|  |  |
| --- | --- |
| **Question 1 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <vector>**

**#include <list>**

**#include <algorithm>**

**#include <iostream>**

**using namespace std;**

**void main(){**

**vector<int> v1;**

**int dim=20;**

**int x;**

**v1.reserve(dim);**

**for (int i=0; i < dim; i++) {**

**x = (i+1)\*10;**

**v1.push\_back(x);**

**}**

**list<int> l;**

**list<int>::iterator itl;**

**for(int i=0; i<10;i++)**

**l.push\_back(v1[i]);**

**for(int i=10; i<dim; i++)**

**l.insert(l.begin(), v1[i]);**

**for (itl=l.begin(); itl != l.end(); itl++)**

**cout<<(\*itl)<<endl;**

**}**

**Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. A compile-time error is generated; |  |
| |  | | --- | |  | | B. A run-time error is generated; |  |
| |  | | --- | |  | | C. The application displays 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200; |  |
| |  | | --- | |  | | D. The application displays 200, 190, 180, 170, 160, 150, 140, 130, 120, 110, 10, 20, 30, 40, 50, 60, 70, 80, 90, 100; |  |
| |  | | --- | |  | | E. The application displays 200, 190, 180, 170, 160, 150, 140, 130, 120, 110, 100, 90, 80, 70, 60, 50, 40, 30, 20, 10; |  |

|  |  |
| --- | --- |
| **Question 2 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <iostream>**

**#include <string.h>**

**using namespace std;**

**class Person {**

**public:**

**int age;**

**char\* name;**

**Person(int v=0, char\* n="Name"):age(v){**

**this->name = new char[strlen(n)+1];**

**strcpy(this->name,n);**

**}**

**Person(Person& p){**

**this->age = p.age;**

**this->name = new char[strlen(p.name)+1];**

**strcpy(this->name, p.name);**

**}**

**~Person(){**

**delete [] this->name;**

**}**

**void operator=(Person& p){**

**this->age = p.age;**

**delete[] this->name;**

**this->name = new char[strlen(p.name)+1];**

**strcpy(this->name, p.name);**

**}**

**Person operator+(Person p, int v){**

**Person t;**

**t.age = this->age+p.age+v;**

**return t;**

**}**

**};**

**Person Medie(Person a, Person b){**

**Person p;**

**p.age=(a.age+b.age)/2;**

**return p;**

**}**

**void main(){**

**Person p1, p2(20, "John");**

**Person p3;**

**p3 = p1+p2+10;**

**}**

**Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. A run-time error is generated because of the operator = ; |  |
| |  | | --- | |  | | B. A compile-time error is generated because of number of parameters of the operator + overloaded method; |  |
| |  | | --- | |  | | C. A compile-time error is generated because the operator + overloaded method is not called properly; |  |
| |  | | --- | |  | | D. A run-time error is generated because of the copy constructor; |  |
| |  | | --- | |  | | E. The application runs properly and the age attribute of the object p3 is modified; |  |

|  |  |
| --- | --- |
| **Question 3 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <iostream>**

**using namespace std;**

**class Polygon{**

**public:**

**virtual void f(){cout<<"Polygon class"<<endl;}**

**};**

**class Rectangle:public Polygon{**

**public:**

**void f(){cout<<"Rectangle class"<<endl;}**

**};**

**class Triangle:public Polygon{**

**public:**

**virtual void f(){cout<<"Triangle class"<<endl;}**

**};**

**void main(){**

**Polygon p, \*pp;**

**Rectangle r, \*pr;**

**Triangle t, \*pt;**

**pp=&p;**

**pr=&r;**

**pt=&t;**

**p=r;**

**p.f();**

**pp=pr;**

**pp->f();**

**pp=pt;**

**pp->f();**

**}**

**Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. The program displays the strings: Polygon class, Polygon class, Triangle class; |  |
| |  | | --- | |  | | B. A compile-time error is generated because the conversion derived object to base object is not allowed; |  |
| |  | | --- | |  | | C. The program displays the strings: Polygon class, Rectangle class, Triangle class; |  |
| |  | | --- | |  | | D. A run-time error is generated because the methods f are not declared as virtual in all defined classes; |  |
| |  | | --- | |  | | E. The program displays the strings: Polygon class, Polygon class, Polygon class; |  |

|  |  |
| --- | --- |
| **Question 4 of 25** | 2.0 Points |

**In ASN1 V3 certificate structure, the OpenSSL function X509\_set\_version setting the X509 certificate version:**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Writes value 2 in an INTEGER ASN1 Data Type; |  |
| |  | | --- | |  | | B. Writes value 3 in a SEQUENCE ASN1 Data Type; |  |
| |  | | --- | |  | | C. Writes value 3 in an INTEGER ASN1 Data Type; |  |
| |  | | --- | |  | | D. Writes value 2 in the first byte of certificate; |  |
| |  | | --- | |  | | E. Writes value 3 in an OBJECT IDENTIFIER ASN1 Data Type; |  |

See source code of **ex37\_GenerateX509.cpp**

|  |  |
| --- | --- |
| **Question 5 of 25** | 2.0 Points |

**It considers the following OpenSSL function call RSA\_private\_encrypt(sizeof(finalDigest), buf, e\_data, apriv, RSA\_PKCS1\_PADDING). The private key has a length of 1024 bits. Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. RSA\_PKCS1\_PADDING has the same length with the private key; |  |
| |  | | --- | |  | | B. sizeof(finalDigest) represents the digest of the message; |  |
| |  | | --- | |  | | C. A compile-time error is generated because the function has other header; |  |
| |  | | --- | |  | | D. RSA\_PKCS1\_PADDING is a parameter specifying the padding type; |  |
| |  | | --- | |  | | E. RSA\_PKCS1\_PADDING is a parameter specifying the padding length; |  |

|  |  |
| --- | --- |
| **Question 6 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <stdio.h>**

**#define NMAX 100**

**#define then**

**#define BEGIN {**

**#define END }**

**#define INTEGER int**

**void main()**

**BEGIN**

**INTEGER S = 0;**

**INTEGER vector[NMAX];**

**for(INTEGER i=0; i<NMAX; i++)**

**vector[i]=i+1;**

**for(INTEGER i=0; i<NMAX; i++)**

**S+=vector[i];**

**printf("S= %d",S);**

**END**

**At run-time, the result of the above application is:**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. 50; |  |
| |  | | --- | |  | | B. 0; |  |
| |  | | --- | |  | | C. 5050; |  |
| |  | | --- | |  | | D. The application generates a run-time error; |  |
| |  | | --- | |  | | E. The source code cannot be compiled; |  |

|  |  |
| --- | --- |
| **Question 7 of 25** | 2.0 Points |

**In C++, a static attribute declared in a class is:**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Always initialized in class definition; |  |
| |  | | --- | |  | | B. A data definition when it is declared; |  |
| |  | | --- | |  | | C. Used by class to manage the object collection; |  |
| |  | | --- | |  | | D. Always defined as private attribute; |  |
| |  | | --- | |  | | E. A member for each object having the class as data type; |  |

|  |  |
| --- | --- |
| **Question 8 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <iostream>**

**#include <string.h>**

**using namespace std;**

**class Car{**

**private:**

**int prodDate;**

**char\* color;**

**public:**

**Car(){**

**prodDate=20120704;**

**color=0;**

**}**

**Car(int an = 0, char\* cul = ""){**

**this->prodDate = an;**

**this->color = new char[strlen(cul)+1];**

**strcpy(this->color, cul);**

**}**

**~Car(){**

**delete[] this->color;**

**}**

**};**

**void main(){**

**Car carA, carB;**

**}**

**Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. The destructor method is defined in a wrong way; |  |
| |  | | --- | |  | | B. A compile-time error is generated; |  |
| |  | | --- | |  | | C. The attributes are defined in a wrong way; |  |
| |  | | --- | |  | | D. The default constructor method is called 2 times; |  |
| |  | | --- | |  | | E. A run-time error is generated; |  |

Compile error: Class “Car” has more than one default constructor

The **constructor with default values** **parameters** is considered as **default constructor** but there is already a default constructor, so: compile error.

|  |  |
| --- | --- |
| **Question 9 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <iostream>**

**#include <string.h>**

**using namespace std;**

**class Car{**

**private:**

**int prodDate;**

**char\* color;**

**public:**

**Car(){**

**prodDate=20120704;**

**color=0;**

**}**

**Car(int an, char\* cul){**

**this->prodDate = an;**

**this->color = new char[strlen(cul)+1];**

**strcpy(this->color, cul);**

**}**

**~Car(){**

**delete[] this->color;**

**}**

**};**

**void main(){**

**Car carA, carB(20120615, "Red");**

**Car carC = carB;**

**}**

**Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. A run-time error is generated because of destructor method; |  |
| |  | | --- | |  | | B. A compile-time error is generated because of object assignment; |  |
| |  | | --- | |  | | C. A compile-time error is generated because of object defining; |  |
| |  | | --- | |  | | D. A run-time error is generated because of default constructor method; |  |
| |  | | --- | |  | | E. The application runs properly; |  |

|  |  |
| --- | --- |
| **Question 10 of 25** | 2.0 Points |

**It considers the following source code:**

**#include <iostream>**

**#include <string.h>**

**using namespace std;**

**class Person {**

**public:**

**int age;**

**char\* name;**

**Person(int v=0, char\* n="Name"):age(v){**

**this->name = new char[strlen(n)+1];**

**strcpy(this->name,n);**

**// cout << "constructor with parameters" << endl;**

**}**

**Person(Person& p){**

**this->age = p.age;**

**this->name = new char[strlen(p.name)+1];**

**strcpy(this->name, p.name);**

**// cout << "copy constructor" << endl;**

**}**

**~Person(){**

**delete [] this->name;**

**// cout << "destructor" << endl;**

**}**

**void operator=(Person& p){**

**this->age = p.age;**

**delete[] this->name;**

**this->name = new char[strlen(p.name)+1];**

**strcpy(this->name, p.name);**

**// cout << "assign"<<endl;**

**}**

**};**

**Person Medie(Person a, Person b){ // copy constructor is called twice**

**Person p; // default constructor is called**

**p.age=(a.age+b.age)/2;**

**return p; // copy constructor is called again**

**}**

**void main(){**

**Person p1, p2(20, "John"); // default constructor is called for “p1”**

**// but because it is not defined,**

**// instead is called the constructor**

**// with default values parameters**

**Person p3; // same as for “p1”**

**p3 = Medie(p1, p2); // assign (operator =) is called**

**}**

**Which of the following statements is CORRECT?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Operator = is called 2 times; |  |
| |  | | --- | |  | | B. A run-time error is generated because of destructor method; |  |
| |  | | --- | |  | | C. Copy constructor method is called 3 times; |  |
| |  | | --- | |  | | D. A compile-time error is generated because of ambiguity of constructor methods; |  |
| |  | | --- | |  | | E. Copy constructor is called 2 times; |  |

constructor with parameters

constructor with parameters

constructor with parameters

copy constructor

copy constructor

constructor with parameters

copy constructor

destructor

destructor

destructor

assign

destructor

destructor

destructor

destructor

|  |  |
| --- | --- |
| **Question 11 of 25** | 2.0 Points |

**Given**

**class Base{**

**static { System.out.print("b1 ");} // static init block – executed once**

**{ System.out.print("b2 ");} // instance init block – executed for every object created**

**public Base(){**

**System.out.print("b3 ");**

**}**

**}**

**class Subclass extends Base{**

**public Subclass(){**

**System.out.print("s1 ");**

**}**

**}**

**public class Main {**

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

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

**new Base();**

**new Subclass();**

**}**

**}**

**what is the result ?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. main b1 s1 b2 b3 |  |
| |  | | --- | |  | | B. main b2 b3 s1 b1 b2 b3 |  |
| |  | | --- | |  | | C. main b1 b3 s1 b2 b3 |  |
| |  | | --- | |  | | D. main b1 b2 b3 s1 b2 b3 |  |
| |  | | --- | |  | | E. main b1 b2 b3 b2 b3 s1 |  |
| |  | | --- | |  | | F. It can't be determined |  |
| |  | | --- | |  | | G. b1 main b2 b3 s1 b2 b3 |  |

|  |  |
| --- | --- |
| **Question 12 of 25** | 2.0 Points |

**Given:**

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

**double[] frequency = {10.5, 3.45, 15.99, 3.99};**

**int[] values = {5, 8, 11, 29};**

**try {**

**DataOutputStream out = new DataOutputStream(new BufferedOutputStream(new FileOutputStream("test.txt")));**

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

**out.writeDouble(frequency[i]);**

**out.writeInt(values[i]);**

**}**

**out.flush();**

**DataInputStream in = new DataInputStream(new BufferedInputStream(new FileInputStream("test.txt")));**

**try {**

**while (true) {**

**double frecv = in.readDouble();**

**int val = in.readInt();**

**System.out.format("Value %d - %.2f frequency ", val, frecv);**

**}**

**} catch (EOFException e) {**

**//aici iese cand s-a terminat fisierul**

**}**

**} catch (IOException ioe) {**

**ioe.printStackTrace();**

**}**

**}**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Value 5 - 10,50 frequency Value 8 - 3,45 frequency Value 11 - 15,99 frequency Value 29 - 3,99 frequency |  |
| |  | | --- | |  | | B. Value 1076166656 - 0,00 frequency Value 1074502041 - -0,00 frequency Value 1076886241 - 19991590273549737000000000000000000000,00 frequency Value 1074785157 - 0,00 frequency |  |
| |  | | --- | |  | | C. the example generates compiler errors |  |
| |  | | --- | |  | | D. the example does't print something because the text file is empty |  |
| |  | | --- | |  | | E. the example generates runtime exceptions |  |

// The constructor DataOutputStream(BufferedOutputStream) is undefined

|  |  |
| --- | --- |
| **Question 13 of 25** | 2.0 Points |

**Being given the next class**

**public class Box {**

**Object value; // the value is typw Object so it will allow int and String values**

**public Box(Object value){**

**this.value = value;**

**}**

**public Object getValue(){**

**return this.value;**

**}**

**public void setValue(Object value){**

**this.value = value;**

**}**

**}**

**and the test**

**Box[] numericalValues = new Box[3];**

**numericalValues[0] = new Box(10);**

**numericalValues[1] = new Box(20);**

**numericalValues[2] = new Box("30"); // here is the hidden bug**

**int sum = 0;**

**for(Box box : numericalValues)**

**sum += (int)box.getValue(); // for the 3rd object getValue will return a String**

**// that cannot be added to sum**

**System.out.println("The sum is:"+sum);**

**select the correct affirmation:**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. The test will run without problems and it will print "The sum is 60" |  |
| |  | | --- | |  | | B. The test will generate a run-time exception on line sum += (int)box.getValue(); |  |
| |  | | --- | |  | | C. The test will generate a compiler error on line numericalValues[0] = new Box(10); |  |
| |  | | --- | |  | | D. The test will generate a compiler error on line sum += (int)box.getValue(); |  |
| |  | | --- | |  | | E. The test will generate a compiler error on line numericalValues[2] = new Box("30"); |  |

|  |  |
| --- | --- |
| **Question 14 of 25** | 2.0 Points |

**For the next sequence**

**class Foo{**

**Integer code;**

**String description;**

**int[] values;**

**public Foo(int code, String description, int no){**

**this.code = code;**

**this.description = description;**

**for(int i = 0; i<no; i++)**

**values[i] = i+1;**

**}**

**}**

**public class Test {**

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

**Foo[] foos = new Foo[5]; // it is created an array of 5 objects of Foo type**

**System.out.println("Done !");**

**// System.out.println(foos.length); // <- THIS WILL PRINT HOW MANY OBJECTS**

**}**

**}**

**How many objects are created before printing Done!**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. 5 |  |
| |  | | --- | |  | | B. 0 |  |
| |  | | --- | |  | | C. 1 |  |
| |  | | --- | |  | | D. 15 |  |
| |  | | --- | |  | | E. 6 |  |
| |  | | --- | |  | | F. 20 |  |
| |  | | --- | |  | | G. 16 |  |
| |  | | --- | |  | | H. 15 |  |
| |  | | --- | |  | | I. 21 |  |

|  |  |
| --- | --- |
| **Question 15 of 25** | 2.0 Points |

**class Parent{**

**public Parent(String s){**

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

**}**

**}**

**public class Child extends Parent{**

**public Child(String s){System.out.print("C");}**

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

**new Child("2");**

**}**

**}**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Compilation error | Implicit super constructor Parent() is undefined. Must explicitly invoke another constructor |
| |  | | --- | |  | | B. PC |  |
| |  | | --- | |  | | C. CP2 |  |
| |  | | --- | |  | | D. 2 |  |
| |  | | --- | |  | | E. PC2 |  |
| |  | | --- | |  | | F. CP |  |

|  |  |
| --- | --- |
| **Question 16 of 25** | 2.0 Points |

**In the next code sequence**

**public class Test {**

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

**doSomething(1);**

**doSomething(1,2);**

**}**

**//insert here method definition**

**}**

**which of the following code blocks can be inserted independently without compile errors (Choose all that apply)**

|  |  |
| --- | --- |
| A. static void doSomething(int[] args){ }// nu merge pt ca The method doSomething(int[]) in the type Base is not applicable for the arguments (int) |  |
| B. static void doSomething(int... args){ }(corecta) |  |
| C. static void doSomething(int... args, int a){ }// nu merge pt ca The variable argument type int of the method doSomething must be the last parameter |  |
| D. static void doSomething(int args…){ }// nu merge pt ca Syntax error on token "Invalid Character", delete this token |  |
| E. static void doSomething(int a, int... args){ }(corecta) |  |

|  |  |
| --- | --- |
| **Question 17 of 25** | 2.0 Points |

**Which one from the next Java statements it CORRECT:**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. abstract classes can be instantiated in objects; (slide 146) |  |
| |  | | --- | |  | | B. abstract classes can contain non-abstract methods and instance variables; |  |
| |  | | --- | |  | | C. All these statements are NOT correct in Java |  |
| |  | | --- | |  | | D. interfaces can NOT be used as reference type; (slide 150) |  |
| |  | | --- | |  | | E. interfaces can contain instance variables (slide 150) |  |

|  |  |
| --- | --- |
| **Question 18 of 25** | 2.0 Points |

**In the next sequence**

**How many objects are eligible for GC when line //other is reached?**

**class Something{**

**Integer value = 200;**

**Something doSomething(Something s){**

**s = null;**

**return s; // this returns null**

**}**

**public static void main(){**

**Something s1 = new Something();**

**Something s2 = new Something();**

**Something s3 = s1.doSomething(s2); // it will get null from the return**

**s1 = null; // this is another one to get to garbage collector**

**//other**

**}**

**}**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Impossible to determine |  |
| |  | | --- | |  | | B. 2 |  |
| |  | | --- | |  | | C. 1 |  |
| |  | | --- | |  | | D. 3 |  |
| |  | | --- | |  | | E. 0 |  |

class Something{

Integer value = 200;

Something doSomething(Something s){

s = null;

return s; // this returns null

}

}

public class Main {

public static void main(String[] args) {

Something s1 = new Something();

Something s2 = new Something();

Something s3 = s1.doSomething(s2); // it will get null from the return

s1 = null; // this is another one to get to garbage collector

System.out.println("s1:"); test(s1);

System.out.println("s2:"); test(s2);

System.out.println("s3:"); test(s3);

}

static void test(Something a) {

try { System.out.println(a); }

catch(Exception e) { System.out.println((String)null); }

}

}

|  |  |
| --- | --- |
| **Question 19 of 25** | 2.0 Points |

**For the next code sequence**

**2:**

**3: class Student implements Person {**

**4: public void Speak(){}**

**5: }**

**6:**

**7: abstract class MasterStudent extends Student {}**

**8:**

**9: abstract class PhDStudent extends Student {**

**10: public void Speak(String message){}**

**11: }**

**12:**

**13: class Graduate extends Student implements Person{**

**14: public void Speak(){}**

**15: }**

**what is the result ?**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Compilation succeeds |  |
| |  | | --- | |  | | B. Compilation error at line 9 | The hierarchy of the type PhDStudent is inconsistent |
| |  | | --- | |  | | C. Compilation error at line 7 | The hierarchy of the type MasterStudent is inconsistent |
| |  | | --- | |  | | D. Compilation error at line 3 | The type Person cannot be a superinterface of Student; a superinterface must be an interface |
| |  | | --- | |  | | E. Compilation error at line 13 | Graduate:  The hierarchy of the type Graduate is inconsistent  Person:  The type Person cannot be a superinterface of Graduate; a superinterface must be an interface |

|  |  |
| --- | --- |
| **Question 20 of 25** | 2.0 Points |

**The next sequence**

**String name1 = "John";**

**String name2 = "John";**

**if(name1 == name2)**

**System.out.println("Strings are equal");**

**else**

**System.out.println("Strings are NOT equal");**

**String name3 = new String("John");**

**// this will create another object with different reference**

**if(name1 == name3)**

**System.out.println("Strings are equal");**

**else**

**System.out.println("Strings are NOT equal");**

**Integer i1 = 10;**

**Integer i2 = 10;**

**if(i1 == i2)**

**// Integer values ranging from -128 to 127 are stored by the JVM in an area similar to String constant pool because they are used in many situations.**

**System.out.println("Integers are equal");**

**else**

**System.out.println("Integers are NOT equal");**

**Integer i3 = 300;**

**Integer i4 = 300;**

**if(i3 == i4)**

**// because the integers are bigger than 127 (see above) then the references are different**

**System.out.println("Integers are equal");**

**else**

**System.out.println("Integers are NOT equal");**

**will print:**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A. Strings are equal   Strings are NOT equal   Integers are equal   Integers are equal |  |
| |  | | --- | |  | | B. Strings are equal   Strings are equal   Integers are equal   Integers are equal |  |
| |  | | --- | |  | | C. Strings are NOT equal   Strings are NOT equal   Integers are equal   Integers are NOT equal |  |
| |  | | --- | |  | | D. Strings are equal   Strings are equal   Integers are equal   Integers are NOT equal |  |
| |  | | --- | |  | | E. Strings are equal   Strings are NOT equal   Integers are equal   Integers are NOT equal |  |
| |  | | --- | |  | | F. Strings are NOT equal   Strings are equal   Integers are equal   Integers are NOT equal |  |

|  |  |
| --- | --- |
| **Question 21 of 25** | 2.0 Points |

**The next Java example**

**class Base{**

**public int[] valori1;**

**public Base(){**

**System.out.println("DBC call");**

**valori1 = new int[5];**

**}**

**public Base(int n){**

**valori1 = new int[n];**

**System.out.println("BC call");**

**}**

**}**

**class Subclass extends Base{**

**public int[] valori2;**

**public Subclass(int n){**

**valori2 = new int[n];**

**System.out.println("SC call");**

**}**

**}**

**public class Main {**

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

**Base b = new Base(5); // first call Base(5) constructor with params**

**Subclass d = new Subclass(6); // call Subclass(6) will:**

**// - call first Base() default constructor**

**// - call Subclass(6) constructor with params**

**}**

**}**

**prints**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | A.  BC call - SC call - DBC call |  |
| |  | | --- | |  | | B.  BC call - DBC call - SC call |  |
| |  | | --- | |  | | C.  BC call - BC call - SC call |  |
| |  | | --- | |  | | D.  BC call - SC call |  |
| |  | | --- | |  | | E.  DBC calln - BC call - SC call |  |

|  |  |
| --- | --- |
| **Question 22 of 25** | 2.0 Points |

**The next class framework:**

**class Vehicle{**

**protected String model;**

**protected int cc;**

**protected Vehicle(String M, int CC) { model = M; cc = CC; }**

**}**

**class Auto extends Vehicle{**

**String series;**

**public Auto(){**

**super("Model",0);**

**series = "0";**

**}**

**public Auto(String M, int CC, String S){**

**model = M; cc = CC; series = S;**

**}**

**}**

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | |  | | ~~A. generates compiler errors because the Auto class is accessing inherited fields which are protected~~ |  |
| |  | | --- | |  | | ~~B. generates compiler errors because the base class constructor is called with super("Model",0);~~ |  |
| |  | | --- | |  | | C. generates compiler errors because the inheritance is implemented using implements and NOT extends |  |
| |  | | --- | |  | | ~~D. the framework is correct defined, without compiler errors~~ |  |
| |  | | --- | |  | | E. generates compiler errors because the constructor with arguments from Auto is calling the default constructor from Vehicle |  |

Compiler message:

Implicit super constructor Main.Vehicle() is undefined. Must explicitly invoke another constructor