Pytania na interview

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Lp | | | Typ | Pytanie | | | Klucz | | | Waga | Ocena  (1-6) | Komentarz | | |
| 1 | | |  | Czy kod w bloku finally zawsze się wykona? | | | * System.exit(1) | | |  |  |  | | |
| 2 | | |  | Jaka jest różnica między klasą abstrakcyjną a interfejsem? | | | An abstract class can have instance methods that implement a default behavior. An Interface can only declare constants and instance methods, but cannot implement default behavior and all methods are implicitly abstract. An interface has all public members and no implementation. An abstract class is a class which may have the usual flavors of class members (private, protected, etc.), but has some abstract methods. | | |  |  |  | | |
| 3 | | |  | Czy metoda main może być prywatna? | | | "Main method not public." | | |  |  |  | | |
| 4 | |  | | Co się stanie jeśli usuniemy „static” z sygnatury metody main | | NoSuchMethodError | | |  | |  |  | |
| 5 | |  | | Organizacja pamięci w javie | |  | | |  | |  |  | |
| 6 | |  | | Jaka jest różnica między checked i unchecked exceptions? | | * A checked exception is some subclass of Exception (or Exception itself), excluding class RuntimeException and its subclasses. Making an exception checked forces client programmers to deal with the possibility that the exception will be thrown. eg, IOException thrown by java.io.FileInputStream's read() method· Unchecked exceptions are RuntimeException and any of its subclasses. Class Error and its subclasses also are unchecked. With an unchecked exception, however, the compiler doesn't force client programmers either to catch the exception or declare it in a throws clause. In fact, client programmers may not even know that the exception could be thrown. eg, StringIndexOutOfBoundsException thrown by String's charAt() method· Checked exceptions must be caught at compile time. Runtime exceptions do not need to be. Errors often cannot be. | | |  | |  |  | |
| 7 |  | | | Rożnica między klasą Vector a ArrayList | * Vector - synchronizowany | | |  | | |  |  |
| 8 |  | | | W jaki sposób są przekazywane typy proste jako parametry metod? | Przez wartość | | |  | | |  |  |
| 9 |  | | | Czy obiekty sa przekazywane przez wartość czy referencję? | * Przez wartość referencji | | |  | | |  |  |
| 10 |  | | | Czym jest serializacja? | * Serialization is a mechanism by which you can save the state of an object by converting it to a byte stream. | | |  | | |  |  |
| 11 |  | | | Jeżeli serializujemy obiekt, co się stanie jeśli obiekt zawiera referencje do innych obiektów? | The serialization mechanism generates an object graph for serialization. Thus it determines whether the included object references are serializable or not. This is a recursive process. Thus when an object is serialized, all the included objects are also serialized alongwith the original obect. | | |  | | |  |  |
| 12 |  | | | Co się dzieje ze statycznymi polami przy serializacji? | There are three exceptions in which serialization doesnot necessarily read and write to the stream. These are 1. Serialization ignores static fields, because they are not part of ay particular state state. 2. Base class fields are only hendled if the base class itself is serializable. 3. Transient fields | | |  | | |  |  |
| 13 |  | | | Kiedy obiekt może zostać usunięty przez garbage collection? | - brak aktwnych referencji | | |  | | |  |  |
| 14 | W | | | Czy po bloku try zawsze musi być blok catch | It is not necessary that each try block must be followed by a catch block. It should be followed by either a catch block OR a finally block. And whatever exceptions are likely to be thrown should be declared in the throws clause of the method. | | |  | | |  |  |
| 15 |  | | | Jeżeli w bloku try użyjemy instrukcji return – czy blok finally się wykona? | Wykona się | | |  | | |  |  |
| 16 |  | | | Jakie są alternatywy do dziedziczenia? | Delegation is an alternative to inheritance. Delegation means that you include an instance of another class as an instance variable, and forward messages to the instance. It is often safer than inheritance because it forces you to think about each message you forward, because the instance is of a known class, rather than a new class, and because it doesn't force you to accept all the methods of the super class: you can provide only the methods that really make sense. On the other hand, it makes you write more code, and it is harder to re-use (because it is not a subclass). | | |  | | |  |  |
| 17 |  | | | Czy plik .java może zawierać więcej niż jeedną klasę? |  | | |  | | |  |  |
| 18 |  | | | Jakie są domyślne wartości dla zmiennych lokalnych | The local variables are not initialized to any default value, neither primitives nor object references. If you try to use these variables without initializing them explicitly, the java compiler will not compile the code. It will complain abt the local varaible not being initilized.. | | |  | | |  |  |
| 19 |  | | | Co wypisze się na ekran: **System.out.println ("1" + 3);** | 13 | | |  | | |  |  |
| 20 |  | | | Czy w javie jest mozliwe dziedziczenie z wielu klas? | Nie jest | | |  | | |  |  |
| 21 |  | | | Jak zaimplementować relacje one-to-one, one-to-many podczas projektowania tabelek | One-to-One relationship can be implemented as a single table and rarely as two tables with primary and foreign key relationships. One-to-Many relationships are implemented by splitting the data into two tables with primary key and foreign key relationships. Many-to-Many relationships are implemented using a junction table with the keys from both the tables forming the composite primary key of the junction table. | | |  | | |  |  |
| 22 |  | | | Co to jest transakcja i jakie ma właściwości? | A transaction is a logical unit of work in which, all the steps must be performed or none. ACID stands for Atomicity, Consistency, Isolation, Durability. These are the properties of a transaction. For more information and explanation of these properties, see SQL Server books online or any RDBMS fundamentals text book. | | |  | | |  |  |
| 23 |  | | | Rodzaje złączeń tabelek? | Self, LEFT , Right, | | |  | | |  |  |
|  |  | | |  |  | | |  | | |  |  |

# Code example A

Class C {

public static void main(String[] args) {

int[]a1[]=new int[3][3]; //3

int a2[4]={3,4,5,6}; //4

int a2[5]; //5

}}

# Code example B

interface I{

void f1(); // 1

public void f2(); // 2

protected void f3(); // 3

private void f4(); // 4

}

# Code example C

class C{

static int s;

public static void main(String a[]){

C obj=new C();

obj.m1();

System.out.println(s);

}

void m1();

{ int x=1;

m2(x);

System.out.println(x+"");

}

void m2(int x){

x=x\*2;

s=x;

}}

# Code example D

class C {

public static void main(String[] args) {

int i1=1;

switch(i1){

case 1:

System.out.println("one");

case 2:

System.out.println("two");

case 3:

System.out.println("three");

}}}

# Code example E

class C1

{

static interface I

{

static class C2

{

}

}

public static void main(String a[])

{

C1.I.C2 ob1=new C1.I.C2();

System.out.println("object created");

}

}