	3
38	2
39	2
40	2
41	1
42	2
43	3
44	1
45	3, 4
46	2
47	1
48	2
49	2
50	3
51	1
52	2
53	3
54	3
55	2
56	3
57	1
58	1
59	2
60	3
61	3
62	3
63	3

64	1
65	2
66	1
67	3
68	2
69	1
70	2
71	3