

1. Укажите имя файла, в которых могут размещаться следующие классы. Объясните свой выбор.
2. **public** **class** X{} в файле, где лежит код приложения (например, by.epam.task1)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

1. **class** Y{} в файле, где лежит код приложения\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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.in);

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

x = sc.nextInt();

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

y = sc.nextInt();

if (y>x){

min = x;}

else{

min = y;

}

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

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

public class Number {

public static void main(String[] args){

int x;

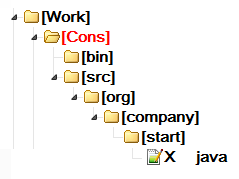
for (x=1; x<11;x++){

System.*out*.println(x);

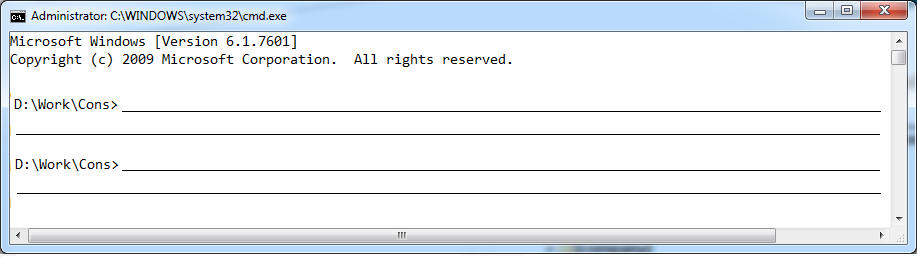
}

}

}

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

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



javac start\X.java

java start.X

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

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

**public** **static** **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: \_\_\_\_\_bc\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Объяснение: \_\_\_\_\_b - ссылки ведут на разные объекты (хранятся в разных местах), с - одинаковые строки с одинаковыми адресами в памяти (обе ссылки ведут в одну и ту же область памяти)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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

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

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

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

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;

}

}



1. Дан код.

**package** pack1;

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

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

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

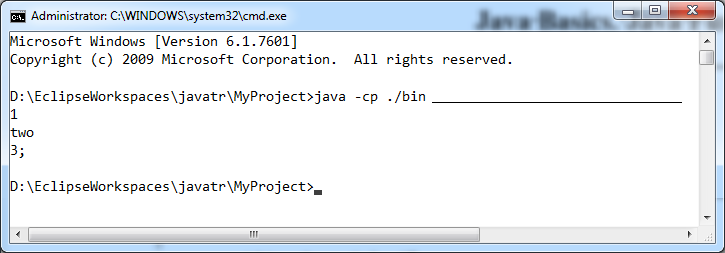
System.***out***.println(args[i]);

}

}

}

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



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*(\_\_\_\_\_\_\_BLUE\_\_\_\_\_\_\_\_\_\_\_\_\_\_);

**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**;

}

}

}

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=0;

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

sum = 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){}

}

В первом случае передается целое число (то есть передача по значению), а во втором - ссылка на объект.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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



int [][] arr = new int [4][5];

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 name;

private String author;

private int yearOfPublish;

private double price;

public Book (String name, String author; int yearOfPublish){

this.name = name;

this.author = author;

this.yearOfPublish = yearOfPublish

}

public Book (String name, String author){

this.name = name;

this.author = author;

}

public Book (double price){

this.price = price;

}

1. Дан класс.

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

}

Создайте объект этого класса и объясните свое решение.

int x;

Если я правильно понимаю, Main это системный класс и создавая объекты мы неявно к нему обращаемся.

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

public - общедоступный, может использоваться другими классами из текущего и внешних пакетов

private - закрытый, доступен только для кода внутри того же класса

protected - доступен внутри класса или в производных классах

отсутствие модификатора - по умолчанию, доступен для всех классов в текущем пакете

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){}

}

В первом случае передается целое число (то есть передача по значению), а во втором - ссылка на объект.

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

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

**private** String note;

**public** Data data;

**public** Note(){

("", **null**);

}

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

**this**.note = note;

**this**.data = data;

}

}

class Data{

}

1. Дайте определение понятиям класс и объект.

Класс - паттерн для создания объектов, набор методов и свойств. Объект - экземпляр класса, имеющий конкретные свойства.

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

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

private 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 someInt);

public void print (String someString);

public void print (int someInt; String SomeString);

}

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

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

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

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

@override

public boolean equals(Object obj){

//some code

}

@override

public int hashCode(){

//some code

}

}

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”); }

}

1. Даны два класса.

**class** A{}

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

Объясните, что “делают” следующие два оператора.

| 1) A a = **new** B();  создается объект класса В | 2) B b = **new** A(); |
| --- | --- |

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?  метод класса В, потому что он переопределяется внутри класса В |
| --- | --- |

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. |
| --- | --- |

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?

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();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

B b = **new** A();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

D d = **new** B();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

X x1 = **new** X();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

X x2 = **new** B();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

C c = **new** B();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

E e = **new** D();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

X x3 = **new** E();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

X x4 = **new** D();\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

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

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);

}

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>{

}

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