**Complete Core Java Basic Questions Part - 2**

**Contents:**

  Non static

  Object and reference variable (Total available questions : 29)

  this key word (Total available questions : 3)

  constructors (Total available questions : 20)

  constructor overloading and this() (Total available questions : 23)

  IIB (Total available questions : 10)

  objects count (Total available questions : 9)

  Object Oriented

  Encapsulation (Total available questions : 5)

  Inheritance (Total available questions : 30)

  super() calling statement (Total available questions : 3)

  packages (Total available questions : 12)

  access levels (Total available questions : 88)

  abstrcat classes (Total available questions : 18)

  interfaces (Total available questions : 26)

  method overloading (Total available questions : 18)

  method overriding (Total available questions : 12)

  auto widening (Total available questions : 10)

  explicit narrowing (Total available questions : 15)

  auto upcasting (Total available questions : 8)

  explicit down casting (Total available questions : 31)

  achieving polymorphism (Total available questions : 9)

  Final key word

  final key word (Total available questions : 57)

  classpath

  classpath (Total available questions : 24)

  Important commands

  Javadoc, javap and jar commands (Total available questions : 4)

  Scanner class

  Scanner class (Total available questions : 8)

  Eclipse

  Eclipse

Top of Form

|  |  |
| --- | --- |
| **1.** | Defined method should not have an abstract key word.Q72 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  No | | |
| **2.** | method without a body should not have an abstract key word.Q73 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **3.** | method should have both definition (body) and also an abstract key wordQ74 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **4.** | any class can be declared as an abstractQ75 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **5.** | if class containing minimum one abstract method, then it should be declared as an abstractQ76 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **6.** | abstract class can become a member of java file.Q77 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **7.** | for abstract class also class file generating while compiling.Q78 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **8.** | we can create an object to an abstract classQ79 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |  |
| --- | --- | --- |
| **9.** | we can create a reference to an abstract classQ80 | |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |
| **10.** | | we can use abstract class for the data type purpose.Q81 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |

|  |  |
| --- | --- |
| **11.** | attribute can be an abstract class type.Q82 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **12.** | method argument cant be an abstract class type.Q83 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **13.** | if super class abstract method inheriting to subclass and subclass not implemented. thenQ84 |
| |  | | --- | | A.  subclass should declare as an abstract | | |
| **14.** | is it possible to declare a class as an abstract if that class doesnt have any abstract methods.Q85 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **15.** | Is it possible to declare a constructor as an abstract?Q86 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **16.** | constructors are inheriting to the subclass.Q87 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **17.** | is it possible to define a constructor in the abstract class?Q88 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **18.** | When constructor of the abstract class executingQ89 |
| |  | | --- | | A.  while object creating to an abstract class |  |  | | --- | | B.  while object creating to subclass which is concrete | | |

|  |  |
| --- | --- |
| **19.** | by default interface is an abstract.Q99 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **20.** | we can develop defined methods in an interfaceQ100 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **21.** | interface attributes should initialize while declaring itselfQ101 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **22.** | abstract key word is an optional while declaring a method in an interface.Q102 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **23.** | which access specifier allowed for an interface membersQ103 |
| |  | | --- | | A.  private |  |  | | --- | | B.  protected |  |  | | --- | | C.  public | | |
| **24.** | which is the default access specifier for the members of an interfaceQ104 |
| |  | | --- | | A.  private |  |  | | --- | | B.  protected |  |  | | --- | | C.  public | | |

|  |  |
| --- | --- |
| **25.** | by default interface is a public.Q105 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **26.** | we can develop constructor inside an interfaceQ106 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **27.** | interface attributes areQ107 |
| |  | | --- | | A.  private |  |  | | --- | | B.  static | | |
| **28.** | We cant create an object to an interfaceQ108 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **29.** | We cant create a reference to an interfaceQ109 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **30.** | we can use interface for the data type purpose.Q110 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **31.** | we can access attributes of an interface through interface name.Q111 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **32.** | interface cab become a member of a java file.Q112 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **33.** | while compiling java file, class file not creating to an interfaceQ113 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **34.** | which key word is used to develop a class by making use of super interfaceQ114 |
| |  | | --- | | A.  extends |  |  | | --- | | B.  implements | | |

|  |  |
| --- | --- |
| **35.** | which key word is used to develop an interface by making use of super interfaceQ115 |
| |  | | --- | | A.  extends |  |  | | --- | | B.  implements | | |
| **36.** | is it possible to implement interface method in the subclass with protected access specifier?Q116 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **37.** | it it required to implement all methods of super interface inside a subclass inorder to create an object to subclass?Q117 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **38.** | if any method of super interface not implemented in the subclass, then which key word should be used to the subclassQ118 |
| |  | | --- | | A.  public |  |  | | --- | | B.  abstract | | |

|  |  |
| --- | --- |
| **39.** | which is the correct order?Q119 |
| |  | | --- | | A.  first extends and then implements |  |  | | --- | | B.  first implements and then extends | | |
| **40.** | class can implements any number of super interfaces.Q120 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **41.** | class can extends one super class and also can implements any number of super interfaces.Q121 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **42.** | interface canextends any number of super interfaces.Q122 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **43.** | multiple inheritance possible only through super interfacesQ123 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **44.** | multiple inheritance possible only through super classesQ124 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **45.** | overloaded methods should have same name.Q191 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **46.** | overloaded methods should have different name.Q192 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **47.** | overloaded methods should have same name and difference in the number of arguments or difference in any one argument data type.Q193 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **48.** | overloaded methods should have same return type.Q194 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **49.** | overloaded methods should have different return type.Q195 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **50.** | overloaded methods can have different return type.Q196 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **51.** | overloaded methods can have same return type.Q197 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **52.** | overloaded methods can be both static and not staic also.Q198 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **53.** | overloaded methods can be only staticQ199 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **54.** | overloaded methods can be only non staticQ200 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **55.** | overloaded methods can be only abstractQ201 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **56.** | defined methd cant be overloaded with an abstract oneQ202 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **57.** | all overloaded methods should have same access specifierQ203 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **58.** | all overloaded methods can have same access specifierQ204 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **59.** | all overloaded methods can have different access specifierQ205 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **60.** | inherited method can overload with an incorporated method in the subclassQ206 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **61.** | super class method can overload with a subclass method in the subclassQ207 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **62.** | abstract method of super class can overload in the subclassQ208 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **63.** | while overriding a method, signature should be sameQ209 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **64.** | while overriding a method, signature should differQ210 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **65.** | while overriding a method, return type should differQ211 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **66.** | while overriding a method, return type shold be same except in the derived datatype.Q212 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **67.** | what could be the return type while overring super class method in the sub class, if super class method return type is void.Q213 |
| |  | | --- | | A.  void only |  |  | | --- | | B.  any return type | | |
| **68.** | what could be the return type while overring super class method in the sub class, if super class method return type is int.Q214 |
| |  | | --- | | A.  byte or int |  |  | | --- | | B.  int only | | |

|  |  |
| --- | --- |
| **69.** | what could be the return type while overring super class method in the sub class, if super class method return type is A type. Assume B is a subclass to A.Q215 |
| |  | | --- | | A.  A |  |  | | --- | | B.  B | | |
| **70.** | what could be the return type while overring super class method in the sub class, if super class method return type is A type. Assume A is a subclass to B.Q216 |
| |  | | --- | | A.  A |  |  | | --- | | B.  B | | |

|  |  |
| --- | --- |
| **71.** | what could be the the access specifier while overring super class method in the sub class, if super class method access specifier is protected.Q217 |
| |  | | --- | | A.  private |  |  | | --- | | B.  protected |  |  | | --- | | C.  public | | |
| **72.** | what could be the the access specifier while overring super class method in the sub class, if super class method access specifier is package scope.Q218 |
| |  | | --- | | A.  private |  |  | | --- | | B.  protected |  |  | | --- | | C.  public | | |

|  |  |
| --- | --- |
| **73.** | is it possible to override static method of super class with non static in the sub class.Q219 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **74.** | which key word is used to call super class method while its got overrided in the subclassQ220 |
| |  | | --- | | A.  super |  |  | | --- | | B.  extends | | |

|  |  |
| --- | --- |
| **75.** | which data type is narrower among?Q221 |
| |  | | --- | | A.  short |  |  | | --- | | B.  float |  |  | | --- | | C.  byte | | |
| **76.** | which data type is narrower among?Q222 |
| |  | | --- | | A.  long |  |  | | --- | | B.  float |  |  | | --- | | C.  double | | |

|  |  |
| --- | --- |
| **77.** | which one is automaticQ223 |
| |  | | --- | | A.  widening |  |  | | --- | | B.  narrowing | | |
| **78.** | short assigning to int isQ224 |
| |  | | --- | | A.  widening |  |  | | --- | | B.  narrowing | | |

|  |  |
| --- | --- |
| **79.** | long assigning to double isQ225 |
| |  | | --- | | A.  widening |  |  | | --- | | B.  narrowing | | |
| **80.** | class A  {  public static void main(String[] args)  {  byte b = 100;  short s = 1000;  s = b;  System.out.println("done");  }  }  Q226 |
| |  | | --- | | A.  Compilation error |  |  | | --- | | B.  done | | |

|  |  |
| --- | --- |
| **81.** | class B  {  public static void main(String[] args)  {  byte b = 100;  short s = 1000;  s = (short) b;  System.out.println("done");  }  }  Q227 |
| |  | | --- | | A.  Compilation error |  |  | | --- | | B.  done | | |
| **82.** | class C  {  public static void main(String[] args)  {  byte b = 100;  short s = 1000;  int i;  double d;  s = b;  i = b;  i = s;  d = b;  d = s;  d = i;  System.out.println("done");  }  }  Q228 |
| |  | | --- | | A.  Compilation error |  |  | | --- | | B.  done | | |

|  |  |
| --- | --- |
| **83.** | class D  {  public static void main(String[] args)  {  System.out.println("main begin");  byte b = 20;  test(b);  System.out.println("main end");  }  public static void test(int i)  {  System.out.println("test");  }  }  Q229 |
| |  | | --- | | A.  Compilation error |  |  | | --- | | B.  done |  |  | | --- | | C.  main begin  test  main end | | |
| **84.** | class E  {  public static void main(String[] args)  {  System.out.println("main begin");  byte b = 20;  test(b);  short s = 20;  test(s);  int i = 20;  test(i);  long j = 20;  test(j);  System.out.println("main end");  }  public static void test(double i)  {  System.out.println("test");  }  }  Q230 |
| |  | | --- | | A.  Compilation error |  |  | | --- | | B.  main begin  test  test  test  test  main end |  |  | | --- | | C.  main begin  test  main end | | |

|  |  |
| --- | --- |
| **85.** | which data type is wider among?Q231 |
| |  | | --- | | A.  short |  |  | | --- | | B.  float |  |  | | --- | | C.  byte | | |
| **86.** | which data type is wider among?Q232 |
| |  | | --- | | A.  long |  |  | | --- | | B.  float |  |  | | --- | | C.  double | | |

|  |  |
| --- | --- |
| **87.** | which one an explicitQ233 |
| |  | | --- | | A.  widening |  |  | | --- | | B.  narrowing | | |
| **88.** | long assigning to int isQ234 |
| |  | | --- | | A.  widening |  |  | | --- | | B.  narrowing | | |

|  |  |
| --- | --- |
| **89.** | double assigning to byte isQ235 |
| |  | | --- | | A.  widening |  |  | | --- | | B.  narrowing | | |
| **90.** | which is widermostQ236 |
| |  | | --- | | A.  long |  |  | | --- | | B.  float |  |  | | --- | | C.  double | | |

|  |  |
| --- | --- |
| **91.** | which is narrowestQ237 |
| |  | | --- | | A.  long |  |  | | --- | | B.  byte |  |  | | --- | | C.  double | | |
| **92.** | class I  {  public static void main(String[] args)  {  int x = 1000;  byte y = 10;  y = x;  System.out.println("done");  }  }  Q238 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  done | | |

|  |  |
| --- | --- |
| **93.** | class I  {  public static void main(String[] args)  {  int x = 1000;  byte y = 10;  y = x;  System.out.println("done");  }  }  Q239 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  done | | |
| **94.** | class J  {  public static void main(String[] args)  {  int x = 1000;  byte y = 10;  y = (byte) x;  System.out.println("done");  }  }  Q240 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  done | | |

|  |  |
| --- | --- |
| **95.** | class K  {  public static void main(String[] args)  {  int x = 129;  byte y = 10;  y = (byte) x;  System.out.println(x + "," + y);  }  }  Q241 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  129,-127 |  |  | | --- | | C.  129,129 | | |
| **96.** | class L  {  public static void main(String[] args)  {  int x = -130;  byte y = 10;  y = (byte) x;  System.out.println(x + "," + y);  }  }  Q242 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  129,-127 |  |  | | --- | | C.  -130,126 | | |

|  |  |
| --- | --- |
| **97.** | class M  {  public static void main(String[] args)  {  int i = 10;  short j = i;  System.out.println(j);  }  }  Q243 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  10 | | |
| **98.** | class N  {  public static void main(String[] args)  {  int i = 32768;  short j = (short) i;  System.out.println(j);  }  }  Q244 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  32768 |  |  | | --- | | C.  -32768 | | |

|  |  |  |
| --- | --- | --- |
| **99.** | class O  {  public static void main(String[] args)  {  int i = 130;  short j = (byte) i;  System.out.println(j);  }  }  Q245 | |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  -126 |  |  | | --- | | C.  -130 | | | |
| **100.** | | final key word is used to the abstract methodsQ328 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | | |

|  |  |
| --- | --- |
| **101.** | final key word is used to the private methodsQ329 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **102.** | final key word is used to the static methodsQ330 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **103.** | static methods can be an abstractQ331 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **104.** | absstract class can be finalQ332 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **105.** | constructor can be finalQ333 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **106.** | package src;  class A  {  A()  {  System.out.println("A()");  }  public static void main(String[] args)  {  System.out.println("main begin");  A a1 = new A();  System.out.println("-------------");  A a2 = new A();  System.out.println("-------------");  A a3 = new A();  System.out.println("-------------");  System.out.println("main end");  }  }  Q730 |
| |  | | --- | | A.  main begin  -------------  -------------  -------------  main end |  |  | | --- | | B.  main begin  A()  -------------  A()  -------------  A()  -------------  main end |  |  | | --- | | C.  Compie time error | | |

|  |  |
| --- | --- |
| **107.** | package src;  class B  {  int i;  B()  {  System.out.println("B()");  this.i = 20;  }  public static void main(String[] args)  {  System.out.println("main begin");  B obj1 = new B();  System.out.println("-------------");  B obj2 = new B();  System.out.println("-------------");  B obj3 = new B();  System.out.println("-------------");  System.out.println(obj1.i + ", " + obj2.i + ", " + obj3.i);  System.out.println("main end");  }  }  Q731 |
| |  | | --- | | A.  main begin  B()  -------------  B()  -------------  B()  -------------  20, 20, 20  main end |  |  | | --- | | B.  main begin  -------------  -------------  -------------  0, 0, 0  main end |  |  | | --- | | C.  Compie time error | | |
| **108.** | package src;  class C  {  int i;  C()  {  System.out.println("C()");  i = 20;  }  public static void main(String[] args)  {  System.out.println("main begin");  C obj1 = new C();  System.out.println("-------------");  C obj2 = new C();  System.out.println("-------------");  C obj3 = new C();  System.out.println("-------------");  System.out.println(obj1.i + ", " + obj2.i + ", " + obj3.i);  System.out.println("main end");  }  }  Q732 |
| |  | | --- | | A.  main begin  -------------  -------------  -------------  20, 20, 20  main end |  |  | | --- | | B.  main begin  C()  -------------  C()  -------------  C()  -------------  20, 20, 20  main end |  |  | | --- | | C.  Compie time error | | |

|  |  |
| --- | --- |
| **109.** | package src;  class D  {  int i;  D()  {  System.out.println("D():" + i);  i = 20;  }  public static void main(String[] args)  {  System.out.println("main begin");  D obj1 = new D();  System.out.println("-------------");  D obj2 = obj1;  System.out.println("-------------");  D obj3 = obj2;  System.out.println("-------------");  System.out.println(obj1.i + ", " + obj2.i + ", " + obj3.i);  System.out.println("main end");  }  }  Q733 |
| |  | | --- | | A.  main begin  -------------  -------------  -------------  20, 20, 20  main end |  |  | | --- | | B.  main begin  D():0  -------------  D():0  -------------  D():0  -------------  20, 20, 20  main end |  |  | | --- | | C.  main begin  D():0  -------------  -------------  -------------  20, 20, 20  main end |  |  | | --- | | D.  Compile time error | | |
| **110.** | package src;  class E  {  static E()  {  System.out.println("E()");  }  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q734 |
| |  | | --- | | A.  Hello World |  |  | | --- | | B.  Hello world  E() |  |  | | --- | | C.  E()  Hello World |  |  | | --- | | D.  Compile time error | | |

|  |  |
| --- | --- |
| **111.** | package src;  class F  {  void F()  {  System.out.println("F()");  }  public static void main(String[] args)  {  System.out.println("main begin");  F f1 = new F();  System.out.println("main end");  }  }  Q735 |
| |  | | --- | | A.  main begin  F()  main end |  |  | | --- | | B.  main begin  main end |  |  | | --- | | C.  Compile time error | | |
| **112.** | package src;  class G  {  public static void main(String[] args)  {  System.out.println("main begin");  G g1 = new G();  System.out.println("-----------");  G g2 = new G();  System.out.println("-----------");  System.out.println("main end");  }  }  Q736 |
| |  | | --- | | A.  main begin  -----------  -----------  main end |  |  | | --- | | B.  main begin  G()  -----------  G()  -----------  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **113.** | package src;  class H  {  public static void main(String[] args)  {  System.out.println("main begin");  H obj = new H(90);  System.out.println("main end");  }  }  /\*find Whether it compile and running successfullr or not? \*/Q737 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **114.** | package src;  class I  {  I(int i)  {  System.out.println("I(int)");  }  public static void main(String[] args)  {  System.out.println("main begin");  I obj = new I(90);  System.out.println("main end");  }  }  Q738 |
| |  | | --- | | A.  main begin  I(int)  main end |  |  | | --- | | B.  main begin  I(90)  main end |  |  | | --- | | C.  main begin  main end | | |

|  |  |
| --- | --- |
| **115.** | package src;  class J  {  J(int i)  {  System.out.println("J(int)");  }  public static void main(String[] args)  {  System.out.println("main begin");  J obj = new J();  System.out.println("main end");  }  }  Q739 |
| |  | | --- | | A.  main begin  J(int)  main end |  |  | | --- | | B.  main begin  J(int)  main end |  |  | | --- | | C.  main begin  main end |  |  | | --- | | D.  Compile time error | | |
| **116.** | package src;  class K  {  K()  {  System.out.println("K()");  }  public static void main(String[] args)  {  System.out.println("main begin");  K obj = new K(20);  System.out.println("main end");  }  }  Q740 |
| |  | | --- | | A.  main begin  K(int)  main end |  |  | | --- | | B.  main begin  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **117.** | package src;  class L  {  int i;  L()  {  System.out.println("L()");  i = 20;  }  public static void main(String[] args)  {  System.out.println("main begin");  L obj1 = new L();  System.out.println("---------");  L obj2 = new L();  System.out.println("---------");  L obj3 = new L();  System.out.println("---------");  System.out.println(obj1.i + ", " + obj2.i + "," + obj3.i);  System.out.println("main end");  }  }  Q741 |
| |  | | --- | | A.  main begin  L()  ---------  L()  ---------  L()  ---------  20, 20,20  main end |  |  | | --- | | B.  main begin  ---------  ---------  ---------  20, 20,20  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **118.** | package src;  class M  {  int i;  M(int k)  {  System.out.println("M(int)");  i = k;  }  public static void main(String[] args)  {  System.out.println("main begin");  M obj1 = new M(10);  System.out.println("---------");  M obj2 = new M(50);  System.out.println("---------");  M obj3 = new M(20);  System.out.println("---------");  System.out.println(obj1.i + ", " + obj2.i + "," + obj3.i);  System.out.println("main end");  }  }  Q742 |
| |  | | --- | | A.  main begin  ---------  ---------  ---------  10, 50,20  main end |  |  | | --- | | B.  main begin  M(int)  ---------  M(int)  ---------  M(int)  ---------  10, 50,20  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **119.** | package src;  class N  {  int i;  N(int i)  {  System.out.println("N(int)");  i = i;  }  public static void main(String[] args)  {  System.out.println("main begin");  N obj1 = new N(10);  System.out.println("---------");  N obj2 = new N(50);  System.out.println("---------");  N obj3 = new N(20);  System.out.println("---------");  System.out.println(obj1.i + ", " + obj2.i + "," + obj3.i);  System.out.println("main end");  }  }  Q743 |
| |  | | --- | | A.  main begin  N(int)  ---------  N(int)  ---------  N(int)  ---------  0, 0,0  main end |  |  | | --- | | B.  main begin  N(int)  ---------  N(int)  ---------  N(int)  ---------  10, 50,20  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **120.** | package src;  class O  {  int i;  O(int i)  {  System.out.println("O(int)");  this.i = i;  }  public static void main(String[] args)  {  System.out.println("main begin");  O obj1 = new O(10);  System.out.println("---------");  O obj2 = new O(50);  System.out.println("---------");  O obj3 = new O(20);  System.out.println("---------");  System.out.println(obj1.i + ", " + obj2.i + "," + obj3.i);  System.out.println("main end");  }  }  Q744 |
| |  | | --- | | A.  main begin  O(int)  ---------  O(int)  ---------  O(int)  ---------  0, 0,0  main end |  |  | | --- | | B.  main begin  O(int)  ---------  O(int)  ---------  O(int)  ---------  10, 50,20  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **121.** | package src;  class Person  {  String firstName;  int age;  Person(String firstName, int age)  {  this.firstName = firstName;  this.age = age;  }  public static void main(String[] args)  {  Person p1 = new Person("Vijay", 22);  Person p2 = new Person("Raghu", 24);  Person p3 = new Person("Ravi", 25);  System.out.println(p1.firstName + ", " + p1.age);  System.out.println(p2.firstName + ", " + p2.age);  System.out.println(p3.firstName + ", " + p3.age);  }  }  Q745 |
| |  | | --- | | A.  Vijay, 22  Raghu, 24  Ravi, 25 |  |  | | --- | | B.  Ravi, 25  Vijay, 22  Raghu, 24 |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **122.** | package src;  class A  {  int i;  public static void main(String[] args)  {  System.out.println(i);  }  }  Q816 |
| |  | | --- | | A.  0 |  |  | | --- | | B.  Compile time error | | |

|  |  |
| --- | --- |
| **123.** | package is used for grouping similar kind of classesQ989 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **124.** | package keyword can be anywhere in the fileQ990 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **125.** | package keyword should be as a first statement in the java fileQ991 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **126.** | how to compile if A.java file which is in pack1 folder with a package as pack1 declaration.Q992 |
| |  | | --- | | A.  javac pack1/A.java |  |  | | --- | | B.  javac pack1.A.java | | |

|  |  |
| --- | --- |
| **127.** | how to run class file(A.class) which is in pack1 packageQ993 |
| |  | | --- | | A.  java pack1/A |  |  | | --- | | B.  java pack1.A | | |
| **128.** | package is used for avoiding naming conflicts to the classesQ994 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **129.** | private member of any class should be used in the same class.Q995 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **130.** | we cant use private member of one class in another class.Q996 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **131.** | we can use private method of A class inside a B classQ997 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **132.** | we can use private method of A class inside a B class, if both the classes are in the same java file.Q998 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **133.** | constructor can be private.Q999 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **134.** | if class containing private constructor, then object should be created within the same class.Q1000 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false |  |  | | --- | | C.  we cant make constructor as a private. | | |

|  |  |
| --- | --- |
| **135.** | if class containing only private constror/constructors, then we cant develop a subclass.Q1001 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false |  |  | | --- | | C.  we cant make constructor as a private. | | |
| **136.** | private member of a class can be accessed from its subclass.Q1002 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **137.** | subclass constructor always should call only super class non private constructor.Q1003 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false |  |  | | --- | | C.  we cant make constructor as a private. | | |
| **138.** | private static methods of super class can be accessed from its subclassQ1004 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **139.** | outer classes can be privateQ1005 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **140.** | inner classes can be privateQ1006 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **141.** | outer classes can't be staticQ1007 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **142.** | inner classes can't be staticQ1008 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **143.** | satic initialization block can be privateQ1009 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **144.** | instance initialization block can't be privateQ1010 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **145.** | private members of super class inheriting to subclassQ1011 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **146.** | private members are surviving in the inheritanceQ1012 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **147.** | default scope also considered as a packageQ1013 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **148.** | default scoped members are allowed to use in the same package.Q1014 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **149.** | default scoped members are allowed to use in the other packages also.Q1015 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **150.** | default scoped members are inheriting to the subclasses of same package.Q1016 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **151.** | default scoped members are inheriting to the subclasses of any package.Q1017 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **152.** | protected scoped members are allowed to use in the same package.Q1018 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **153.** | protected scoped members are allowed to use in the other packages also.Q1019 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **154.** | protected and default both are one and the same in case of usageQ1020 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **155.** | protected scoped member can be inherited to every subclass including subclass of another packageQ1021 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **156.** | protected scoped member can be inherited to subclass of another packageQ1022 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **157.** | if protected member is inherining to subclass of another package, then it should be used in the within that subclass by using subclass reference type.Q1023 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **158.** | if protected member is inherining to subclass of another package, then it should be used in the within that subclass by using super class reference type.Q1024 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **159.** | if protected member is inherining to subclass of another package, then it can be used in outside a subclassQ1025 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **160.** | if protected member is inherining to subclass of another package, then it should be used in the same subclassQ1026 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **161.** | public scoped members are allowed to use only in the same package.Q1027 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **162.** | public scoped members are allowed to use in the other packages also.Q1028 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **163.** | public and protected both are one and the same in case of usageQ1029 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **164.** | public scoped member can be inherited to every subclass including subclass of another packageQ1030 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **165.** | public scoped member can be inherited to subclass of another packageQ1031 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **166.** | if public member is inherining to subclass of another package, then it should be used only in the within that subclass by using subclass reference type.Q1032 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **167.** | if public member is inherining to subclass of another package, then it should be used only in the within that subclass by using super class reference type.Q1033 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **168.** | if public member is inherining to subclass of another package, then it can be used in outside a subclassQ1034 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **169.** | if public member is inherining to subclass of another package, then it should be used only in the same subclassQ1035 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **170.** | only public classes can be used in another packages.Q1036 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **171.** | we can use one class in another class of the same package without import.Q1037 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **172.** | we can use another package class without importQ1038 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **173.** | package is used for grouping similar kind of classesQ1039 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **174.** | package keyword can be anywhere in the fileQ1040 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **175.** | package keyword should be as a first statement in the java fileQ1041 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **176.** | how to compile if A.java file which is in pack1 folder with a package as pack1 declaration.Q1042 |
| |  | | --- | | A.  javac pack1/A.java |  |  | | --- | | B.  javac pack1.A.java | | |

|  |  |
| --- | --- |
| **177.** | how to run class file(A.class) which is in pack1 packageQ1043 |
| |  | | --- | | A.  java pack1/A |  |  | | --- | | B.  java pack1.A | | |
| **178.** | package is used for avoiding naming conflicts to the classesQ1044 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **179.** | private member of any class should be used in the same class.Q1045 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **180.** | we cant use private member of one class in another class.Q1046 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **181.** | we can use private method of A class inside a B classQ1047 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **182.** | we can use private method of A class inside a B class, if both the classes are in the same java file.Q1048 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **183.** | constructor can be private.Q1049 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **184.** | if class containing private constructor, then object should be created within the same class.Q1050 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false |  |  | | --- | | C.  we cant make constructor as a private. | | |

|  |  |
| --- | --- |
| **185.** | if class containing only private constror/constructors, then we cant develop a subclass.Q1051 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false |  |  | | --- | | C.  we cant make constructor as a private. | | |
| **186.** | private member of a class can be accessed from its subclass.Q1052 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **187.** | subclass constructor always should call only super class non private constructor.Q1053 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false |  |  | | --- | | C.  we cant make constructor as a private. | | |
| **188.** | private static methods of super class can be accessed from its subclassQ1054 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **189.** | outer classes can be privateQ1055 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **190.** | inner classes can be privateQ1056 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **191.** | outer classes can't be staticQ1057 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **192.** | inner classes can't be staticQ1058 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **193.** | satic initialization block can be privateQ1059 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **194.** | instance initialization block can't be privateQ1060 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **195.** | private members of super class inheriting to subclassQ1061 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **196.** | private members are surviving in the inheritanceQ1062 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **197.** | default scope also considered as a packageQ1063 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **198.** | default scoped members are allowed to use in the same package.Q1064 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **199.** | default scoped members are allowed to use in the other packages also.Q1065 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **200.** | default scoped members are inheriting to the subclasses of same package.Q1066 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **201.** | default scoped members are inheriting to the subclasses of any package.Q1067 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **202.** | protected scoped members are allowed to use in the same package.Q1068 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **203.** | protected scoped members are allowed to use in the other packages also.Q1069 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **204.** | protected and default both are one and the same in case of usageQ1070 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **205.** | protected scoped member can be inherited to every subclass including subclass of another packageQ1071 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **206.** | protected scoped member can be inherited to subclass of another packageQ1072 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **207.** | if protected member is inherining to subclass of another package, then it should be used in the within that subclass by using subclass reference type.Q1073 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **208.** | if protected member is inherining to subclass of another package, then it should be used in the within that subclass by using super class reference type.Q1074 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **209.** | if protected member is inherining to subclass of another package, then it can be used in outside a subclassQ1075 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **210.** | if protected member is inherining to subclass of another package, then it should be used in the same subclassQ1076 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **211.** | public scoped members are allowed to use only in the same package.Q1077 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **212.** | public scoped members are allowed to use in the other packages also.Q1078 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **213.** | public and protected both are one and the same in case of usageQ1079 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **214.** | public scoped member can be inherited to every subclass including subclass of another packageQ1080 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **215.** | public scoped member can be inherited to subclass of another packageQ1081 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **216.** | if public member is inherining to subclass of another package, then it should be used only in the within that subclass by using subclass reference type.Q1082 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **217.** | if public member is inherining to subclass of another package, then it should be used only in the within that subclass by using super class reference type.Q1083 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **218.** | if public member is inherining to subclass of another package, then it can be used in outside a subclassQ1084 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **219.** | if public member is inherining to subclass of another package, then it should be used only in the same subclassQ1085 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **220.** | only public classes can be used in another packages.Q1086 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **221.** | we can use one class in another class of the same package without import.Q1087 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |
| **222.** | we can use another package class without importQ1088 |
| |  | | --- | | A.  true |  |  | | --- | | B.  false | | |

|  |  |
| --- | --- |
| **223.** | incase of derive data type, upcasting is an automatic.Q1224 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **224.** | incase of derive data type, upcasting should be an explicit.Q1225 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **225.** | incase of derive data type, downcasting is an automatic.Q1226 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **226.** | incase of derive data type, upcasting should be an explicit.Q1227 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **227.** | method argument is super class type and while calling the same method supplying subclasstype is proper syntax.Q1228 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **228.** | method argument is subclass type and while calling the same method supplying super classtype is proper syntax.Q1229 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **229.** | method return type is Object type and while calling the same, is it possible to take the return value in to String type variable?Q1230 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **230.** | method return type is String type and while calling the same, is it possible to take the return value in to Object type variable?Q1231 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **231.** | method return type is String type and while calling the same, is it possible to take the return value in to Object type variable?Q1232 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **232.** | Object obj = new String(); //will it compileQ1233 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **233.** | String obj = new Object(); //will it compileQ1234 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **234.** | Object obj = (Object) new String(); //will it compileQ1235 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **235.** | String obj = (String) new Object(); //will it compileQ1236 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **236.** | Which option gives runtime excfeption?Q1237 |
| |  | | --- | | A.  Object obj = new Object() |  |  | | --- | | B.  Object obj = new String() |  |  | | --- | | C.  String obj = (String) new Object(); |  |  | | --- | | D.  Object Obj = (Object) new String(); | | |

|  |  |
| --- | --- |
| **237.** | What is the output?  String s1 = new String();  System.out.print(s1 instanceof Object);  System.out.print(s1 instanceof String);Q1238 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  truetrue |  |  | | --- | | D.  truefalse |  |  | | --- | | E.  falsefalse | | |
| **238.** | What is the output?  Object s1 = new String();  System.out.print(s1 instanceof Object);  System.out.print(s1 instanceof String);Q1239 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  truetrue |  |  | | --- | | D.  truefalse |  |  | | --- | | E.  falsefalse | | |

|  |  |
| --- | --- |
| **239.** | What is the output?  String s1 = new Object();  System.out.print(s1 instanceof Object);  System.out.print(s1 instanceof String);Q1240 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  truetrue |  |  | | --- | | D.  truefalse |  |  | | --- | | E.  falsefalse | | |
| **240.** | What is the output?  String s1 = (String) new Object();  System.out.print(s1 instanceof Object);  System.out.print(s1 instanceof String);Q1241 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  truetrue |  |  | | --- | | D.  truefalse |  |  | | --- | | E.  falsefalse | | |

|  |  |
| --- | --- |
| **241.** | What is the output?  Object s1 = new Object();  System.out.print(s1 instanceof Object);  System.out.print(s1 instanceof String);Q1242 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  truetrue |  |  | | --- | | D.  truefalse |  |  | | --- | | E.  falsefalse | | |
| **242.** | What is the output?  String s1 = new String("4.5");  System.out.print(s1 instanceof Integer);  System.out.print(s1 instanceof Double);Q1243 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  truetrue |  |  | | --- | | D.  truefalse |  |  | | --- | | E.  falsefalse | | |

|  |  |
| --- | --- |
| **243.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  A a1 = new A();  System.out.print(a1.i);  System.out.print(a1.j);  }  }Q1244 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |
| **244.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  A a1 = (B)new A();  System.out.print(a1.i);  System.out.print(a1.j);  }  }Q1245 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |

|  |  |
| --- | --- |
| **245.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  A a1 = new B();  System.out.print(a1.i);  System.out.print(a1.j);  }  }Q1246 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |
| **246.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  A a1 = (B) new B();  System.out.print(a1.i);  System.out.print(a1.j);  }  }Q1247 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |

|  |  |
| --- | --- |
| **247.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  B b1 = new B();  System.out.print(b1.i);  System.out.print(b1.j);  }  }Q1248 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |
| **248.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  B b1 = new A();  System.out.print(b1.i);  System.out.print(b1.j);  }  }Q1249 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |

|  |  |
| --- | --- |
| **249.** | What is the output?  class A  {  int i = 10;  }  class B extends A  {  int j = 20;  }  class C  {  public static void main(String[] args)  {  B b1 = (B) new A();  System.out.print(b1.i);  System.out.print(b1.j);  }  }Q1250 |
| |  | | --- | | A.  compilation error |  |  | | --- | | B.  runtime error |  |  | | --- | | C.  1020 |  |  | | --- | | D.  2010 |  |  | | --- | | E.  1010 | | |
| **250.** | method overriding is required to achieve polimorphism.Q1251 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **251.** | constructor overriding is required to achieve polimorphism.Q1252 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **252.** | upcasting is required to achieve polimorphism.Q1253 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **253.** | downcasting is required to achieve polimorphism.Q1254 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **254.** | which two are required to achieve polimorphism.Q1255 |
| |  | | --- | | A.  down casting |  |  | | --- | | B.  up casting |  |  | | --- | | C.  constructor overriding |  |  | | --- | | D.  method overriding | | |

|  |  |
| --- | --- |
| **255.** | static methods cant be overridedQ1256 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **256.** | static methods can be overridedQ1257 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **257.** | static methods are involving in the polymorphismQ1258 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **258.** | subclass doen't required to load while accessing super class static member with a subclass name.Q1259 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **259.** | developing a class to represent entity information is called as \_\_\_\_\_Q1260 |
| |  | | --- | | A.  Encapsulation |  |  | | --- | | B.  Inheritance |  |  | | --- | | C.  abstraction |  |  | | --- | | D.  polymorphism | | |
| **260.** | How to represent an entity in the object oriented programming languages?Q1261 |
| |  | | --- | | A.  package |  |  | | --- | | B.  class |  |  | | --- | | C.  file |  |  | | --- | | D.  project | | |

|  |  |
| --- | --- |
| **261.** | Is it adisable to develop two entity information in one class?Q1262 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **262.** | Is it adisable to develop one entity information in two classes?Q1263 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **263.** | What could be the relation between entity and classQ1264 |
| |  | | --- | | A.  one-to-one |  |  | | --- | | B.  one-to-many |  |  | | --- | | C.  many-to-one | | |
| **264.** | classpath is required while using any class wich is not in the current developmentQ1265 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **265.** | which option is used to specify location of third party classQ1266 |
| |  | | --- | | A.  path |  |  | | --- | | B.  classpath |  |  | | --- | | C.  java\_dev | | |
| **266.** | is it possible to specify zip file name and its location under classpath?Q1267 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **267.** | is it possible to specify jar file name and its location under classpath?Q1268 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **268.** | -classpath is one of the option to javac command.Q1269 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **269.** | -classpath is used to javac command to specify third part class location.Q1270 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **270.** | -cp is one of the option to javac command.Q1271 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **271.** | -cp is used to javac command to specify third part class location.Q1272 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **272.** | -classpath is one of the option to java command.Q1273 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **273.** | -classpath is used to java command to specify third part class location.Q1274 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **274.** | -cp is one of the option to java command.Q1275 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **275.** | -cp is used to java command to specify third part class location.Q1276 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **276.** | -cp is used to java command to specify jar file name and its locationQ1277 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **277.** | -cp is used to javac command to specify jar file name and its locationQ1278 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **278.** | -cp is used to javac command to specify jar file name and its locationQ1279 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **279.** | while updating a classpath to javac comand, you should incorporate dot as one of the entry without fail.Q1280 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **280.** | while updating a classpath to java comand, you should incorporate dot as one of the entry without fail.Q1281 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **281.** | -cp option is used for one command. this effect will not be available to any subsequent commands.Q1282 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **282.** | what is the delimiter to multiple entries in the classpath in the windows environmentQ1283 |
| |  | | --- | | A.  comma |  |  | | --- | | B.  dot |  |  | | --- | | C.  colon |  |  | | --- | | D.  semi colon | | |

|  |  |
| --- | --- |
| **283.** | what is the delimiter to multiple entries in the classpath in the linux environmentQ1284 |
| |  | | --- | | A.  comma |  |  | | --- | | B.  dot |  |  | | --- | | C.  colon |  |  | | --- | | D.  semi colon | | |
| **284.** | set classpath is usd to set a classpath for the command prompt. available to all the commands of the same command prompt.Q1285 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **285.** | How to update a classpath while triggering a javac command by considering existing classpath of the command prompt?Q1286 |
| |  | | --- | | A.  javac -cp some\_new\_location A.java |  |  | | --- | | B.  javac -cp some\_new\_location;classpath A.java |  |  | | --- | | C.  javac -cp some\_new\_location;%classpath A.java |  |  | | --- | | D.  javac -cp some\_new\_location;%classpath% A.java | | |
| **286.** | How to read a classpath of the command prompt?Q1287 |
| |  | | --- | | A.  echo classpath |  |  | | --- | | B.  echo %classpath |  |  | | --- | | C.  echo %classpath% | | |

|  |  |
| --- | --- |
| **287.** | how to set a classpath permanantly to all the command prompts.Q1288 |
| |  | | --- | | A.  with -cp |  |  | | --- | | B.  with set classpath |  |  | | --- | | C.  under env variables in the my computer | | |
| **288.** | constructors are inheriting to sub classQ1420 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **289.** | initialzation blocks are inheriting to sub classQ1421 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **290.** | super class constructor not inheriting to subclass. subclass constructor calling super class constructor.Q1422 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **291.** | It is not possible to develop multiple constructors in the same classQ1757 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **292.** | While overloading a constructor, constructor arguments must be sameQ1758 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **293.** | While overloading a constructor, constructor signature must not be sameQ1759 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **294.** | we can create only one object by using one constructorQ1760 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **295.** | We can call one constructor from another constructor by using this calling statmentQ1761 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **296.** | this calling statment always should be the last statment in the constructor body.Q1762 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **297.** | this calling statment can be used to call one method from another method.Q1763 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **298.** | For every this calling statment corresponding constructor should be availableQ1764 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **299.** | Recursive constructor invocation is not allowed in constructorsQ1765 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **300.** | this with paranthesis is not a calling statmentQ1766 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **301.** | class A  {  A(){  System.out.println("A()");  }  A(int i){  System.out.println("A(int)");  }  public static void main(String[] args)  {  System.out.println("main begin");  A a1 = new A();  System.out.println("---------------------");  A a2 = new A(20);  System.out.println("---------------------");  A a3 = new A(20);  System.out.println("---------------------");  System.out.println("main end");  }  }  Q1767 |
| |  | | --- | | A.  main begin  A()  ---------------------  A(int)  ---------------------  A(int)  ---------------------  main end |  |  | | --- | | B.  main begin  A()  ---------------------  A(int)  ---------------------  A()  ---------------------  main end |  |  | | --- | | C.  main begin  A()  ---------------------  A()  ---------------------  A(int)  ---------------------  main end |  |  | | --- | | D.  main begin  A(int)  ---------------------  A(int)  ---------------------  A(int)  ---------------------  main end | | |
| **302.** | class B  {  B(double i){  System.out.println("B(double)");  }  B(int i){  System.out.println("B(int)");  }  public static void main(String[] args)  {  System.out.println("main begin");  B a1 = new B(20);  System.out.println("---------------------");  B b2 = new B(20.5);  System.out.println("---------------------");  B b3 = new B(20);  System.out.println("---------------------");  System.out.println("main end");  }  }  Q1768 |
| |  | | --- | | A.  main begin  A(int)  ---------------------  A(int)  ---------------------  A(double)  ---------------------  main end |  |  | | --- | | B.  main begin  A(double)  ---------------------  A(int)  ---------------------  A(int)  ---------------------  main end |  |  | | --- | | C.  main begin  B(int)  ---------------------  B(double)  ---------------------  B(int)  ---------------------  main end | | |

|  |  |
| --- | --- |
| **303.** | class C  {  C(){  System.out.println("C()");  }  C(double i){  System.out.println("C(double)");  }  C(int i){  System.out.println("C(int)");  }  C(int i, double j){  System.out.println("C(int, double)");  }  public static void main(String[] args)  {  System.out.println("main begin");  C a1 = new C(20);  System.out.println("---------------------");  C b2 = new C(20.5);  System.out.println("---------------------");  C b3 = new C(20);  System.out.println("---------------------");  C b4 = new C();  System.out.println("---------------------");  System.out.println("main end");  }  }  Q1769 |
| |  | | --- | | A.  main begin  C()  ---------------------  C(double)  ---------------------  C(int)  ---------------------  C(int)  ---------------------  main end |  |  | | --- | | B.  main begin  C(int)  ---------------------  C(double)  ---------------------  C(int)  ---------------------  C()  ---------------------  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **304.** | class D  {  D(int i){  System.out.println("D(int i)");  }  D(int j){  System.out.println("D(int j)");  System.out.println("D(int j)");  System.out.println("D(int j)");  System.out.println("D(int j)");  }  public static void main(String[] args)  {  System.out.println("main begin");  D d1 = new D(20);  System.out.println("-------------");  D d1 = new D(30);  System.out.println("-------------");  System.out.println("Main end");  }  }  Q1770 |
| |  | | --- | | A.  Main begin  20  -------------  30  30  30  30  -------------  Main end |  |  | | --- | | C.  Compile time error | | |

|  |  |
| --- | --- |
| **305.** | class E  {  E(int i, int j){  }  E(int i, double j){  }  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  // Check wether this program compiles successfully or notQ1771 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **306.** | class F  {  F(){  System.out.println("F()");  }  F(int i){  System.out.println("F(int)");  }  public static void main(String[] args)  {  System.out.println("Main begin");  F obj1 = new F();  System.out.println("------------");  F obj2 = new F();  System.out.println("------------");  F obj3 = new F(30);  System.out.println("------------");  System.out.println("Main end");  }  }  Q1772 |
| |  | | --- | | A.  Main begin  F()  ------------  F()  ------------  F(int)  ------------  Main end |  |  | | --- | | B.  Main begin  F()  ------------  F(int)  ------------  F()  ------------  Main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **307.** | class G  {  G(){  System.out.println("G()");  }  G(int i){  this();  System.out.println("G(int)");  }  public static void main(String[] args)  {  System.out.println("Main begin");  G obj1 = new G();  System.out.println("------------");  G obj2 = new G();  System.out.println("------------");  G obj3 = new G(30);  System.out.println("------------");  System.out.println("Main end");  }  }  Q1773 |
| |  | | --- | | C.  Main begin  G()  ------------  G()  G(int)  ------------  G()  ------------  Main end |  |  | | --- | | C.  Main begin  G()  ------------  G(int)  ------------  G()  G(int)  ------------  Main end |  |  | | --- | | C.  Main begin  G()  ------------  G()  ------------  G()  G(int)  ------------  Main end |  |  | | --- | | D.  Compilation error | | |
| **308.** | class H  {  H(){  this(10);  System.out.println("H()");  }  H(int i){  System.out.println("H(int)");  }  public static void main(String[] args)  {  System.out.println("Main begin");  H obj1 = new H();  System.out.println("------------");  H obj2 = new H();  System.out.println("------------");  H obj3 = new H(30);  System.out.println("------------");  System.out.println("Main end");  }  }  Q1774 |
| |  | | --- | | A.  Main begin  H(int)  H()  ------------  H(int)  H()  ------------  H(int)  ------------  Main end |  |  | | --- | | B.  Main begin  H(int)  ------------  H()  ------------  H(int)  H()  ------------  Main end |  |  | | --- | | C.  Main begin  H()  H()  ------------  H(int)  H()  ------------  H(int)  ------------  Main end |  |  | | --- | | D.  Compilation error |  |  | | --- | | E.  None | | |

|  |  |
| --- | --- |
| **309.** | class I  {  I(){  this(10);  System.out.println("I()");  }  I(int i){  this(i, i);  System.out.println("I(int)");  }  I(int i, int j){  System.out.println("I(int, int)");  }  public static void main(String[] args)  {  System.out.println("Main begin");  I obj1 = new I();  System.out.println("------------");  I obj2 = new I();  System.out.println("------------");  I obj3 = new I(30);  System.out.println("------------");  I obj4 = new I(30, 40);  System.out.println("------------");  System.out.println("Main end");  }  }  Q1775 |
| |  | | --- | | B.  Main begin  I(int)  I(int)  I()  ------------  I(int, int)  I()  I(int)  ------------  I(int, int)  I(int)  ------------  I(int, int)  ------------  Main end |  |  | | --- | | B.  Main begin  I(int, int)  I(int)  I()  ------------  I(int, int)  I(int)  I()  ------------  I(int, int)  I(int)  ------------  I(int, int)  ------------  Main end |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |
| **310.** | class J  {  J(){  System.out.println("J()");  this(90);  }  J(int i){  System.out.println("J(int)");  }  public static void main(String[] args)  {  J j1 = new J();  J j1 = new J(10);  System.out.println("Hello World!");  }  }  Q1776 |
| |  | | --- | | A.  J()  J(int)  J(int)  Hello World! |  |  | | --- | | B.  J(int)  J()  J(int)  Hello World! |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **311.** | class K  {  K(){  this(90);  System.out.println("K()");  }  public static void main(String[] args)  {  K k1 = new K();  System.out.println("Hello World!");  }  }  Q1777 |
| |  | | --- | | A.  Hello World! |  |  | | --- | | B.  K()  Hello World! |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |
| **312.** | class L  {  L(){  this(90);  System.out.println("L()");  }  L(int i){  this();  System.out.println("L()");  }  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  // Will it compiles succesfull or not?Q1778 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **313.** | class M  {  M(){  this();  System.out.println("M()");  }  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  // Will it compiles succesfull or not?Q1779 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **314.** | package src;  class A  {  public static void main(String[] args)  {  final int i = 10;  int j = 20;  System.out.println(i);  System.out.println(j);  i = 10;  j = 20;  System.out.println(i);  System.out.println(j);  }  }  Q1780 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  10  20  10  20 |  |  | | --- | | C.  10 20 10 20 |  |  | | --- | | D.  Runtime Error | | |

|  |  |
| --- | --- |
| **315.** | package src;  class B  {  public static void main(String[] args)  {  final int i;  i = 10;  int j = 20;  System.out.println(i);  System.out.println(j);  i = 10;  j = 20;  System.out.println(i);  System.out.println(j);  }  }  Q1781 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  10  20  10  20 |  |  | | --- | | C.  10 20 10 20 |  |  | | --- | | D.  Runtime Error | | |
| **316.** | package src;  class C  {  public static void main(String[] args)  {  C c1 = new C();  c1 = new C();  final C c2 = new C();  c2 = new C();  System.out.println("done");  }  }  Will it compiles successfully are not?Q1782 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **317.** | package src;  class D  {  int i;  public static void main(String[] args)  {  D d1 = new D();  d1.i = 10;  final D d2 = new D();  d2.i = 10;  d1.i = 10;  d2.i = 10;  System.out.println("Hello World!");  }  }  Q1783 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Hello World! |  |  | | --- | | D.  None | | |
| **318.** | package src;  class E  {  final int i = 10;  public static void main(String[] args)  {  E e2 = new E();  e2 = new E();  System.out.println("done");  }  }  Q1784 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Runtime Error | | |

|  |  |
| --- | --- |
| **319.** | package src;  class F  {  public static void main(String[] args)  {  final int[] x = new int[3];  x[2] = 20;  System.out.println("done");  }  }  Q1785 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Runtime Error | | |
| **320.** | package src;  class G  {  public static void main(String[] args)  {  final int[] x = new int[3];  x = new int[3];  System.out.println("done");  }  }  Will it compiles fine or not?Q1786 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **321.** | package src;  class H  {  public static void main(final String[] args)  {  args[0] = "abc";  System.out.println("done");  }  }  Q1787 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  ArrayIndexOutOfBoundsException |  |  | | --- | | D.  None | | |
| **322.** | package src;  class I  {  public static void main(final String[] args)  {  args = null;  System.out.println("done");  }  }  Will it compiles fine are not?Q1788 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **323.** | package src;  class J  {  static void test(int x, final int y)  {  x = 10;  y = 10;  System.out.println("---------");  x = 10;  y = 10;  System.out.println("---------");  }  }  Q1789 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  ---------  --------- |  |  | | --- | | C.  Runtime Error |  |  | | --- | | D.  None | | |
| **324.** | package src;  class K  {  final int i;  }  Will it compiles fine are not?Q1790 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **325.** | package src;  class L  {  final int i = 0;  }  Will it compiles fine are not?Q1791 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **326.** | package src;  class M  {  final int i;  M()  {  i = 20;  }  }  Will it compiles fine are not?Q1792 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **327.** | package src;  class N  {  final int i = 20;  N()  {  i = 20;  }  }  Will it compiles fine are not?Q1793 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **328.** | package src;  class O  {  final int i;  {  i = 20;  }  }  Will it compiles fine are not?Q1794 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **329.** | package src;  class P  {  final int i = 20;  {  i = 20;  }  }  Will it compiles fine are Not?Q1795 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **330.** | package src;  class Q  {  final int i;  Q()  {  i = 20;  }  {  i = 20;  }  }  Q1796 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **331.** | package src;  class R  {  final int i = 10;  public static void main(String[] args)  {  R r1 = new R();  r1.i = 20;  System.out.println("done");  }  }  Q1797 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Runtime Error | | |
| **332.** | package src;  class S  {  final int i;  S()  {  i = 10;  }  S(int x)  {  i = 20;  }  }  Will it compiles fine are not?Q1798 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **333.** | package src;  class T  {  final int i;  T()  {  i = 10;  }  T(int x)  {  this();  i = 20;  }  }  Will it compiles Fine are not?Q1799 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **334.** | package src;  class U  {  final int i;  U()  {  i = 10;  }  U(int x)  {  }  }  Will it compiles fine are not?Q1800 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **335.** | package src;  class V  {  final int i;  V()  {  i = 10;  }  V(int x)  {  this();  }  }  Will it compiles fine are not?Q1801 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **336.** | package src;  class W  {  static final int x;  public static void main(String[] args)  {  System.out.println("done");  }  }  Q1802 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Runtime Error | | |

|  |  |
| --- | --- |
| **337.** | package src;  class X  {  static final int x = 0;  public static void main(String[] args)  {  System.out.println("done");  }  }  Q1803 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Runtime Error |  |  | | --- | | D.  0 | | |
| **338.** | package src;  class Y  {  static final int x = 0;  public static void main(String[] args)  {  x = 0;  System.out.println("done");  }  }  Q1804 |
| |  | | --- | | A.  done |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Runtime Error |  |  | | --- | | D.  0 | | |

|  |  |
| --- | --- |
| **339.** | package src;  class Z  {  final static int i;    static  {  i = 10;  }  }  Will it compiles fine are not?Q1805 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **340.** | package src;  class Z1  {  final static int i = 10;    static  {  i = 10;  }  }  Will it compiles fine are not?Q1806 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **341.** | package src;  class Z2  {  static final int SOME\_VALUE = 100;  public static void main(String[] args)  {  System.out.println(SOME\_VALUE);  }  }  Q1807 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  100 |  |  | | --- | | D.  0 | | |
| **342.** | package src;  class Z3  {  public static void main(String[] args)  {  System.out.println(Byte.MIN\_VALUE);  System.out.println(Byte.MAX\_VALUE);  System.out.println(Thread.MIN\_PRIORITY);  }  }  Q1808 |
| |  | | --- | | A.  -128  127  1 |  |  | | --- | | B.  128  -127  1 |  |  | | --- | | C.  Compilation Error | | |

|  |  |
| --- | --- |
| **343.** | package src;  interface A  {  int i = 10;  }  class Z4  {  public static void main(String[] args)  {  System.out.println(A.i);  }  }  Q1809 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  10 |  |  | | --- | | C.  0 |  |  | | --- | | D.  None | | |
| **344.** | package src;  interface A  {  int i;  }  class Z5  {  public static void main(String[] args)  {  System.out.println(A.i);  }  }  Q1810 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  10 |  |  | | --- | | C.  0 |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **345.** | package src;  interface A3  {  String COUNTRY\_NAME = "Great India";  }  class Z6  {  public static void main(String[] args)  {  System.out.println(A3.COUNTRY\_NAME);  }  }  Q1811 |
| |  | | --- | | A.  Great India |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Running successfully with no output | | |
| **346.** | class A  {  void test1()  {  }  final void test2()  {  }  }  class B extends A  {  void test1()  {  }  void test2()  {  }  }  class M1  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1812 |
| |  | | --- | | A.  Hello World! |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  None | | |

|  |  |
| --- | --- |
| **347.** | package src;  abstract class A  {  abstract final void test1();  }  class M2  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Will it compiles fine are not?Q1813 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **348.** | package src;  abstract class A  {  static abstract void test1();  }  class M3  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1814 |
| |  | | --- | | A.  Hello World! |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Compilation Error | | |

|  |  |
| --- | --- |
| **349.** | package src;  class A  {  static final void test1()  {  }  static void test2()  {  }  }  class B extends A  {  static void test1()  {  }  static void test2()  {  }  }  class M4  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1815 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Hello World! | | |
| **350.** | class A  {  private final void test1()  {  }  }  class B extends A  {  void test1()  {  }  }  class M5  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1816 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Hello World! | | |

|  |  |
| --- | --- |
| **351.** | package src;  class A  {  final A()  {  }  }  class M6  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1817 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Hello World! | | |
| **352.** | package src;  final class A  {  }  class B extends A  {  }  class M7  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1818 |
| |  | | --- | | A.  Hello World! |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Compilation Error | | |

|  |  |
| --- | --- |
| **353.** | package src;  abstract final class A  {  }  class M8  {  public static void main(String[] args)  {  System.out.println("Hello World!");  }  }  Q1819 |
| |  | | --- | | A.  Hello World! |  |  | | --- | | B.  RunTime Error |  |  | | --- | | C.  Compilation Error | | |
| **354.** | final variab;les can be modified furtherQ1820 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **355.** | Is it possible to declare main mathod arguments as finalQ1821 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **356.** | Final global variable can survive with a default valueQ1822 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **357.** | Is it possible to intilize final global variable with intilizersQ1823 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **358.** | static is loading into the memory while class is loadingQ1824 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **359.** | In java static and final is perfect constantQ1825 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **360.** | Is it possible to keep method as finalQ1826 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **361.** | Is it possible to override a final methodQ1827 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **362.** | Is it possible to use both abstract and final togetherQ1828 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **363.** | static method can be qulaified as abstract?Q1829 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **364.** | Class can not be finalQ1830 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **365.** | In java how many types of comments are there?Q1831 |
| |  | | --- | | A.  1 |  |  | | --- | | B.  2 |  |  | | --- | | C.  3 |  |  | | --- | | D.  4 | | |
| **366.** | We can keep single line comment any where in the programQ1832 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **367.** | We can keep multi line comment any where in the programQ1833 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **368.** | We can keep javadoc comment any where in the programQ1834 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **369.** | package src;  class A  {  int i;  public static void main(String[] args)  {  System.out.println(i);  }  }  Q1837 |
| |  | | --- | | A.  0 |  |  | | --- | | B.  Compile time error | | |
| **370.** | class G  {  static int p;  G(){  p += 10;  }  {  p += 20;  }  void test1(){  p += 30;  }  static void test2(){  p += 40;  }  public static void main(String[] args)  {  System.out.println("a: " + p);  G g1 = new G();  System.out.println("b: " + p);  g1.test1();  System.out.println("c: " + p);  test2();  System.out.println("d: " + p);  }  }  Q1838 |
| |  | | --- | | A.  a: 0  b: 30  c: 60  d: 100 |  |  | | --- | | B.  a: 20  b: 20  c: 20  d: 20 |  |  | | --- | | C.  Compilation error | | |

|  |  |
| --- | --- |
| **371.** | class H  {  static int p = 10;  public static void main(String[] args)  {  System.out.println("a: " + p);  System.out.println("b: " + H.p);  H obj = new H();  System.out.println("c: " + obj.p);  }  }  Q1839 |
| |  | | --- | | A.  a: 10  b: 10  c: 10 |  |  | | --- | | B.  Compilation error |  |  | | --- | | C.  None | | |
| **372.** | class I  {  static int count;  I(){  count++;  }  public static void main(String[] args)  {  I obj1 = new I();  System.out.println("a: " + count);  I obj2 = new I();  System.out.println("b: " + count);  I obj3 = new I();  System.out.println("c: " + count);  I obj4 = new I();  System.out.println("d: " + count);  }  }  Q1840 |
| |  | | --- | | A.  a: 1  b: 1  c: 1  d: 1 |  |  | | --- | | B.  a: 1  b: 2  c: 3  d: 4 |  |  | | --- | | C.  Compilation error |  |  | | --- | | D.  none | | |

|  |  |
| --- | --- |
| **373.** | class J  {  static int count;  J(){  count++;  }  public static void main(String[] args)  {  J obj1 = new J();  System.out.println("a: " + count);  J obj2 = new J();  System.out.println("b: " + count);  J obj3 = new J();  System.out.println("c: " + count);  J obj4 = new J();  System.out.println("d: " + count);  System.out.println("e: " + obj1.count);  System.out.println("f: " + obj2.count);  System.out.println("g: " + obj3.count);  System.out.println("h: " + obj4.count);  }  }  Q1841 |
| |  | | --- | | A.  a: 1  b: 2  c: 3  d: 4  e: 1  f: 2  g: 3  h: 4 |  |  | | --- | | B.  a: 1  b: 1  c: 1  d: 1  e: 1  f: 2  g: 3  h: 4 |  |  | | --- | | C.  a: 1  b: 2  c: 3  d: 4  e: 4  f: 4  g: 4  h: 4 |  |  | | --- | | D.  Compilation error | | |
| **374.** | class K  {  static int count;  K(){  count ++;  }  K(int i){  count ++;  }  K(int i, int j){  count ++;  }  public static void main(String[] args)  {  K k1 = new K();  K k2 = new K(10);  K k3 = new K(10, 20);  K k4 = new K(20);  K k5 = new K();  System.out.println(count);  }  }  Q1842 |
| |  | | --- | | A.  5 |  |  | | --- | | B.  3 |  |  | | --- | | C.  2 |  |  | | --- | | D.  Compilation error | | |

|  |  |
| --- | --- |
| **375.** | class L  {  static int count;  L(){  }  L(int i){  }  L(int i, int j){  }  {  count ++;  }  public static void main(String[] args)  {  L obj1 = new L();  L obj2 = new L();  L obj3 = new L(20);  L obj4 = new L();  L obj5 = new L();  L obj6 = new L(1, 4);  L obj7 = new L();  L obj8 = new L();  L obj9 = new L();  L obj10 = new L(7, 8);  System.out.println(count);  }  }  Q1843 |
| |  | | --- | | A.  7 |  |  | | --- | | B.  10 |  |  | | --- | | C.  3 |  |  | | --- | | D.  Compilation error | | |
| **376.** | Is it possible to use static varibles any wherr in the classQ1844 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **377.** | Is this possible static member of one class can be used in another class without class nameQ1845 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **378.** | Is it possible to access static variable along with referance variableQ1846 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **379.** | Scanner class available in the java.util packageQ1907 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **380.** | Scanner class is used read input while program is under executionQ1908 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **381.** | nextLine() method is used to read multiple strings with space as a seperatorQ1909 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **382.** | next() method read only one stringQ1910 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **383.** | nextInt() method is used to read StringQ1911 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **384.** | nextDouble() method is used to read double valuesQ1912 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **385.** | nextBoolean() method is used to read boolean valuesQ1913 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **386.** | nextChar() method is used to read char valuesQ1914 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **387.** | Which of the following is a derived data typeQ1915 |
| |  | | --- | | A.  class |  |  | | --- | | B.  interface |  |  | | --- | | C.  int |  |  | | --- | | D.  Both a and b |  |  | | --- | | E.  Both b and c | | |
| **388.** | The members which are under one inheritance are involved in derived castingQ1916 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **389.** | Is it possible to use use sub class where ever supper class is requiredQ1917 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **390.** | Which type of casting is automaticQ1918 |
| |  | | --- | | A.  Down casting |  |  | | --- | | B.  Up casting | | |

|  |  |
| --- | --- |
| **391.** | Object class is the super most classQ1919 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **392.** | We need to do up casting explicitlyQ1920 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **393.** | Compiler only do the down casting automaticallyQ1921 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **394.** | Every class is a sub class to object classesQ1922 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |

|  |  |
| --- | --- |
| **395.** | Is it possible to inject variables once after object got createdQ1923 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **396.** | To avoid getting class cast exception we can use instanceof operatorQ1924 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **397.** | If object type is a specific class type instanceof operator returnsQ1925 |
| |  | | --- | | A.  True |  |  | | --- | | B.  False | | |
| **398.** | Is it possible to use instanceof accross the classesQ1926 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **399.** | package src;  class A  {  int i;  public static void main(String[] args)  {  System.out.println(i);  }  }  Q2169 |
| |  | | --- | | A.  0 |  |  | | --- | | B.  Compile time error | | |
| **400.** | package src;  class A  {  int i;  public static void main(String[] args)  {  System.out.println(i);  }  }  Q2170 |
| |  | | --- | | A.  0 |  |  | | --- | | B.  Compile time error | | |

|  |  |
| --- | --- |
| **401.** | package src;  class B  {  void test()  {    }  public static void main(String[] args)  {  test();  System.out.println("Hello World!");  }  }  Q2171 |
| |  | | --- | | A.  Hello World |  |  | | --- | | B.  Compile time error | | |
| **402.** | package src;  class C  {  static void test1()  {  System.out.println("Hello World!");  test2();  }  void test2()  {  System.out.println("Hello World!");  }  }  // Find out whether it compiles succesfully are not?Q2172 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **403.** | package src;  class D  {  public void test1()  {  System.out.println("Hello World!");  }  static  {  test1();  }  }  //Find out whether it compiles succesfully are not?Q2173 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **404.** | package src;  class E  {  int i;  static  {  i = 20;  }  }  // Find out whether it compiles succesfully are not?Q2174 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **405.** | package src;  class F  {  int i;  public static void main(String[] args)  {  F f1 = new F();  System.out.println(f1.i);  }  }  Q2175 |
| |  | | --- | | A.  0 |  |  | | --- | | B.  Compile time error | | |
| **406.** | package src;  class G  {  void test()  {  System.out.println("from test");  }  public static void main(String[] args)  {  System.out.println("main begin");  G g1 = new G();  g1.test();  System.out.println("main end");  }  }  Q2176 |
| |  | | --- | | A.  main begin  from test  main end |  |  | | --- | | B.  main begin  main end |  |  | | --- | | C.  Compile time error | | |

|  |  |
| --- | --- |
| **407.** | package src;  class H  {  void test()  {  }  static  {  H h1 = new H();  h1.test();  }  }  /\* Find out whether it compiles successfully are not? \*/Q2177 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **408.** | package src;  class I  {  int x;  static void test()  {  I obj = new I();  System.out.println(obj.x);  }  }  /\* Find out whether it compiles successfully are not? \*/Q2178 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **409.** | package src;  class J  {  int x;    static  {  J ref = new J();  System.out.println(ref.x);  }  }  /\* Find out whether it compiles successfully are not? \*/Q2179 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **410.** | package src;  class K  {  int i;  public static void main(String[] args)  {  K k1 = new K();  System.out.println(k1.i);  k1.i = 20;  System.out.println(k1.i);  }  }  Q2180 |
| |  | | --- | | A.  0  0 |  |  | | --- | | B.  0  20 |  |  | | --- | | C.  20  0 |  |  | | --- | | D.  Compile time error | | |

|  |  |
| --- | --- |
| **411.** | package src;  class L  {  void test()  {  System.out.println("from test");  }  public static void main(String[] args)  {  System.out.println("main begin");  L ref = new L();  ref.test();  System.out.println("-----------");  ref.test();  System.out.println("-----------");  ref.test();  System.out.println("-----------");  System.out.println("main end");  }  }  Q2181 |
| |  | | --- | | A.  Compile time error |  |  | | --- | | B.  main begin  -----------  -----------  -----------  main end |  |  | | --- | | C.  main begin  from test  -----------  from test  -----------  from test  -----------  main end |  |  | | --- | | D.  None | | |
| **412.** | package src;  class M  {  int i;  public static void main(String[] args)  {  M m1 = new M();  M m2 = m1;  System.out.println(m1.i);  System.out.println(m2.i);  m1.i = 10;  System.out.println(m1.i);  System.out.println(m2.i);  m2.i = 20;  System.out.println(m1.i);  System.out.println(m2.i);  }  }  Q2182 |
| |  | | --- | | A.  Compile time error |  |  | | --- | | B.  0  0  10  10  20  20 |  |  | | --- | | C.  0 0 10 10 20 20 |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **413.** | package src;  class N  {  int i;  public static void main(String[] args)  {  N n1 = new N();  N n2 = n1;  N n3 = n2;  N n4 = n1;  N n5 = n3;  n4.i = 20;  System.out.println(n1.i);  System.out.println(n2.i);  System.out.println(n3.i);  System.out.println(n4.i);  System.out.println(n5.i);  n2.i += 20;  System.out.println(n1.i);  System.out.println(n2.i);  System.out.println(n3.i);  System.out.println(n4.i);  System.out.println(n5.i);  }  }  Q2183 |
| |  | | --- | | A.  Compile time error |  |  | | --- | | B.  20  20  20  20  20  40  40  40  40  40 |  |  | | --- | | C.  20  20  20  20  20  20  20  20  20  20 |  |  | | --- | | D.  None | | |
| **414.** | package src;  class O  {  int i;  public static void main(String[] args)  {  O ref1 = new O();  ref1.i = 10;  O ref2 = new O();  ref2.i = 20;  O ref3 = new O();  ref3.i = 30;  System.out.println(ref1.i);  System.out.println(ref2.i);  System.out.println(ref3.i);  ref2.i = ref1.i + ref3.i;  System.out.println(ref1.i);  System.out.println(ref2.i);  System.out.println(ref3.i);  }  }  Q2184 |
| |  | | --- | | A.  Compile time error |  |  | | --- | | B.  10  20  30  10  30  40 |  |  | | --- | | C.  10  20  30  10  40  30 |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **415.** | package src;  class P  {  int i;  public static void test()  {  System.out.println("test begin");  P obj = new P();  System.out.println(obj.i);  obj.i = 10;  System.out.println(obj.i);  System.out.println("test end");  }  public static void main(String[] args)  {  System.out.println("main begin");  test();  System.out.println("main end");  }  }  Q2185 |
| |  | | --- | | A.  main begin  test begin  0  10  test end  main end |  |  | | --- | | B.  main begin  test begin  10  0  test end  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **416.** | package src;  class Q  {  int i;  public static void test(Q obj)  {  System.out.println("test begin");  System.out.println("c:" + obj.i);  obj.i = 10;  System.out.println("d:" + obj.i);  System.out.println("test end");  }  public static void main(String[] args)  {  System.out.println("main begin");  Q ref = new Q();  ref.i = 5;  System.out.println("a:" + ref.i);  test(ref);  System.out.println("b:" + ref.i);  System.out.println("main end");  }  }  Q2186 |
| |  | | --- | | A.  main begin  a:5  test begin  c:0  d:10  test end  b:5  main end |  |  | | --- | | B.  main begin  a:5  test begin  c:5  d:10  test end  b:10  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **417.** | package src;  class R  {  int i;  public static R test()  {  System.out.println("test begin");  R obj = new R();  System.out.println("c:" + obj.i);  obj.i = 10;  System.out.println("d:" + obj.i);  System.out.println("test end");  return obj;  }  public static void main(String[] args)  {  System.out.println("main begin");  R ref = test();  System.out.println("a:" + ref.i);  System.out.println("main end");  }  }  Q2187 |
| |  | | --- | | A.  main begin  test begin  c:0  d:10  test end  a:10  main end |  |  | | --- | | B.  main begin  test begin  a:0  c:0  d:10  test end  main end |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **418.** | package src;  class S  {  void test()  {  System.out.println("test:" + this);  }  public static void main(String[] args)  {  S s1 = new S();  System.out.println("main:" + s1);  s1.test();  }  }  Q2188 |
| |  | | --- | | A.  main:0  test:src.S@15db9 |  |  | | --- | | B.  main:src.S@15db9742  test:src.S@15db9742 |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **419.** | package src;  class T  {  int i;  void test()  {  System.out.println("test:" + this.i);  this.i = 20;  }  public static void main(String[] args)  {  T t1 = new T();  t1.i = 10;  System.out.println("main1:" + t1.i);  t1.test();  System.out.println("main2:" + t1.i);  }  }  Q2189 |
| |  | | --- | | A.  main1:10  test:10  main2:20 |  |  | | --- | | B.  main1:10  test:0  main2:20 |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |
| **420.** | package src;  class U  {  int i;  void test()  {  System.out.println("test:" + i);  i = 20;  }  public static void main(String[] args)  {  U obj = new U();  obj.i = 10;  System.out.println("main1:" + obj.i);  obj.test();  System.out.println("main2:" + obj.i);  }  }  Q2190 |
| |  | | --- | | A.  main1:10  test:10  main2:20 |  |  | | --- | | B.  main1:10  test:0  main2:20 |  |  | | --- | | C.  Compile time error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **421.** | package src;  class A  {  int i;  }  class B extends A  {  int j;  public static void main(String[] args)  {  B b1 = new B();  System.out.println(b1.i);  System.out.println(b1.j);  }  }  Q2347 |
| |  | | --- | | A.  0  0 |  |  | | --- | | B.  Compilation Error |  |  | | --- | | C.  Run Time Error | | |
| **422.** | package src;  class C  {  void test1()  {  System.out.println("from test1");  }  }  class D extends C  {  void test2()  {  System.out.println("from test2");  }  public static void main(String[] args)  {  D d1 = new D();  d1.test1();  d1.test2();  }  }  Q2348 |
| |  | | --- | | A.  from test1  from test2 |  |  | | --- | | B.  from test2  from test1 |  |  | | --- | | C.  Run Time Error |  |  | | --- | | D.  Compilation error | | |

|  |  |
| --- | --- |
| **423.** | package src;  class E  {  static int i;  }  class F extends E  {  static int j;  public static void main(String[] args)  {  System.out.println(F.i);  System.out.println(F.j);  }  }  Q2349 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  0  0 |  |  | | --- | | C.  Run Time Error | | |
| **424.** | package src;  class G  {  static void test1()  {  System.out.println("G-test1");  }  }  class H extends G  {  static void test2()  {  System.out.println("H-test2");  }  public static void main(String[] args)  {  H.test1();  H.test2();  }  }  Q2350 |
| |  | | --- | | A.  H-test1  G-test2 |  |  | | --- | | B.  G-test1  H-test2 |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Runtime Error | | |

|  |  |
| --- | --- |
| **425.** | package src;  class I  {  static int m;  int n;  static void test1()  {  System.out.println("from test1");  }  void test2()  {  System.out.println("from test2");  }  }  class J extends I  {  public static void main(String[] args)  {  System.out.println(J.m);  J.test1();  J ref = new J();  System.out.println(ref.n);  ref.test2();  }  }  Q2351 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  from test1  0  from test2  0 |  |  | | --- | | C.  0  from test1  0  from test2 |  |  | | --- | | D.  Runtime Error | | |
| **426.** | package src;  class K  {  K()  {  super();  System.out.println("K()");  }  }  class L extends K  {  L()  {  super();  System.out.println("L()");  }  public static void main(String[] args)  {  L obj = new L();  }  }  Q2352 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  K()  L() |  |  | | --- | | C.  L()  K() | | |

|  |  |
| --- | --- |
| **427.** | package src;  class M  {  M()  {  System.out.println("M()");  }  }  class N extends M  {  N()  {  System.out.println("N()");  }  }  class O extends N  {  public static void main(String[] args)  {  M m1 = new M();  System.out.println("---------");  N n1 = new N();  System.out.println("---------");  O o1 = new O();  System.out.println("---------");  }  }  Q2353 |
| |  | | --- | | A.  M()  ---------  M()  N()  ---------  M()  N()  --------- |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  none | | |
| **428.** | package src;  class A  {  A()  {  System.out.println("A()");  }  }  class B extends A  {  B()  {  System.out.println("B()");  }  }  class C extends B  {  C()  {  System.out.println("C()");  }  }  class P extends C  {  P()  {  System.out.println("P()");  }  public static void main(String[] args)  {  A a1 = new A();  System.out.println("---------");  B b1 = new B();  System.out.println("---------");  C c1 = new C();  System.out.println("---------");  P p1 = new P();  System.out.println("---------");  }  }  Q2354 |
| |  | | --- | | A.  ---------  ---------  ---------  P()  --------- |  |  | | --- | | B.  A()  ---------  A()  B()  ---------  A()  B()  C()  ---------  A()  B()  C()  P()  --------- |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Runtime Error | | |

|  |  |
| --- | --- |
| **429.** | class A  {  A()  {  System.out.println("A()");  }  A(int i)  {  System.out.println("A(int)");  }  }  class B extends A  {  B()  {  System.out.println("B()");  }  B(int i)  {  System.out.println("B(int)");  }  }  class C extends B  {  C()  {  System.out.println("C()");  }  C(int i)  {  super();  System.out.println("C(int)");  }  }  class Q extends C  {  Q()  {  System.out.println("Q()");  }  Q(int i)  {  super();  System.out.println("Q(int)");  }  public static void main(String[] args)  {  A a1 = new A();  System.out.println("---------");  B b1 = new B();  System.out.println("---------");  C c1 = new C();  System.out.println("---------");  Q q1 = new Q();  System.out.println("---------");  A a2 = new A(10);  System.out.println("---------");  B b2 = new B(10);  System.out.println("---------");  C c2 = new C(20);  System.out.println("---------");  Q q2 = new Q(30);  System.out.println("---------");  }  }  Q2355 |
| |  | | --- | | A.  A()  ---------  A()  B()  ---------  A()  B()  C()  ---------  A()  B()  C()  Q()  ---------  A(int)  ---------  A()  B(int)  ---------  A()  B()  C(int)  ---------  A()  B()  C()  Q(int)  --------- |  |  | | --- | | B.  A()  ---------  B()  ---------  C()  ---------  Q()  ---------  A(int)  ---------  A()  B(int)  ---------  A()  B()  C(int)  ---------  A()  B()  C()  Q(int)  --------- |  |  | | --- | | C.  A()  ---------  B()  ---------  C()  ---------  Q()  ---------  A(int)  ---------  B(int)  ---------  C(int)  ---------  Q(int)  --------- |  |  | | --- | | D.  None | | |
| **430.** | class A  {  A()  {  System.out.println("A()");  }  A(int i)  {  System.out.println("A(int)");  }  }  class B extends A  {  B()  {  super(10);  System.out.println("B()");  }  B(int i)  {  System.out.println("B(int)");  }  }  class C extends B  {  C()  {  System.out.println("C()");  }  C(int i)  {  super(i);  System.out.println("C(int)");  }  }  class R extends C  {  R()  {  super(20);  System.out.println("R()");  }  R(int i)  {  super(200);  System.out.println("R(int)");  }  public static void main(String[] args)  {  A a1 = new A();  System.out.println("---------");  B b1 = new B();  System.out.println("---------");  C c1 = new C();  System.out.println("---------");  R r1 = new R();  System.out.println("---------");  A a2 = new A(10);  System.out.println("---------");  B b2 = new B(10);  System.out.println("---------");  C c2 = new C(20);  System.out.println("---------");  R r2 = new R(30);  System.out.println("---------");  }  }  Q2356 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  Runtime Error |  |  | | --- | | C.  A()  ---------  A(int)  B()  ---------  A(int)  B()  C()  ---------  A()  B(int)  C(int)  R()  ---------  A(int)  ---------  A()  B(int)  ---------  A()  B(int)  C(int)  ---------  --------- |  |  | | --- | | D.  A()  ---------  A(int)  B()  ---------  A(int)  B()  C()  ---------  A()  B(int)  C(int)  R()  ---------  A(int)  ---------  A()  B(int)  ---------  A()  B(int)  C(int)  ---------  A()  B(int)  C(int)  R(int)  --------- | | |

|  |  |
| --- | --- |
| **431.** | class A  {  A()  {  System.out.println("A()");  }  A(int i)  {  this();  System.out.println("A(int)");  }  }  class B extends A  {  B()  {  super(10);  System.out.println("B()");  }  B(int i)  {  this();  System.out.println("B(int)");  }  }  class C extends B  {  C()  {  this(90);  System.out.println("C()");  }  C(int i)  {  super(i);  System.out.println("C(int)");  }  }  class S extends C  {  S()  {  this(90);  System.out.println("S()");  }  S(int i)  {  super(200);  System.out.println("S(int)");  }  public static void main(String[] args)  {  A a1 = new A();  System.out.println("---------");  B b1 = new B();  System.out.println("---------");  C c1 = new C();  System.out.println("---------");  S s1 = new S();  System.out.println("---------");  A a2 = new A(10);  System.out.println("---------");  B b2 = new B(10);  System.out.println("---------");  C c2 = new C(20);  System.out.println("---------");  S s2 = new S(30);  System.out.println("---------");  }  }  Q2357 |
| |  | | --- | | A.  A()  ---------  A()  A(int)  B()  ---------  A()  A(int)  B()  B(int)  C(int)  C()  ---------  A()  A(int)  B()  B(int)  C(int)  S(int)  S()  ---------  A()  A(int)  ---------  A()  A(int)  B()  B(int)  ---------  A()  A(int)  B()  B(int)  C(int)  ---------  A()  A(int)  B()  B(int)  C(int)  S(int)  --------- |  |  | | --- | | B.  A()  ---------  A()  A(int)  B()  ---------  A()  A(int)  B()  B(int)  C(int)  C()  ---------  A()  A(int)  B()  B(int)  C(int)  S(int)  S()  ---------  A()  A(int)  ---------  A()  A(int)  B()  B(int)  ---------  A()  A(int)  B()  B(int)  C(int)  ---------  --------- |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Runtime Error | | |
| **432.** | class A  {  A(int i)  {  System.out.println("A(int)");  }  }  class T extends A  {  T(int i)  {  System.out.println("T(int)");  }  public static void main(String[] args)  {  T t1 = new T(90);  System.out.println("done");  }  }  Q2358 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  T(int)  done |  |  | | --- | | C.  A(int)  T(int) |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **433.** | class A  {  A(int i)  {  System.out.println("A(int)");  }  }  class U extends A  {  U(int i)  {  super(i);  System.out.println("U(int)");  }  public static void main(String[] args)  {  U u1 = new U(90);  System.out.println("done");  }  }  Q2359 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  A(int)  U(int)  done |  |  | | --- | | C.  U(int)  A(int)  done |  |  | | --- | | D.  None | | |
| **434.** | class A  {  A()  {  System.out.println("A()");  }    {  System.out.println("A-IIB");  }  }  class V  {  public static void main(String[] args)  {  A a1 = new A();  System.out.println("----------");  A a2 = new A();  }  }  Q2360 |
| |  | | --- | | A.  A-IIB  A()  ----------  A-IIB  A() |  |  | | --- | | B.  A()  A-IIB  ----------  A()  A-IIB |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Runtime Error | | |

|  |  |
| --- | --- |
| **435.** | class A  {  A()  {  System.out.println("A()");  }    {  System.out.println("A-IIB");  }  }  class B extends A  {  B()  {  System.out.println("B()");  }    {  System.out.println("B-IIB1");  }  {  System.out.println("B-IIB2");  System.out.println("B-IIB2");  System.out.println("B-IIB2");  }  }  class W  {  public static void main(String[] args)  {  A a1 = new A();  System.out.println("----------");  A a2 = new A();  System.out.println("----------");  B b1 = new B();  System.out.println("----------");  B b2 = new B();  }  }  Q2361 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  A-IIB  A()  ----------  A-IIB  A()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B() |  |  | | --- | | C.  A()  A-IIB  ----------  A()  A-IIB  ----------  A()  A-IIB  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A()  A-IIB  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B() |  |  | | --- | | D.  None | | |
| **436.** | class A  {  A()  {  System.out.println("A()");  }  A(int i)  {  System.out.println("A(int)");  }    {  System.out.println("A-IIB");  }  }  class B extends A  {  B()  {  System.out.println("B()");  }  B(int i)  {  this();  System.out.println("B(int)");  }    {  System.out.println("B-IIB1");  }  {  System.out.println("B-IIB2");  System.out.println("B-IIB2");  System.out.println("B-IIB2");  }  }  class X  {  public static void main(String[] args)  {  A a1 = new A();  System.out.println("----------");  A a2 = new A();  System.out.println("----------");  B b1 = new B();  System.out.println("----------");  B b2 = new B();  System.out.println("----------");  A a3 = new A(20);  System.out.println("----------");  A a4 = new A(30);  System.out.println("----------");  B b3 = new B(40);  System.out.println("----------");  B b4 = new B(60);  }  }  Q2362 |
| |  | | --- | | A.  A-IIB  A()  ----------  A-IIB  A()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A-IIB  A(int)  ----------  A-IIB  A(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int) |  |  | | --- | | B.  A-IIB  A()  ----------  A()  A-IIB  ----------  A()  A-IIB  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A()  A-IIB  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A(int)  A-IIB  ----------  A(int)  A-IIB  ----------  A()  A-IIB  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  ----------  A()  A-IIB  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int) |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **437.** | class A  {  A()  {  System.out.println("A()");  }  A(int i)  {  System.out.println("A(int)");  }    {  System.out.println("A-IIB");  }  }  class B extends A  {  B()  {  System.out.println("B()");  }  B(int i)  {  this();  System.out.println("B(int)");  }    {  System.out.println("B-IIB1");  }  {  System.out.println("B-IIB2");  System.out.println("B-IIB2");  System.out.println("B-IIB2");  }  }  class C extends B  {  C()  {  super(90);  System.out.println("C()");  }  {  System.out.println("C-IIB");  }  C(int i)  {  this();  System.out.println("C(int)");  }  }  class Y  {  public static void main(String[] args)  {  A a1 = new A();  System.out.println("----------");  B b1 = new B();  System.out.println("----------");  A a2 = new A(20);  System.out.println("----------");  B b2 = new B(40);  System.out.println("----------");  C c1 = new C();  System.out.println("----------");  C c2 = new C(20);  System.out.println("----------");  }  }  Q2363 |
| |  | | --- | | A.  A-IIB  A()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A-IIB  A(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  C-IIB  C()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  C-IIB  C()  C(int)  ---------- |  |  | | --- | | B.  A-IIB  A()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A-IIB  A(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  C-IIB  C()  C(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  C-IIB  C()  ---------- |  |  | | --- | | C.  A-IIB  A()  ----------  A-IIB  A(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  C-IIB  C()  ----------  A-IIB  A()  B-IIB1  B-IIB2  B-IIB2  B-IIB2  B()  B(int)  C-IIB  C()  C(int)  ---------- |  |  | | --- | | D.  None | | |
| **438.** | class A  {  static  {  System.out.println("A-SIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  }  class Z extends C  {  static  {  System.out.println("Z-SIB");  }  public static void main(String[] args)  {  System.out.println("main");  }  }  Q2364 |
| |  | | --- | | A.  main  A-SIB  B-SIB  C-SIB  Z-SIB |  |  | | --- | | B.  A-SIB  B-SIB  C-SIB  Z-SIB  main |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **439.** | class A  {  static  {  System.out.println("A-SIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  }  class Z1  {  static  {  System.out.println("Z1-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  A a1 = new A();  System.out.println("main end");  }  }  Q2365 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  Z1-SIB  main begin  main end  A-SIB |  |  | | --- | | C.  Z1-SIB  main begin  A-SIB  main end |  |  | | --- | | D.  None | | |
| **440.** | class A  {  static  {  System.out.println("A-SIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  }  class Z2  {  static  {  System.out.println("Z2-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  A a1 = new A();  System.out.println("-----------");  B b1 = new B();  System.out.println("main end");  }  }  Q2366 |
| |  | | --- | | A.  Z2-SIB  main begin  A-SIB  -----------  B-SIB  main end |  |  | | --- | | B.  Z2-SIB  main begin  A-SIB  B-SIB  -----------  main end |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **441.** | class A  {  static  {  System.out.println("A-SIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  }  class Z3  {  static  {  System.out.println("Z3-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  B b1 = new B();  System.out.println("-----------");  A a1 = new A();  System.out.println("main end");  }  }  Q2367 |
| |  | | --- | | A.  Z3-SIB  main begin  A-SIB  -----------  B-SIB  main end |  |  | | --- | | B.  Z3-SIB  main begin  A-SIB  B-SIB  -----------  main end |  |  | | --- | | C.  None | | |
| **442.** | class A  {  static  {  System.out.println("A-SIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  }  class Z4  {  static  {  System.out.println("Z4-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  B b1 = new B();  System.out.println("-----------");  C c1 = new C();  System.out.println("-----------");  A a1 = new A();  System.out.println("main end");  }  }  Q2368 |
| |  | | --- | | A.  Z4-SIB  main begin  A-SIB  B-SIB  -----------  A-SIB  B-SIB  C-SIB  -----------  main end |  |  | | --- | | B.  Z4-SIB  main begin  B-SIB  -----------  C-SIB  -----------  A-SIB  main end |  |  | | --- | | C.  Z4-SIB  main begin  A-SIB  B-SIB  -----------  C-SIB  -----------  main end |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **443.** | class A  {  static  {  System.out.println("A-SIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  }  class Z5  {  static  {  System.out.println("Z5-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  C c1 = new C();  System.out.println("-----------");  B b1 = new B();  System.out.println("-----------");  A a1 = new A();  System.out.println("main end");  }  }  Q2369 |
| |  | | --- | | A.  Z5-SIB  main begin  A-SIB  B-SIB  C-SIB  -----------  -----------  main end |  |  | | --- | | B.  Z5-SIB  main begin  A-SIB  -----------  B-SIB  -----------  C-SIB  main end |  |  | | --- | | C.  Z5-SIB  main begin  A-SIB  -----------  A-SIB  B-SIB  -----------  A-SIB  B-SIB  C-SIB  main end |  |  | | --- | | D.  Compilation error | | |
| **444.** | class A  {  static  {  System.out.println("A-SIB");  }  A()  {  System.out.println("A()");  }    {  System.out.println("A-IIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  B()  {  System.out.println("B()");  }  {  System.out.println("B-IIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  C()  {  System.out.println("C()");  }  {  System.out.println("C-IIB");  }  }  class Z6  {  static  {  System.out.println("Z6-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  C c1 = new C();  System.out.println("-----------");  B b1 = new B();  System.out.println("-----------");  A a1 = new A();  System.out.println("main end");  }  }  Q2370 |
| |  | | --- | | A.  Z6-SIB  main begin  A-SIB  B-SIB  C-SIB  A-IIB  A()  B-IIB  B()  C-IIB  C()  -----------  A-IIB  A()  B-IIB  B()  -----------  A-IIB  A()  main end |  |  | | --- | | B.  Z6-SIB  main begin  A-IIB  A()  -----------  A-IIB  A()  B-IIB  B()  -----------  A-SIB  B-SIB  C-SIB  A-IIB  A()  B-IIB  B()  C-IIB  C()  main end |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  None | | |

|  |  |
| --- | --- |
| **445.** | // is-a  class A  {  static  {  System.out.println("A-SIB");  }  A()  {  System.out.println("A()");  }    {  System.out.println("A-IIB");  }  }  class B extends A  {  static  {  System.out.println("B-SIB");  }  B()  {  System.out.println("B()");  }  {  System.out.println("B-IIB");  }  }  class C extends B  {  static  {  System.out.println("C-SIB");  }  C()  {  System.out.println("C()");  }  {  System.out.println("C-IIB");  }  }  class Z7  {  static  {  System.out.println("Z7-SIB");  }  public static void main(String[] args)  {  System.out.println("main begin");  A a1 = new A();  System.out.println("-----------");  B b1 = new B();  System.out.println("-----------");  C c1 = new C();  System.out.println("main end");  }  }  Q2371 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  Z7-SIB  main begin  A()  A-IIB  A-SIB  -----------  B-SIB  A-IIB  A()  B-IIB  B()  -----------  C-SIB  A-IIB  A()  B-IIB  B()  C-IIB  C()  main end |  |  | | --- | | C.  Z7-SIB  main begin  A-SIB  A-IIB  A()  -----------  B-SIB  A-IIB  A()  B-IIB  B()  -----------  C-SIB  A-IIB  A()  B-IIB  B()  C-IIB  C()  main end | | |
| **446.** | class A  {  int i;  void test1()  {  System.out.println("A-test1");  }  }  class B extends A  {  void test2()  {  System.out.println("B-test2");  System.out.println(i);  test1();  }  public static void main(String[] args)  {  B b1 = new B();  b1.test2();  }  }  Q2372 |
| |  | | --- | | A.  A-test2  0  B-test2 |  |  | | --- | | B.  B-test2  0  A-test1 |  |  | | --- | | C.  Compilation Error |  |  | | --- | | D.  Runtime Error |  |  | | --- | | E.  None | | |

|  |  |
| --- | --- |
| **447.** | class A  {  int i;  void test1()  {  System.out.println("A-test1");  }  }  class C  {  A obj;  void test2()  {  System.out.println("C-test2");  System.out.println(i);  test1();  }  public static void main(String[] args)  {  C c1 = new C();  c1.test2();  }  }  Q2373 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  Runtime Error |  |  | | --- | | C.  C-test2  0  A-test1 |  |  | | --- | | D.  None | | |
| **448.** | class A  {  int i;  void test1()  {  System.out.println("A-test1");  }  }  class D  {  A obj;  void test2()  {  System.out.println("D-test2");  System.out.println(obj.i);  obj.test1();  }  public static void main(String[] args)  {  D d1 = new D();  d1.test2();  }  }  Q2374 |
| |  | | --- | | A.  D-test2  0  A-test1 |  |  | | --- | | B.  D-test2  Exception in thread "main" java.lang.NullPointerException  at D.test2(D.java:15)  at D.main(D.java:21) |  |  | | --- | | C.  A-test1  0  D-test2 |  |  | | --- | | D.  Compilation error | | |

|  |  |
| --- | --- |
| **449.** | class A  {  int i;  void test1()  {  System.out.println("A-test1");  }  }  class E  {  A obj = new A();  void test2()  {  System.out.println("E-test2");  System.out.println(obj.i);  obj.test1();  }  public static void main(String[] args)  {  E e1 = new E();  e1.test2();  }  }  Q2375 |
| |  | | --- | | A.  E-test2  0  A-test1 |  |  | | --- | | B.  A-test1  0  E-test2 |  |  | | --- | | C.  Compilation Error | | |
| **450.** | class A  {  int i;  void test1()  {  System.out.println("A-test1");  }  }  class F  {  A obj;  F(A obj)  {  this.obj = obj;  }  void test2()  {  System.out.println("F-test2");  System.out.println(obj.i);  obj.test1();  }  public static void main(String[] args)  {  A a1 = new A();  F f1 = new F(a1);  f1.test2();  F f2 = new F(new A());  f2.test2();  }  }  Q2376 |
| |  | | --- | | A.  Compilation Error |  |  | | --- | | B.  F-test2  A-test1  F-test2  A-test1 |  |  | | --- | | C.  F-test2  0  A-test1  F-test2  0  A-test1 | | |

|  |  |
| --- | --- |
| **451.** | Is it possible to to assign same reference to more than one object?Q2397 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **452.** | Same object can be refered by multiple referencesQ2398 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **453.** | Is it possible to refer same object from the different methods?Q2399 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **454.** | is it possible to create more than one object to the same class???Q2400 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **455.** | is it possible to supply reference of an object as an argument to a method while method is calling???Q2401 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **456.** | this is default reference in a main methodQ2402 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **457.** | this is default reference in a constructorQ2403 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **458.** | which key word is used to differenciate local variable and global non static variable which are having same name?Q2404 |
| |  | | --- | | A.  public |  |  | | --- | | B.  int |  |  | | --- | | C.  this |  |  | | --- | | D.  static | | |

|  |  |
| --- | --- |
| **459.** | constructor should have same name as a class nameQ2405 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **460.** | constructor should not have a return typeQ2406 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **461.** | default constructor from a compiler can be even with an arguments.Q2407 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |
| **462.** | we can use any constructor to create an object even the same constructor not available in the same classQ2408 |
| |  | | --- | | A.  Yes |  |  | | --- | | B.  No | | |

|  |  |
| --- | --- |
| **463.** | IIB is a member of what?Q2844 |
| |  | | --- | | A.  object |  |  | | --- | | B.  method |  |  | | --- | | C.  class |  |  | | --- | | D.  function | | |
| **464.** | IIB stands for what?Q2845 |
| |  | | --- | | A.  instance initialization block |  |  | | --- | | B.  instance interpreted block |  |  | | --- | | C.  instance inherited block |  |  | | --- | | D.  interpreted instance block | | |

|  |  |
| --- | --- |
| **465.** | The block which has no name,identifier,keyword is called as what?Q2846 |
| |  | | --- | | A.  SIB |  |  | | --- | | B.  IIB |  |  | | --- | | C.  abstract block |  |  | | --- | | D.  interface | | |
| **466.** | for one obect creation how many times 1 IIB will get exicuted?Q2847 |
| |  | | --- | | A.  1 |  |  | | --- | | B.  2 |  |  | | --- | | C.  3 | | |

|  |  |
| --- | --- |
| **467.** | the block which has exicuted each time for every object creation known as what?Q2848 |
| |  | | --- | | A.  SIB |  |  | | --- | | B.  interface |  |  | | --- | | C.  IIB |  |  | | --- | | D.  abstarct | | |
| **468.** | can we keep the IIB anywhere in the program?Q2849 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **469.** | can we define more than 1 IIB in a single program?Q2850 |
| |  | | --- | | A.  no |  |  | | --- | | B.  yes | | |
| **470.** | did the user require to specify explicitly for the execution of IIB's?Q2851 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |

|  |  |
| --- | --- |
| **471.** | IIB's are independent of constructer or not?Q2852 |
| |  | | --- | | A.  yes |  |  | | --- | | B.  no | | |
| **472.** | class B{  B(){  System.out.println("B()");}  {System.out.println("IIB");}  public static void main (String[] args){  System.out.println("main begin");  B obj1=new B();  System.out.println("----------");  B obj2=new B();  System.out.println("main end");}}  what will be the output?Q2853 |
| |  | | --- | | A.  main begin  IIB  B()  -------  IIB  B()  MAIN END |  |  | | --- | | B.  main begin  IIB  B()  -------  IIB  B()  main end |  |  | | --- | | C.  main begin  IIB  B()  -------  IIB  B()  MAIN END |  |  | | --- | | D.  main begin  IIB  B()  -------  B()  MAIN END | | |

[1](https://lara.co.in/#qid72)

[2](https://lara.co.in/#qid73)

[3](https://lara.co.in/#qid74)

[4](https://lara.co.in/#qid75)

[5](https://lara.co.in/#qid76)

[6](https://lara.co.in/#qid77)

[7](https://lara.co.in/#qid78)

[8](https://lara.co.in/#qid79)

[9](https://lara.co.in/#qid80)

[10](https://lara.co.in/#qid81)

[11](https://lara.co.in/#qid82)

[12](https://lara.co.in/#qid83)

[13](https://lara.co.in/#qid84)

[14](https://lara.co.in/#qid85)

[15](https://lara.co.in/#qid86)

[16](https://lara.co.in/#qid87)

[17](https://lara.co.in/#qid88)

[18](https://lara.co.in/#qid89)

[19](https://lara.co.in/#qid99)

[20](https://lara.co.in/#qid100)

[21](https://lara.co.in/#qid101)

[22](https://lara.co.in/#qid102)

[23](https://lara.co.in/#qid103)

[24](https://lara.co.in/#qid104)

[25](https://lara.co.in/#qid105)

[26](https://lara.co.in/#qid106)

[27](https://lara.co.in/#qid107)

[28](https://lara.co.in/#qid108)

[29](https://lara.co.in/#qid109)

[30](https://lara.co.in/#qid110)

[31](https://lara.co.in/#qid111)

[32](https://lara.co.in/#qid112)

[33](https://lara.co.in/#qid113)

[34](https://lara.co.in/#qid114)

[35](https://lara.co.in/#qid115)

[36](https://lara.co.in/#qid116)

[37](https://lara.co.in/#qid117)

[38](https://lara.co.in/#qid118)

[39](https://lara.co.in/#qid119)

[40](https://lara.co.in/#qid120)

[41](https://lara.co.in/#qid121)

[42](https://lara.co.in/#qid122)

[43](https://lara.co.in/#qid123)

[44](https://lara.co.in/#qid124)

[45](https://lara.co.in/#qid191)

[46](https://lara.co.in/#qid192)

[47](https://lara.co.in/#qid193)

[48](https://lara.co.in/#qid194)

[49](https://lara.co.in/#qid195)

[50](https://lara.co.in/#qid196)

[51](https://lara.co.in/#qid197)

[52](https://lara.co.in/#qid198)

[53](https://lara.co.in/#qid199)

[54](https://lara.co.in/#qid200)

[55](https://lara.co.in/#qid201)

[56](https://lara.co.in/#qid202)

[57](https://lara.co.in/#qid203)

[58](https://lara.co.in/#qid204)

[59](https://lara.co.in/#qid205)

[60](https://lara.co.in/#qid206)

[61](https://lara.co.in/#qid207)

[62](https://lara.co.in/#qid208)

[63](https://lara.co.in/#qid209)

[64](https://lara.co.in/#qid210)

[65](https://lara.co.in/#qid211)

[66](https://lara.co.in/#qid212)

[67](https://lara.co.in/#qid213)

[68](https://lara.co.in/#qid214)

[69](https://lara.co.in/#qid215)

[70](https://lara.co.in/#qid216)

[71](https://lara.co.in/#qid217)

[72](https://lara.co.in/#qid218)

[73](https://lara.co.in/#qid219)

[74](https://lara.co.in/#qid220)

[75](https://lara.co.in/#qid221)

[76](https://lara.co.in/#qid222)

[77](https://lara.co.in/#qid223)

[78](https://lara.co.in/#qid224)

[79](https://lara.co.in/#qid225)

[80](https://lara.co.in/#qid226)

[81](https://lara.co.in/#qid227)

[82](https://lara.co.in/#qid228)

[83](https://lara.co.in/#qid229)

[84](https://lara.co.in/#qid230)

[85](https://lara.co.in/#qid231)

[86](https://lara.co.in/#qid232)

[87](https://lara.co.in/#qid233)

[88](https://lara.co.in/#qid234)

[89](https://lara.co.in/#qid235)

[90](https://lara.co.in/#qid236)

[91](https://lara.co.in/#qid237)

[92](https://lara.co.in/#qid238)

[93](https://lara.co.in/#qid239)

[94](https://lara.co.in/#qid240)

[95](https://lara.co.in/#qid241)

[96](https://lara.co.in/#qid242)

[97](https://lara.co.in/#qid243)

[98](https://lara.co.in/#qid244)

[99](https://lara.co.in/#qid245)

[100](https://lara.co.in/#qid328)

[101](https://lara.co.in/#qid329)

[102](https://lara.co.in/#qid330)

[103](https://lara.co.in/#qid331)

[104](https://lara.co.in/#qid332)

[105](https://lara.co.in/#qid333)

[106](https://lara.co.in/#qid730)

[107](https://lara.co.in/#qid731)

[108](https://lara.co.in/#qid732)

[109](https://lara.co.in/#qid733)

[110](https://lara.co.in/#qid734)

[111](https://lara.co.in/#qid735)

[112](https://lara.co.in/#qid736)

[113](https://lara.co.in/#qid737)

[114](https://lara.co.in/#qid738)

[115](https://lara.co.in/#qid739)

[116](https://lara.co.in/#qid740)

[117](https://lara.co.in/#qid741)

[118](https://lara.co.in/#qid742)

[119](https://lara.co.in/#qid743)

[120](https://lara.co.in/#qid744)

[121](https://lara.co.in/#qid745)

[122](https://lara.co.in/#qid816)

[123](https://lara.co.in/#qid989)

[124](https://lara.co.in/#qid990)

[125](https://lara.co.in/#qid991)

[126](https://lara.co.in/#qid992)

[127](https://lara.co.in/#qid993)

[128](https://lara.co.in/#qid994)

[129](https://lara.co.in/#qid995)

[130](https://lara.co.in/#qid996)

[131](https://lara.co.in/#qid997)

[132](https://lara.co.in/#qid998)

[133](https://lara.co.in/#qid999)

[134](https://lara.co.in/#qid1000)

[135](https://lara.co.in/#qid1001)

[136](https://lara.co.in/#qid1002)

[137](https://lara.co.in/#qid1003)

[138](https://lara.co.in/#qid1004)

[139](https://lara.co.in/#qid1005)

[140](https://lara.co.in/#qid1006)

[141](https://lara.co.in/#qid1007)

[142](https://lara.co.in/#qid1008)

[143](https://lara.co.in/#qid1009)

[144](https://lara.co.in/#qid1010)

[145](https://lara.co.in/#qid1011)

[146](https://lara.co.in/#qid1012)

[147](https://lara.co.in/#qid1013)

[148](https://lara.co.in/#qid1014)

[149](https://lara.co.in/#qid1015)

[150](https://lara.co.in/#qid1016)

[151](https://lara.co.in/#qid1017)

[152](https://lara.co.in/#qid1018)

[153](https://lara.co.in/#qid1019)

[154](https://lara.co.in/#qid1020)

[155](https://lara.co.in/#qid1021)

[156](https://lara.co.in/#qid1022)

[157](https://lara.co.in/#qid1023)

[158](https://lara.co.in/#qid1024)

[159](https://lara.co.in/#qid1025)

[160](https://lara.co.in/#qid1026)

[161](https://lara.co.in/#qid1027)

[162](https://lara.co.in/#qid1028)

[163](https://lara.co.in/#qid1029)

[164](https://lara.co.in/#qid1030)

[165](https://lara.co.in/#qid1031)

[166](https://lara.co.in/#qid1032)

[167](https://lara.co.in/#qid1033)

[168](https://lara.co.in/#qid1034)

[169](https://lara.co.in/#qid1035)

[170](https://lara.co.in/#qid1036)

[171](https://lara.co.in/#qid1037)

[172](https://lara.co.in/#qid1038)

[173](https://lara.co.in/#qid1039)

[174](https://lara.co.in/#qid1040)

[175](https://lara.co.in/#qid1041)

[176](https://lara.co.in/#qid1042)

[177](https://lara.co.in/#qid1043)

[178](https://lara.co.in/#qid1044)

[179](https://lara.co.in/#qid1045)

[180](https://lara.co.in/#qid1046)

[181](https://lara.co.in/#qid1047)

[182](https://lara.co.in/#qid1048)

[183](https://lara.co.in/#qid1049)

[184](https://lara.co.in/#qid1050)

[185](https://lara.co.in/#qid1051)

[186](https://lara.co.in/#qid1052)

[187](https://lara.co.in/#qid1053)

[188](https://lara.co.in/#qid1054)

[189](https://lara.co.in/#qid1055)

[190](https://lara.co.in/#qid1056)

[191](https://lara.co.in/#qid1057)

[192](https://lara.co.in/#qid1058)

[193](https://lara.co.in/#qid1059)

[194](https://lara.co.in/#qid1060)

[195](https://lara.co.in/#qid1061)

[196](https://lara.co.in/#qid1062)

[197](https://lara.co.in/#qid1063)

[198](https://lara.co.in/#qid1064)

[199](https://lara.co.in/#qid1065)

[200](https://lara.co.in/#qid1066)

[201](https://lara.co.in/#qid1067)

[202](https://lara.co.in/#qid1068)

[203](https://lara.co.in/#qid1069)

[204](https://lara.co.in/#qid1070)

[205](https://lara.co.in/#qid1071)

[206](https://lara.co.in/#qid1072)

[207](https://lara.co.in/#qid1073)

[208](https://lara.co.in/#qid1074)

[209](https://lara.co.in/#qid1075)

[210](https://lara.co.in/#qid1076)

[211](https://lara.co.in/#qid1077)

[212](https://lara.co.in/#qid1078)

[213](https://lara.co.in/#qid1079)

[214](https://lara.co.in/#qid1080)

[215](https://lara.co.in/#qid1081)

[216](https://lara.co.in/#qid1082)

[217](https://lara.co.in/#qid1083)

[218](https://lara.co.in/#qid1084)

[219](https://lara.co.in/#qid1085)

[220](https://lara.co.in/#qid1086)

[221](https://lara.co.in/#qid1087)

[222](https://lara.co.in/#qid1088)

[223](https://lara.co.in/#qid1224)

[224](https://lara.co.in/#qid1225)

[225](https://lara.co.in/#qid1226)

[226](https://lara.co.in/#qid1227)

[227](https://lara.co.in/#qid1228)

[228](https://lara.co.in/#qid1229)

[229](https://lara.co.in/#qid1230)

[230](https://lara.co.in/#qid1231)

[231](https://lara.co.in/#qid1232)

[232](https://lara.co.in/#qid1233)

[233](https://lara.co.in/#qid1234)

[234](https://lara.co.in/#qid1235)

[235](https://lara.co.in/#qid1236)

[236](https://lara.co.in/#qid1237)

[237](https://lara.co.in/#qid1238)

[238](https://lara.co.in/#qid1239)

[239](https://lara.co.in/#qid1240)

[240](https://lara.co.in/#qid1241)

[241](https://lara.co.in/#qid1242)

[242](https://lara.co.in/#qid1243)

[243](https://lara.co.in/#qid1244)

[244](https://lara.co.in/#qid1245)

[245](https://lara.co.in/#qid1246)

[246](https://lara.co.in/#qid1247)

[247](https://lara.co.in/#qid1248)

[248](https://lara.co.in/#qid1249)

[249](https://lara.co.in/#qid1250)

[250](https://lara.co.in/#qid1251)

[251](https://lara.co.in/#qid1252)

[252](https://lara.co.in/#qid1253)

[253](https://lara.co.in/#qid1254)

[254](https://lara.co.in/#qid1255)

[255](https://lara.co.in/#qid1256)

[256](https://lara.co.in/#qid1257)

[257](https://lara.co.in/#qid1258)

[258](https://lara.co.in/#qid1259)

[259](https://lara.co.in/#qid1260)

[260](https://lara.co.in/#qid1261)

[261](https://lara.co.in/#qid1262)

[262](https://lara.co.in/#qid1263)

[263](https://lara.co.in/#qid1264)

[264](https://lara.co.in/#qid1265)

[265](https://lara.co.in/#qid1266)

[266](https://lara.co.in/#qid1267)

[267](https://lara.co.in/#qid1268)

[268](https://lara.co.in/#qid1269)

[269](https://lara.co.in/#qid1270)

[270](https://lara.co.in/#qid1271)

[271](https://lara.co.in/#qid1272)

[272](https://lara.co.in/#qid1273)

[273](https://lara.co.in/#qid1274)

[274](https://lara.co.in/#qid1275)

[275](https://lara.co.in/#qid1276)

[276](https://lara.co.in/#qid1277)

[277](https://lara.co.in/#qid1278)

[278](https://lara.co.in/#qid1279)

[279](https://lara.co.in/#qid1280)

[280](https://lara.co.in/#qid1281)

[281](https://lara.co.in/#qid1282)

[282](https://lara.co.in/#qid1283)

[283](https://lara.co.in/#qid1284)

[284](https://lara.co.in/#qid1285)

[285](https://lara.co.in/#qid1286)

[286](https://lara.co.in/#qid1287)

[287](https://lara.co.in/#qid1288)

[288](https://lara.co.in/#qid1420)

[289](https://lara.co.in/#qid1421)

[290](https://lara.co.in/#qid1422)

[291](https://lara.co.in/#qid1757)

[292](https://lara.co.in/#qid1758)

[293](https://lara.co.in/#qid1759)

[294](https://lara.co.in/#qid1760)

[295](https://lara.co.in/#qid1761)

[296](https://lara.co.in/#qid1762)

[297](https://lara.co.in/#qid1763)

[298](https://lara.co.in/#qid1764)

[299](https://lara.co.in/#qid1765)

[300](https://lara.co.in/#qid1766)

[301](https://lara.co.in/#qid1767)

[302](https://lara.co.in/#qid1768)

[303](https://lara.co.in/#qid1769)

[304](https://lara.co.in/#qid1770)

[305](https://lara.co.in/#qid1771)

[306](https://lara.co.in/#qid1772)

[307](https://lara.co.in/#qid1773)

[308](https://lara.co.in/#qid1774)

[309](https://lara.co.in/#qid1775)

[310](https://lara.co.in/#qid1776)

[311](https://lara.co.in/#qid1777)

[312](https://lara.co.in/#qid1778)

[313](https://lara.co.in/#qid1779)

[314](https://lara.co.in/#qid1780)

[315](https://lara.co.in/#qid1781)

[316](https://lara.co.in/#qid1782)

[317](https://lara.co.in/#qid1783)

[318](https://lara.co.in/#qid1784)

[319](https://lara.co.in/#qid1785)

[320](https://lara.co.in/#qid1786)

[321](https://lara.co.in/#qid1787)

[322](https://lara.co.in/#qid1788)

[323](https://lara.co.in/#qid1789)

[324](https://lara.co.in/#qid1790)

[325](https://lara.co.in/#qid1791)

[326](https://lara.co.in/#qid1792)

[327](https://lara.co.in/#qid1793)

[328](https://lara.co.in/#qid1794)

[329](https://lara.co.in/#qid1795)

[330](https://lara.co.in/#qid1796)

[331](https://lara.co.in/#qid1797)

[332](https://lara.co.in/#qid1798)

[333](https://lara.co.in/#qid1799)

[334](https://lara.co.in/#qid1800)

[335](https://lara.co.in/#qid1801)

[336](https://lara.co.in/#qid1802)

[337](https://lara.co.in/#qid1803)

[338](https://lara.co.in/#qid1804)

[339](https://lara.co.in/#qid1805)

[340](https://lara.co.in/#qid1806)

[341](https://lara.co.in/#qid1807)

[342](https://lara.co.in/#qid1808)

[343](https://lara.co.in/#qid1809)

[344](https://lara.co.in/#qid1810)

[345](https://lara.co.in/#qid1811)

[346](https://lara.co.in/#qid1812)

[347](https://lara.co.in/#qid1813)

[348](https://lara.co.in/#qid1814)

[349](https://lara.co.in/#qid1815)

[350](https://lara.co.in/#qid1816)

[351](https://lara.co.in/#qid1817)

[352](https://lara.co.in/#qid1818)

[353](https://lara.co.in/#qid1819)

[354](https://lara.co.in/#qid1820)

[355](https://lara.co.in/#qid1821)

[356](https://lara.co.in/#qid1822)

[357](https://lara.co.in/#qid1823)

[358](https://lara.co.in/#qid1824)

[359](https://lara.co.in/#qid1825)

[360](https://lara.co.in/#qid1826)

[361](https://lara.co.in/#qid1827)

[362](https://lara.co.in/#qid1828)

[363](https://lara.co.in/#qid1829)

[364](https://lara.co.in/#qid1830)

[365](https://lara.co.in/#qid1831)

[366](https://lara.co.in/#qid1832)

[367](https://lara.co.in/#qid1833)

[368](https://lara.co.in/#qid1834)

[369](https://lara.co.in/#qid1837)

[370](https://lara.co.in/#qid1838)

[371](https://lara.co.in/#qid1839)

[372](https://lara.co.in/#qid1840)

[373](https://lara.co.in/#qid1841)

[374](https://lara.co.in/#qid1842)

[375](https://lara.co.in/#qid1843)

[376](https://lara.co.in/#qid1844)

[377](https://lara.co.in/#qid1845)

[378](https://lara.co.in/#qid1846)

[379](https://lara.co.in/#qid1907)

[380](https://lara.co.in/#qid1908)

[381](https://lara.co.in/#qid1909)

[382](https://lara.co.in/#qid1910)

[383](https://lara.co.in/#qid1911)

[384](https://lara.co.in/#qid1912)

[385](https://lara.co.in/#qid1913)

[386](https://lara.co.in/#qid1914)

[387](https://lara.co.in/#qid1915)

[388](https://lara.co.in/#qid1916)

[389](https://lara.co.in/#qid1917)

[390](https://lara.co.in/#qid1918)

[391](https://lara.co.in/#qid1919)

[392](https://lara.co.in/#qid1920)

[393](https://lara.co.in/#qid1921)

[394](https://lara.co.in/#qid1922)

[395](https://lara.co.in/#qid1923)

[396](https://lara.co.in/#qid1924)

[397](https://lara.co.in/#qid1925)

[398](https://lara.co.in/#qid1926)

[399](https://lara.co.in/#qid2169)

[400](https://lara.co.in/#qid2170)

[401](https://lara.co.in/#qid2171)

[402](https://lara.co.in/#qid2172)

[403](https://lara.co.in/#qid2173)

[404](https://lara.co.in/#qid2174)

[405](https://lara.co.in/#qid2175)

[406](https://lara.co.in/#qid2176)

[407](https://lara.co.in/#qid2177)

[408](https://lara.co.in/#qid2178)

[409](https://lara.co.in/#qid2179)

[410](https://lara.co.in/#qid2180)

[411](https://lara.co.in/#qid2181)

[412](https://lara.co.in/#qid2182)

[413](https://lara.co.in/#qid2183)

[414](https://lara.co.in/#qid2184)

[415](https://lara.co.in/#qid2185)

[416](https://lara.co.in/#qid2186)

[417](https://lara.co.in/#qid2187)

[418](https://lara.co.in/#qid2188)

[419](https://lara.co.in/#qid2189)

[420](https://lara.co.in/#qid2190)

[421](https://lara.co.in/#qid2347)

[422](https://lara.co.in/#qid2348)

[423](https://lara.co.in/#qid2349)

[424](https://lara.co.in/#qid2350)

[425](https://lara.co.in/#qid2351)

[426](https://lara.co.in/#qid2352)

[427](https://lara.co.in/#qid2353)

[428](https://lara.co.in/#qid2354)

[429](https://lara.co.in/#qid2355)

[430](https://lara.co.in/#qid2356)

[431](https://lara.co.in/#qid2357)

[432](https://lara.co.in/#qid2358)

[433](https://lara.co.in/#qid2359)

[434](https://lara.co.in/#qid2360)

[435](https://lara.co.in/#qid2361)

[436](https://lara.co.in/#qid2362)

[437](https://lara.co.in/#qid2363)

[438](https://lara.co.in/#qid2364)

[439](https://lara.co.in/#qid2365)

[440](https://lara.co.in/#qid2366)

[441](https://lara.co.in/#qid2367)

[442](https://lara.co.in/#qid2368)

[443](https://lara.co.in/#qid2369)

[444](https://lara.co.in/#qid2370)

[445](https://lara.co.in/#qid2371)

[446](https://lara.co.in/#qid2372)

[447](https://lara.co.in/#qid2373)

[448](https://lara.co.in/#qid2374)

[449](https://lara.co.in/#qid2375)

[450](https://lara.co.in/#qid2376)

[451](https://lara.co.in/#qid2397)

[452](https://lara.co.in/#qid2398)

[453](https://lara.co.in/#qid2399)

[454](https://lara.co.in/#qid2400)

[455](https://lara.co.in/#qid2401)

[456](https://lara.co.in/#qid2402)

[457](https://lara.co.in/#qid2403)

[458](https://lara.co.in/#qid2404)

[459](https://lara.co.in/#qid2405)

[460](https://lara.co.in/#qid2406)

[461](https://lara.co.in/#qid2407)

[462](https://lara.co.in/#qid2408)

[463](https://lara.co.in/#qid2844)

[464](https://lara.co.in/#qid2845)

[465](https://lara.co.in/#qid2846)

[466](https://lara.co.in/#qid2847)

[467](https://lara.co.in/#qid2848)

[468](https://lara.co.in/#qid2849)

[469](https://lara.co.in/#qid2850)

[470](https://lara.co.in/#qid2851)