**Var-arg methods** (Variable no of methods)

Until 1.4 we can’t declare a method with variable number of arguments if there is a change in no of arguments, we should go for new method it increases length of the code and reduces readability. To overcome this problem sun people introduced var-arg methods in 1.5 version, according to this we can declare a method which can take variable number of arguments, such type of methods are called var-arg methods.

We can declare a var-arg method as follows

Ex: m1(int...x){ }

We can call this method by passing any number of int values including zero number

Ex m1();m1(10);m2(10, 20);m1(10,20,30);

|  |
| --- |
| **public** **class** VarArgMethod {  **public** **static** **void** main(String[] args) {  VarArgMethod objRef = **new** VarArgMethod();  objRef.m1();  objRef.m1(10);  objRef.m1(10, 20);  objRef.m1(10, 20, 30);  }  **public** **void** m1(**int**... x) {  System.***out***.println("var arg method");  }  } |

Internally var-arg parameter will be converted into one dimensions array hence within the method we can differentiate values by using index.

|  |
| --- |
| **public** **class** VarArgMethodArray {  **public** **static** **void** main(String[] args) {  VarArgMethodArray objRef = **new** VarArgMethodArray();  objRef.add();  objRef.add(10);  objRef.add(10, 20);  objRef.add(10, 20, 30);  }  **public** **void** add(**int**... array) {  **int** result=0;  **for**( **int** x: array) {  result=result+x;  }  System.***out***.println(result);  }  } |

Which of the following valid var-arg method declarations

m(int[] x)

m()

m()

1. We can mix var-arg parameter with normal parameter.
2. If we mix normal parameter with var-arg parameter then var-arg parameter should be last parameter.
3. Inside var-arg method we can take only one var-arg parameter and we can’t take more than one var-arg parameter
4. Inside a class we can’t declare var-arg method and one dimensional array method simultaneously otherwise we will see compile time error.
5. In general var-arg method will set least priority i.e if no other method matched then only var-arg method will get the chance it is exactly same as default case inside switch.
6. Whenever one dimensional array present we can replace with var-arg parameter
7. Whenever var-arg parameter present we can’t replace with one dimensional array.
8. We can call this method by passing a group of int values and x will become one dimensional array {int[] x}.
9. We can call this method by passing a group of one dimensional int arrays an x will become 2 dimensional int array int [][] x.