Java Question with Answer:-

------------------------------------

**Note:**

* **All Questions are based on Java 7 or earlier versions.**
* **Questions are having three level as Beginner, Intermediate and Complex.**

|  |
| --- |
| Question : 1 Level : Beginner |
| **Question: What is the exact output of this code?**    class A {    }    public class B{  void m1(){  System.out.println("This is method of Class B");  }  }    public class C{  public static void main(String[] args){  B objB = new B();  System.out.print("This is Class C");  objB.m1();    }  }    **Output :-**     1. **This is method of Class B** 2. **This is Class C.** 3. **This is Class C, This is method of Class B.** 4. **Compilation Error.**   **Answer : Option C**  **Explanation : It first executes main class statements and then the objB call the method m1() then it prints.** |

|  |
| --- |
| Question : 2 Level : Beginner |
| **Question: What is the output of this code?**    **Note: Save this code as GlobalClass.java, Compile it and execute it.**    class A {  public static void main(String[] args) {  System.out.print("This is Class A");  }  }    class B {  public static void main(String[] args) {  System.out.print("This is Class B");  }  }    class C {  public static void main(String[] args) {  System.out.print("This is Class C");  }  }    class D {    }    **Output :-**     1. **In a Class, Cannot be define more than one Main method.** 2. **Code successfully compile and Execute.** 3. **NoClassDefFoundError.** 4. **None of the above.**   **Answer : Option D**  **Explanation : Could not find or load main class GlobalClass.** |

|  |
| --- |
| Question : 3 Level : Intermediate |
| **Question: What is the output of this code?**    public class DemoTestArrays { public static void main(String[] args) { int arrOne[] = { 1, 2, 3, 4, 5 };  int arrTwo[] = { 0, 0, 0, 0, 0 };    for (int i = 0; i < arrOne.length; i++) {  arrTwo[i] = arrOne[arrOne.length - i - 1];  }    System.out.println(Arrays.toString(arrTwo));  }  }    **Output :-**     1. **[0, 0, 0, 0, 0].** 2. **[5, 4, 3, 2, 1].** 3. **[1, 2, 3, 4, 5].** 4. **Runtime Error.**   **Answer : CompileTimeError**  **Explanation : Cannot find symbol in system.out.println , variable Arrays.** |

|  |
| --- |
| Question : 4 Level : Intermediate |
| **Question: What is the output of this code?**    public class DemoTestClass {  public static void main(String[] args) {    String[] elements = { "AAA", "BBB", "CCC" };  String first = (elements.length > 0) ? elements[0] : null;  System.out.println(first);  }  }    **Output :-**     1. **BBB.** 2. **CCC.** 3. **AAA.** 4. **Runtime Error.**   **Answer : Option C**    **Explanation : In the “first” variable, we check element length is greater than zero. The Statement is true, so element[0] is printed. That is “AAA”.** |

|  |
| --- |
| Question : 5 Level : Intermediate |
| **Question: Is there a destructor for Java?**       1. **No, Because Java is a garbage collected language, you cannot predict when (or even if) an object will be destroyed.**      1. **Yes, Java is quite mature as a language and memory leak can be fixed.**        1. **Java objects are heap allocated and garbage collected, that's why destructor used in java.**      1. **None of the above.**   **Answer : Option A**  **Explanation : That there is no concept of destructor in java. In place of the destructor , java provides the garbage collector that works the same as the destructor. The garbage collector is a program (thread) that runs on the JVM.** |

Question : 6 Level : Beginner

|  |
| --- |
| **Question: Read carefully below code and identify the correct answer?**    public class ClassMain {    public static void main(String[] args) {  String main = "main is incorrect defined";  System.out.println(main);  }  }         1. **Yes, it compiles and execute because, the character sequence "main" is an identifier.**      1. **No, because main is a keyword/reserve word in java.**      1. **It does not compile.**      1. **In Java, Main keyword is not used twice.**   **Answer : Option A**  **Explanation : Yes, it compiles and execute because, the character sequence “main” is an identifier.** |

|  |
| --- |
| Question : 7 Level : Beginner |
| **Question: Read the given below code and identify correct Output?**    class MyProgram {  int count = 0;    public static void main(String[] args) {  System.out.println(count);  }  }    **Output :-**     1. **null.** 2. **0.** 3. **Error.** 4. **None of the above.**   **Answer : Option C**  **Explanation : A global variable cannot be accessed in static method.**  **Non-static variable count cannot be referenced from a static context**  **.** |

Question : 8 Level : Beginner

|  |
| --- |
| **Question: How many Objects created in the below code?**    class X {  X() {  System.out.println(this.hashCode());  }  }    class Y extends X {  Y() {  System.out.println(this.hashCode());  }  }    public class TestClass {  public static void main(String[] args) {  Y y = new Y();  System.out.println(y.hashCode());  }  }      **Output :-**     1. **3.** 2. **2.** 3. **1.** 4. **None of the above.**   **Answer : Option C**  **Explanation : Because only one object is created for class y in the main class .** |

|  |
| --- |
| Question : 9 Level : Intermediate |
| **Question: What is the correct output of the given code?**    public class Test {  public static double calculation(double a, double b) {  if (a == b) {  return 0; } else {  return 2 / (a - b);  }  }    public static void main(String[] args) {  double d1 = Double.MIN\_VALUE;  double d2 = 2.0 \* Double.MIN\_VALUE;  System.out.println("Result: " + calculation(d1, d2));  }  }    **Output :-**     1. **0.0** 2. **0** 3. **Error** 4. **-Infinity**   **Answer : Option D**  **Explanation :** |

|  |
| --- |
| Question : 10 Level : Intermediate |
| **Question: What is the correct answer of the below code?**    public class Test {  public static void main(String[] args) {  int j = 0;  if ((8 > 4) | (j++ == 7))  System.out.println("j = " + j);  }  }      **Output :-**     1. **0** 2. **1** 3. **2** 4. **ArithmeticException (Divided by zero)**   **Answer : Option B**  **Explanation : || condition returns statement when any one condition gets true , so it returns j value increament by 1 by using post increament operator.** |

|  |
| --- |
| Question : 11 Level : Beginner |
| **Question: What is the output of below code?**    public class Test {  public static void main(String[] args) { int[] array = { 1, 2, 3, 4, 5 };    int sum = 0;    for (int i : array)  sum += ++i;    System.out.println(--sum);  }  }      **Output :-**     1. **15** 2. **16** 3. **20 D. 19**   **Answer : Option D**  **Explanation : In for loop i is accessing all the array elements and adding them in ++i. So, the value is increament 1 in all the elements in the array. And the sum will be 20. In the final**  **Value it prints –sum so the answer will be 19.** |

|  |
| --- |
| Question : 12 Level : Beginner |
| **Question: Find Out the correct output of the given code?**    public class MathTest {    public void main(String[] args) {    int x = 10 \* 10 - 10;    System.out.println(++x);  }    }        **Output :-**     1. **0** 2. **90** 3. **91** 4. **Runtime Error**   **Answer : Option D**  **Explanation : Main method is not static in class.**  **Main method is not static in class Test, please define the main method as :**    **public static void main(String[] args)** |

|  |
| --- |
| Question : 13 Level : Beginner |
| **Question:**  **Can we create a user defined immutable class, pick the correct option?**      **Output :-**     1. **Make the class as final and** 2. **Make the data members as private and final.** 3. **Both A and B are Correct** 4. **None of the above**   **Answer : Option D**  **Explanation :**  **Make all mutable fields final so that its value can be assigned only once** |

|  |
| --- |
| Question : 14 Level : Beginner |
| **Question:**  **How to define Vector class??**      **Output :-**    **A. Synchronized and Non-serialized B. Non-Synchronized and Serialized.**   1. **Both A and B are Correct** 2. **None of the above**   **Answer : Option D**  **Explanation :**  **The vector class is Synchronized i.e., one thread is working on the vector,**  **So other thread can get a hold of it. And vector is Serialized.**  **i.e., Vector<String> vec = new Vector<String>();** |

|  |
| --- |
| Question : 15 Level : Beginner |
|  |
| **Question:**  **What is the output of the below code?**    public class TestString1 {  public static void main(String[] args) {  String str = "420"; str += 42;  System.out.print(str);  }  }      **Output :-**    **A. 420**  **B. 42042.**   1. **Compilation fails** 2. **An exception is thrown at runtime**   **Answer : Option B**  **Explanation :**  **str=”420”+42;**  **When string is added with integer , the integer value is considered**  **as String.**  **So, the output will be 42042.** |

|  |
| --- |
| Question : 16 Level : Beginner |
| **Question:**  **What is the output of the below code?**    class Test {  public static void main(String[] args) {  int x = 0; int y = 10;  do { y--; ++x;  } while (x < 5);  System.out.print(x + "," + y);  }  }      **Output :-**    **A. 5, 6**  **B. 5, 5.**   1. **6, 5** 2. **Error**   **Answer : Output**  **Explanation :**  **Y-- decreament the value and ++X increases the value until while loop condition gets terminated.** |

|  |
| --- |
| Question : 17 Level : Beginner |
| **Question:**  **What is the output of the below code?**    class Test {  public static void main(String[] args) {  int x = 0; int y = 10;  do { y--; ++x;  } while (x < 5);  System.out.print(x + "," + y);  }  }      **Output :-**    **A. 5, 6**  **B. 5, 5.**   1. **6, 5** 2. **Error**   **Answer : Output B**  **Explanation :**  **Y-- decreament the value and ++X increases the value until while loop condition gets terminated.** |

|  |
| --- |
| Question : 18 Level : Beginner |
| **Question:**  **What definition exactly match for abstract class?**      **Output :-**     1. **public abstract class A {**   **public Bark speak();**  **}**     1. **public abstract class A {**   **public Bark speak() {**  **}**  **}**     1. **public class A { public abstract Bark speak();**   **}**     1. **public class A abstract{ public abstract Bark speak();**   **}**  **Answer : A**  **Explanation : The abstract keyword can be declared before class name and method name is the right format of creating abstract class and method .** |

|  |
| --- |
| Question : 19 Level : Beginner |
| **Question:**  **Read the below code and pick correct option?**    class LoopTestDemo {  public static void main(String[] args) {  int x = 12; while (x < 10) {  x--;  }  System.out.print(x);  }  }      **Output :-**     1. **11** 2. **10** 3. **12** 4. **9**     **Answer : Option C**  **Explanation :**  **While (12<10) so it exists the loop and prints x value.**  **i.e.,12.** |

Question : 20 Level : Beginner

|  |
| --- |
| **Question:**  **Read the below code and pick correct option?**    class BitwiseTestDemo {  public static void main(String[] args) {  int x = 5; int y = 7;  System.out.print(((y \* 2) % x));  System.out.print(" " + (y % x));  }  }      **Output :-**     1. **6, 8** 2. **7, 9** 3. **4, 6 D. 4, 2**   **Answer : Option D**  **Explanation : The % Oprator returns remainder of a division operation**. |

|  |
| --- |
| Question : 21 Level : Intermediate |
| **Question:**  **Read the below code and pick correct option?**    class TestFormatSpecifier {    static final long num = 343L;    static long testMethod(long num) {  System.out.print(++num + " "); return ++num;  }    public static void main(String[] args) { System.out.print(num + " "); final long num = 340L;  new TestString1().testMethod(num);  System.out.println(num);  }  }    **Output :-**     1. **343 340 342** 2. **343 341 342** 3. **343 341 340** 4. **An exception is thrown at runtime**   **Answer CompileTimeError**  **Explanation :**  **Can’t find symbol new TestString1().testMethod(num)** |

|  |
| --- |
| Question : 22 Level : Intermediate |
| **Question:**  **Read the below code and pick correct option?**    public class TestBooleanDemo {  public static void main(String[] args) {  int x = 5; boolean b1 = true;  boolean b2 = false;    if ((x == 4) && !b2) System.out.print("1 "); System.out.print("2 "); if ((b2 = true) && b1)  System.out.print("3 ");  }  }      **Output :-**     1. **2, 3** 2. **1, 2** 3. **3, 2** 4. **An exception is thrown at runtime**   **Answer : Option A**  **Explanation : In the first condition loop excute the else part i.e.,2 and in the second**  **Condition it excutes the condition and the answer is 3. Therefore the answer will be 2, 3.** |

|  |
| --- |
| Question : 23 Level : Intermediate |
| **Question:**  **Read the below code and pick correct option?**    public class Test {  public void main(String[] args) {  int x = 6;  Test test = new Test(); test.doSomething(x);  System.out.print(" main x = " + x);  }    void doSomething(int x) {  System.out.print(" method x = " + x++);  }  }    **Output :-**     1. **An exception is thrown at runtime** 2. **method x = 6, main x = 6** 3. **method x = 6 main x = 7 D. method x = 7 main x = 6**   **Answer : Option A**  **Explanation : Main method is not static in class Test. Please define main method as**  **Public static void main(String []args)** |

|  |
| --- |
| Question : 24 Level : Intermediate |
| **Question:**  **Read the below code and pick correct option?**    class TernanryTestDemo {  public static void main(String[] args) {  int i = 42;  String str = (i < 40) ? "Computer" : (i > 50) ? "Java" : "Everything";  System.out.println(str);  }  }      **Output :-**     1. **An exception is thrown at runtime** 2. **Computer** 3. **Java** 4. **Everything**   **Answer : Option D**  **Explanation : The condition checks (42<40) it is false. Then switch to the next condition**  **(42>50), which is also false.**  **Therefore the output prints the else condition i.e. Everything.** |

|  |
| --- |
| Question : 25 Level : Intermediate |
| **Question:**  **Read the below code and pick correct option?**    class TernanryTestDemo {  public static void main(String[] args) {  int i = 42;  String str = (i < 40) ? "Computer" : (i > 50) ? "Java" : "Everything";  System.out.println(str);  }  }      **Output :-**     1. **An exception is thrown at runtime** 2. **Computer** 3. **Java** 4. **Everything**   **Answer : Option D**  **Explanation : The condition checks (42<40) it is false. Then switch to the next condition (42>50), which is also false.**  **Therefore the output prints the else condition. i.e.Everything.** |

|  |
| --- |
| Question : 26 Level : Beginner |
| **Question:**  **Read the below code and pick correct option?**    class ExceptionTestDemo {  public static void main(String[] args) {    Float valuePie = new Float(3.14f);  try {  if (valuePie > 3)  System.out.print("Pie value is greater than 3"+", ");    else  System.out.print("Pie value is not greater than 3"+", ");  } catch (Exception e) {  e.printStackTrace();  } finally {  System.out.println ("Have a nice day.");  }  }  }    **Output :-**     1. **Pie value is not greater than 3, Have a nice day.** 2. **Pie value is greater than 3, Have a nice day.** 3. **Pie value is not greater than 3.** 4. **An exception is thrown at runtime.**   **Answer : Option B**  **Explanation : First try condition is executed (3.14>3) it prints “pie value is greater than 3”.**  **And catch is not executed because whenever an exception occurs in try block catch is executed that time.**  **Final condition is always executed. Therefore it prints “Have a nice day”.** |

|  |
| --- |
| Question : 27 Level : Beginner |
| **Question:**  **Read the below code and pick correct option?**    class TernaryDemo {  public static void main(String[] args) {    int a = 8;  System.out.println ("" + (int) ((a < 8) ? 9.9 : 9));  }  }  **Output :-**     1. **9.9** 2. **0.** 3. **9.** 4. **Error.**   **Answer : Option C**  **Explanation :**  **It checks the condition (8<8) it fails.**  **Therefore it executes the else part and prints 9.** |

|  |
| --- |
| Question : 28 Level : Beginner |
| **Question:**  **Read the below code and pick correct option?**    class TestDoubleDemo {    public static long round(double a) {  if (a != 0x1.fffffffffffffp-2) {    return (long)Math.floor(a + 0.5d);  } else {  return 0;  }  }  public static void main(String[] args) {  TestDoubleDemo t = new TestDoubleDemo(); t.round(2.5);  }  }    **Output :-**     1. **3** 2. **0.** 3. **-1.** 4. **None of the above.**   **Answer : Option D**  **Explanation : The object t is created and class the round methods. It checks if condition,**  **and it is true and none of them display in the output.**  **Because we didn’t import math class.** |

|  |
| --- |
| Question : 29 Level : Beginner |
| **Question:**  **Create a parent class as below**    class A {  private int a = 0;    }    Which one is tightly encapsulated in the below options    **Output :-**     1. **class B extends A {**   **int a = 0;**  **}**     1. **class C extends A { private int a = 0;**   **}**     1. **class B extends A {**   **static int a = 0;**  **}**     1. **class C extends A {**   **final int a = 0;**  **}**    **Answer : Option B**  **Explanation :**  **If all the data members in class are declared as private and if it’s inherited by another class which too has all private data members.**  **Then it is called tightly encapsulated.** |

|  |
| --- |
| Question : 30 Level : Beginner |
| **Question:**  **Cyclic inheritance allowed in Java or Not??**    class A extends B {  // some methods  }    class B extends A {  // some methods  }       1. **No, Not Allowed.**      1. **Yes, Definitely Allowed.**      1. **With Some condition, Allowed**      1. **None of the Above**   **Answer : Option A**  **Explanation :**  **Cyclic inheritance is simply not possible. It defines an infinite recursive class.**  **For example, in order to define class C, you need class A, to define class A you need class B, and to define class B you need class C – and you are back to the starting point. This goes on infinitely, so compiler can’t do this and it also has no logical meaning.** |

|  |
| --- |
| Question : 31 Level : Beginner |
| **Question:**  **Read the below code and find correct output?**    public class Main {    public static void main(String[] args) { Integer x = 400, y = 400; if (x == y)  System.out.println("Number is Same");  else  System.out.println("Number is Not Same");  }  }     1. **Number is Same**      1. **Number is Not Same**      1. **Runtime Exception**      1. **None of the Above**   **Answer : Option B**  **Explanation :** |