Olga Smolyakova

**Java Basics 2**

Questionnaires

1. Укажите имя файла, в которых могут размещаться следующие классы. Объясните свой выбор.
2. **public** **class** X{} X\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Имя класса должно совпадать с именем файла\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **class** Y{} Y\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

У не public классов имя класса может не совпадать с именем файла \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Впишите в класс MyClass метод main и объявите, что данный класс лежит в пакете by.epam.tr.check

package by.epam.tr.check

**public** **class** MyClass{

public static void main(String[]args){

}

}

1. Напишите недостающий код в пример, решающий задачу нахождения наименьшего из двух чисел.

**int** x;

**int** y;

**int** min;

Scanner sc = new Scanner();

System.***out***.print("Введите x: ");

x = sc.nextInt();

System.***out***.print("Введите y: ");

y = sc.nextInt();

if (x<y) {

min = x} else if(x>y) {

min = y} else {

System.out.println(“Values are equal”)}

Comment: не помню формат параметра передачи в сканнер, но что-то с System\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

System.***out***.println("min = " + min);

1. Напишите компилируемый класс, выводящий на консоль числа от 1 до 10.

public class Counter {

public static void main(String[] args) {

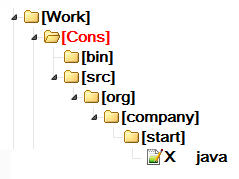
for(int i=1; i<=10; i++) {

System.out.println(i);

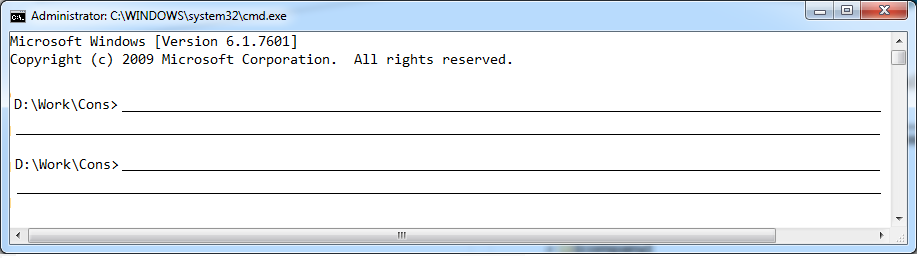
}

}

}

1. Дан public класс X, лежащий в пакете org.company.start

Запишите команды компиляции и запуска этого класса из-под командной строки.



Javac org.company.start.X.class

Java X

1. Исправьте ошибки в коде так, чтобы программа могла быть скомпилирована и запущена.

**public** **class** Calc {

**public** **void** main(String[] args) {

**double** x = 3.4;

**double** y = 5.6;

**double** z = 7.8;

**double** result;

result = *func*(x, y, z);

System.***out***.println("result = " + result);

}

**public** **static** **double** func(**double** a, **double** b, **double** c){

**return** a+b+c;

}

}

1. Запишите результат выполнения кода, данного ниже. Объясните результат.

String str1 = "string1";

String str2 = **new** String("string1");

String str3 = "string1";

**if**(str1 == str2){

System.***out***.print("a");

}**else**{

System.***out***.print("b");

}

**if**(str1 == str3){

System.***out***.print("c");

}**else**{

System.***out***.print("d");

}

Результат:

a

c

Объяснение: Одинаковые строки складываются в одну область памяти, соответсвенно ссылка на них будет одна

1. В классе Calc дан метод extent. Напишите для этого метода junit-тест, проверяющий его работу при корректных параметрах.

**public** **class** Calc {

**public** **static** **double** extent(**double** x, **double** y){

**double** result = 1;

**if** (y == 0){

**return** 1;

}

**for**(**int** i=0; i<Math.*abs*(y); i++){

result = result\*x;

}

**if**(y<0){

result = 1 / result;

}

**return** result;

}

}

@Test

public void test() {

double x = 5;

double y = 1;’

doubleExpectedResult = 5;

assertEquals(extent(x, y), doubleExpectedResult);

}

1. Дан код.

**package** pack1;

**public** **class** Calc {

**public** **static** **void** main(String[] args) {

**for** (**int** i = 0; i < args.length; i++) {

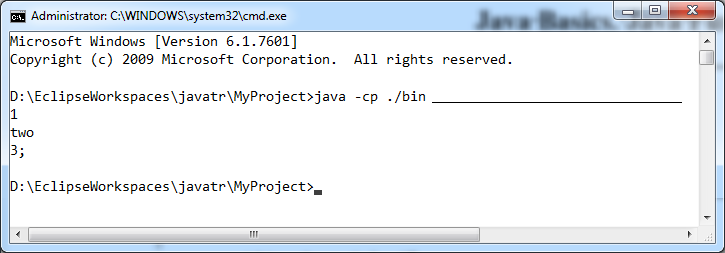
System.***out***.println(args[i]);

}

}

}

Напишите команду запуска этого класса из-под консоли так, чтобы получился следующий результат.



javac Calc.java

java Calc 1 two 3

1. Какой параметр нужно передать в метод valueOf, чтобы при запуске на консоль вывелось сообщение “blue”?

**public** **class** Main {

**public** **static** **void** main(String[] args) {

Color c;

c = Color.*valueOf*(\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_);

**switch**(c){

**case** ***RED***:

System.***out***.println("red");

**break**;

**case** ***GREEN***:

System.***out***.println("green");

**break**;

**case** ***BLUE***:

System.***out***.println("blue");

**break**;

**case** ***YELLOW***:

System.***out***.println("yellow");

**break**;

}

}

}

BLUE

1. Допиште код, позволяющй найти сумму элементов в массиве.

**public** **class** Main {

**public** **static** **void** main(String[] args) {

**int**[] mas = **new** **int**[]{23,54,12,76,45,98,12,56};

**int** sum;

for (int i; i < mas.length; i++) {

sum+=mas[i];

}

System.***out***.println("sum = " + sum);

}

}

1. В чем разница при передаче параметров для методов *x* и *y*? Дайте развернутое объяснение.

**public** **class** Main {

**public** **void** x(**int** x){}

**public** **void** y(Date d){}

}

Для метода x передаётся примитивная переменная, т.е. передаётся непосредственно копия переменной и если внутри метода будут произведены действия, то изначальная переменная не изменится и нужно делать return, если необходимо.

В методе y передаётся ссылочная переменная. Это значит, что передаётся не переменная, а ссылка на неё и соответственно, если буду произведены действия с этой ссылкой, то это приведёт к изменению первоначальной переменной.

1. Напишите код, создающий и инициализирующий массив следущего вида.



int [][] twoArray = {{0, 0}, {0. 0, 0}, {0, 0 }, {0. 0, 0, 0, 0}};

1. Создайте класс так, чтобы его объекты можно было создать приведенными ниже способами:

Book b1 = **new** Book("Java", "Blinov", 2015);

Book b2 = **new** Book("Java", "Blinov");

Book b3 = **new** Book(20000.0);

public class Book {

private String programmingLanguage;

private String author;

private int year;

private int price;

public Book(String programmingLanguage, String author, String price) {

this.programmingLanguage = programmingLanguage;

this.author = author;

this.price = price;

}

public Book(String programmingLanguage, String author) {

this.programmingLanguage = programmingLanguage;

this.author = author;

}

public Book(String price) {

this.price = price;

}

}

1. Дан класс.

**public** **class** Main {

}

Создайте объект этого класса и объясните свое решение.

Main main = new Main();

Конструкторы не объявлены, соответственно польхуемся default конструктором.

1. Перечислите модификаторы довтупа в java и укажите их характеристики.

public – глобальный доступ;

protected – доступ в рамках пакета и для наследников из другого пакета;

default – доступ в рамках класса и пакета;

private – доступ в рамках класса;

1. В чем разница при передаче параметров для методов *x* и *y*? Дайте развернутое объяснение.

**public** **class** Main {

**public** **void** x(**int** x){}

**public** **void** y(Date d){}

}

Для метода x передаётся примитивная переменная, т.е. передаётся непосредственно экземпляр переменной и если внустри метода будут произведены действия, то изначальная переменная не изменится и нужно делать return, если необходимо.

В методе y передаётся ссылочная переменная. Это значит, что передаётся не переменная, а ссылка на неё и соответственно, если буду произведены действия с этой ссылкой, то это приведёт к изменению первоначальной переменной.

1. Дан класс. Исправьте в нем ошибки так, чтобы код класса мог быть скомпилирован. Объясните свои исправления.

**public** **class** Note {

**private** String note;

**private** Data data;

**public** Note(){

}

**public** Note(String note, Data data){

**this**.note = note;

**this**.data = data;

}

}

class Data{

}

В теле default конструктора были непонятные значения, я их удалил. Получился правильно объявленный default контсруктор.

1. Дайте определение понятиям класс и объект.

Класс – это сущность в которой описаны свойства и состояния будущего объекта. Как чертёж будущего объекта.

Объект – это сущность которая создаётся на основании класса.

1. Реализуйте для класса A шаблон Singleton. Объясните, почему при применении шаблона Singlton можно создать только единственный объект класса A.

**public** **class** A {

private static A a;

private A(){}

privet static A getA() {

if (a == null) {

a = new A();

}

retutn a;

}

}

1. Дан класс Print. Перегрузите в классе Print методы так, чтобы был доступен вызов следующих методов через объект класса Print.

Print p = **new** Print();

p.print(234);

p.print("string");

p.print(456, "string");

**public** **class** Print {

public void print(int value) {

System.out.println(value);

}

public void print(String text) {

System.out.println(text);

}

public void print(int value, String text) {

System.out.println(value);

System.out.println(text);

}

}

1. Переопределите для класса Main методы equals() и hashCode().

**public** **class** Main {

**private** **int** x;

**private** **String** y;

public int getX() {

return x;

}

public String getY() {

return y;

}

@Override

public boolean equals(Object obj){

if(obj == this) {

return true;

}

if(obj == null || obj.getClass() != this.getClass()) {

return false;

}

Main mainObj = (Main)obj;

return x == mainObj.getX() &&

(y == mainObj.getY() || (y != null && y.equals(mainObj.getY())));

}

@Override

Public int hashCode() {

final int prime = 31;

int result = y == null ? 0 : y.hashCode();

result = prime \* result + x;

return result;

}

}

1. Даны классы А и В. Напишите в классе В такой конструктор, чтобы объект класса В можно было создать следующим образом

A a = **new** B(123, "bstub", 'd');.

Объясните свое решение.

**(возможно несколько решений, приведите одно)**

**class** A{

A(**int** x){ **this**("stub"); }

A(String str){ **super**(); }

}

**class** B **extends** A{

B(int x, String str, char ch){ super(“sss”); }

}

**Комментарий: может я неправильно понял задание, но тут уже объявлен конструктор.**

1. Даны два класса.

**class** A{}

**class** B **extends** A{}

Объясните, что “делают” следующие два оператора.

|  |  |
| --- | --- |
| 1) A a = **new** B();  Переменной суперкласса А присвается ссылка на объект B. | 2) B b = **new** A();  Переменной клааса B присваевается ссылка суперкласса А.  Здесь будет ошибка. |

1. Дан код.

|  |  |
| --- | --- |
| **public** **class** V {  **public** **static** **void** main(){  A a = **new** B();  a.f();//5  }  }  **class** A{  **public** **void** f(){}  }  **class** B **extends** A{  **public** **void** f(){}  } | Метод какого класса вызовется в строке 5 и почему?  **Вызовется метод класса B, потому что здесь действует динамическая диспетчеризация методов, т.е. вызывается метод, который есть в суперклассе, но в реализации подкласса.** |

1. Дан код

|  |  |
| --- | --- |
| **public** **class** V {  **public** **static** String m(String n) {  **try** {  **if** (n.length() > 0) {  System.***out***.println(n);  }  System.***out***.println(n + "2");  } **catch** (NullPointerException e) {  e.printStackTrace();  }  **return** n;  }  } | Метод m вызывается как V.*m*("1");.  Объясните, что будет в этом случае происходить при выполнении кода и почему.  **На печать выведется n, так как условие выполнится.**  **Потом выведется n + “2” после блока if.**  **Блок catch будет пропущен, так как нет исключения.** |

1. Дан код.

**public** **static** **void** m() {

**try** {

System.***out***.println("a");

**return**;

} **finally**{

System.***out***.println("b");

}

}

Что выведется на консоль при вызове этого метода и почему?

**На консоль выведется “a”, “b” так как блок finally выполняется всегда.**

1. Дана иерархия классов

**abstract** **class** A{}

**class** B **extends** A{}

**class** C **extends** A{}

**interface** X{}

**class** D **extends** B **implements** X{}

**class** E **extends** C **implements** X{}

Укажите, какие из операторов являются верными, а какие нет. Объясните причины своего решения.

A a = **new** A();\_Неверный. Нельзя создать экземпляр абстрактного класса.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

B b = **new** A();Неверный. Нельзя создать экземпляр абстрактного класса.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

D d = **new** B();Неверный. Нельзя создать ссылку на экземпляр суперкласса.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

X x1 = **new** X(); Неверный. Нельзя создать экземпляр интерфейса.

X x2 = **new** B(); Неверный. Переменная интерфейса ссылается на экземпляо класса, который не реализует интерфейс.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

C c = **new** B(); Неверный. Переменная класса С не может ссылать на экземпляр класса B.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

E e = **new** D(); Неверный. Переменная класса E не может ссылать на экземпляр класса D.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

X x3 = **new** E();Верный. Переменная интерфейса ссылается на экземпляр класса, который реализует интерфейс.

X x4 = **new** D(); Верный. Переменная интерфейса ссылается на экземпляр класса, который реализует интерфейс.

x3 = x4;\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Дан интерфейс X, лежащий в пакете xxx. Реализуйте его для класса Y, лежащего в пакете yyy.

**package** xxx;

**public** **interface** X {

**void** a();

**double** b(**double** c);

}

Import xxx.X

public class Y {

@Override

void a() {

System.out.println(a);

}

@Override

double b(double c) {

System.out.println();

}

}

1. Дан класс Student.

**public** **class** Student {

**private** String name;

**private** String surname;

**private** **int** idFaculty;

}

1. Перепишите класс, реализуя для него интерфейс Comparable, сортирующий студентов по имени.

**public** **class** Student implements Comparable<Student>{

private String name;

private String surname;

private int idFaculty;

public String getName() {

return name;

}

public int compareTo(Strudent s) {

return name.compareTo(s.getName);

}

}

1. Напишите для объектов класса Student компаратор, сортирующий студентов по имени и фамилии

public class StudentNameComparator implements Comparator<Student>{

public int nameComparator(Student a, Student b) {

return a.getName().compare(b.getName());

}

}

public class StudentSurnameComparator implements Comparator<Student>{

public int surnameComparator(Student a, Student b) {

return a.getSurname().compare(b.getSurname());

}

}

При вызове используем следующую строчку

Comparator<Student> comparator = new StudentNameComparator().thenComparing(new StudentSurnameComparator());