**中软国际考题-笔试**

1. 单选题

1、一下说法正确的是（A）

A. 在调用方法的时候，如果能够和固定参数的方法匹配，也能够与可变长参数的方法匹配，则选择固定参数的方法

B. 如果要调用的方法可以和两个可变参数匹配，则选择定义靠前的方法执行

C. 一个方法可以有多个可变长参数，但是这些变长参数必须是不同数据类型

D.Java中同一个类只能存在一个带有可变长参数的方法

2、下列描述错误的是（B）

A.参数一般与方法有关

B.在代码创建方法名与方法体时添加的参数为实际参数

C.参数可以为一个，也可以为多个传递

D.无参数的表示方式只需要方法小括号内不声明编写

3、下列描述正确的是（A）

1. 参数除了可以为基本数据类型，也可以为复合数据容器
2. 实际参数比形式参数类型更加灵活多变
3. 参数不允许为真假值
4. 常量允许为参数

4、java中参数描述错误的是（D）

A.参数在方法声明中可设定有参，也可设定无参数

B.参数个数没有多少的限制

C.参数允许跨类传递

D.参数不允许为null值

5、java 可变参数是哪一个版本的新特性（B）

A.1.0 B.1.5 C.1.7 D.1.8

6、下列语句序列执行后能输出正确的是（A）

Public class TextVarArray{

Public static void text1(int... intArray){

for(int i : intArray){

Suystem.out.println(i+””);

}

System.out.println();

}

Pulbic static void main(String args[]){

text1();

Text1(1);

text1(1,2,3);

}

}

A.1 B. C.null D.1

1 2 3 1 1 123

123 1 2 3

7、 Java用什么表示可变参数(A)

A. …

B. ..

C. >>>

D. <<<

8. 以下程序运行结果（C）

package com;

// 这里使用了静态导入

import static java.lang.System.out;

public class VarArgsTest1 {

public void print(String... args) {

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

out.println(args[i]);

}

}

public void print(String test,String...args ){

out.println("----------");

}

public static void main(String[] args) {

VarArgsTest1 test = new VarArgsTest1();

test.print("hello");

test.print("hello", "alexia");

}

}

A. hello hello alexia B. hello

C. 运行出错 D. 无结果

9. Java可变参数是从java哪个版本开始有的（C）

A.jdk 1.1 B.jdk1.2 C.jdk1.5 D.java8新特性

10、Java中，关于可变参数的运用，下面正确的是（D）.

A.void foo(String args) B. void foo(String. args)

C. void foo(String.. args) D. void foo(String… args)

11、下列可变参数定义正确的是（B）

A public void test1(String…args,Object o){}

B public void test1(String…args){}

C public void test1(String…args,Ineger…i){}

D public void test1(String[] args){}

1. 多选题
2. 以下说法正确的有：（A、B、C、D）

A．可变参数只能作为函数的最后一个参数，但其前面可以有也可以没有任何其他参数

B. 由于可变参数必须是最后一个参数，所以一个函数最多只能有一个可变参数

C. Java的可变参数，会被编译器转型为一个数组

D. 变长参数在编译为字节码后，在方法签名中就是以数组形态出现的。和直接使用数组的方法的签名是一致的，不能作为方法的重载。如果同时出现，是不能编译通过的

2、下面哪些项描述不正确？（A、B）

A. 参数一定会参与调用方法代码的逻辑执行

B. 形式参数是在方法名和方法体编写创造时申明的参数

C. 参数与调用方法代码一一对应

D. 参数可以用来区分构造方法

3、以下含有可变参数方法定义正确的有

A.public void main(Object... obj) {} B.public void main(Object... obj,int... a) {}

C.public void main(int... a,Object... obj) {} D.public void main(int a,Object... obj) {}

4、下面两个方法可以怎么样被调用？（A、B、D）

public static void callMe1(String[] args) {

System.out.println(args.getClass() == String[].class);

for(String s : args) {

System.out.println(s);

}

}

public static void callMe2(String... args) {

System.out.println(args.getClass() == String[].class);

for(String s : args) {

System.out.println(s);

}

}

A. callMe1(new String[] {"a", "b", "c"});

B. callMe2("a", "b", "c");

C. lMe1("a", "b", "c");

D. callMe2(new String[] {"a", "b", "c"});

5、以下说法正确的是？（A、B、C）

A.方法可变参只能用在最后一个参B. ...位于变量类型和变量名之间，前后有无空格都可以

C. 调用可变参数的方法时，编译器为该可变参数隐含创建一个数组，在方法体中以数组的形式访问可变参数。

D. 可以传空值

6、Java中，以下说法正确的是（ ABD ） 。

A.Java5中提供了可变长参数 B.可变长参数的实质还是基于数组实现

C. 可变长参数可以和泛型机制配合使用

D.调用一个被重载的方法时，如果此调用既能够和两个可变长参数的重载方法匹配，则编译错误

7、定义方法如下

public class Client {

public void methodA(String str,Integer... is){

}

public void methodA(String str,String... strs){

}

}下列方法调用有误的是(C D)

A, client.methodA("China", 0); B, client.methodA("China", "People");

C, client.methodA("China"); D, client.methodA("China",null);

8、public static void dealArray(int... intArray) {

for (int i : intArray)

System.out.print(i + " ");

System.out.println();

}

以下说法正确的是( ) A D

A 可变参数是兼容数组参数的 B数组参数可以兼容可变参数的

C 可变参数必须放在参数列表最前面 D可变参数必须放在参数列表最后一个

9、public void method(int...args1, int...args2) B. public void method(int i, int j, int...args)

C. public void method(int...args, int i) D. public void method(int i，int...args1, int...args2)

10、分析下列语句：

public void method(String...args); //方法1

public void method(String s1); //方法2

public void method(String s2, String s3); //方法3

public static void main(String[] args){

method("看看调用哪个方法"); //方法调用1

method("猜猜是哪个", "原来是这个！"); //方法调用2

}

语句中两个方法调用，分别调用的是哪两个方法（D）：

A. 方法1，方法3 B. 方法1，方法1

C. 方法2，方法1 D. 方法2，方法3

11、下面选项中的调用dealArray方法正确的是（A、B、C、D）？

public class TestVarArgus {

public static void dealArray(int... intArray){

for (int i : intArray)

System.out.print(i +" ");

System.out.println();

}

public static void main(String args[]){

//调用dealArray方法

//将选项中的内容填写到这个位置

}

}

A. dealArray(1);

B. dealArray(1,2,3);

C. int[] intArray = {1, 2, 3}; dealArray(intArray);

D. dealArray(null);

1. 填空题（每空1分，共10分）
2. 在Java方法中，可变参数被看作\_\_\_\_\_\_\_\_\_\_来处理。 (数组)
3. 参数语法 请填所在位置中文说明：public \_\_\_\_\_\_\_\_\_ void play(\_\_\_\_\_\_\_\_\_ f,int \_\_\_\_\_\_\_\_\_){}。(static，参数数据类型，参数名)
4. 可变参数在方法中可以被认为是\_\_\_\_\_\_\_\_\_\_ 进行使用（数组）
5. 在定义方法时，在最后一个形参后加上三点 …，就表示该形参可以接受\_\_\_\_\_\_\_\_\_参数值，\_\_\_\_\_\_\_\_参数值被当成数组传入(多个, 多个)
6. 使用\_\_\_\_\_表示可变参数。（…）
7. 在 JDK\_\_\_\_版本中提供了变长参数，允许在调用方法时传入不定长度的参数。（5）
8. 由于可变参数必须是\_\_\_\_\_\_\_\_\_\_，所以一个函数最多只能有\_\_\_\_可变参数(最后一个参数,一个)
9. 可变参数要放在形参列表的 。(最后)
10. 形参列表中至多允许 可变参数。(一个)
11. 判断题
12. 每个方法最多可以有一个可变参数，而且可变参数必须位于最后一项。（ T ）
13. 可变参数可以兼容数组，反之则不成立（T）
14. 泛型机制不能和个数可变的形参配合使用。如果把一个能和不确定个实参相匹配的形参的类型，用一个标识符来代表，那么编译器会给出一个“generic array creation”的错误（T）
15. **参数只能由基本数据类型和自定义类组成方法传参。（ N ）**
16. 可变参数至少传递一个值。（F）
17. 可变参数必须出现在参数列表的最后（T）
18. 可变参数类型可以不用作为参数列表的最后一项。（ F ）
19. void foo（String... args）；与void foo（String[] args）的方法作用一样吗？（ T ）
20. 在具有可变长参数的方法中可以把参数当成数组使用。（ T ）
21. 可变参数方法调用的时候可以给出任意多个参数也可不给参数。（ T）
22. 调用的时候可以给出任意多个参数可以给参数（F）
23. 可变参数必须放在方法的最后一个（ T ）
24. 可变参数可以放在参数列表的任意位置（ T ）
25. 可变参数列表不可以和数组 形成重载（T）
26. 可变参数列表可以和数组 形成重载（F）
27. 可变参数在方法体内可以当成数组来使用（T）
28. 可变参数只能作为函数的最后一个参数 （T）
29. 由于可变参数必须是最后一个参数，所以一个函数最多只能有一个可变参数（T）
30. Java的可变参数，会被编译器转型为一个数组（T）
31. 变长参数在编译为字节码后，在方法签名中就是以对象形态出现的 （F）
32. 调用可变参数方法，可以给出零到任意多个参数，编译器会将可变参数转化为一个数组（T）
33. 使用 Object… 作为变长参数时，int[] 无法转型为 Object[], 因而被当作一个单纯的数组对象 ; Integer[] 可以转型为 Object[], 可以作为一个对象数组。（T）
34. 在不确定方法需要处理的对象的数量时可以使用可变长参数，会使得方法调用更简单，无需手动创建数组 new T[]{…} （T）
35. 使用…将参数声明成可变长参数。（T）
36. 可变参数同时可以跟固定的参数混合使用，但是一个方法的参数中不能同时拥有2种类型的可变参数（T）
37. 变长参数在编译为字节码后，在方法签名中就是以数组形态出现的 （T）
38. 可变参数只能作为函数的最后一个参数，但其前面可以有也可以没有任何其他参数（正确）
39. Java的可变参数，会被编译器转型为一个数组（正确）
40. 可变参数的方法能够接受任意个参数。（ T ）
41. 可变参数类型必须作为参数列表的最后一项。（ T ）
42. 方法调用时，如果能够同时匹配定长参数的方法和不定参数的方法，那么优先匹配含有不定参数的方法。（ F ）

五、简答题

1、什么是可变参数？

答案：

Java1.5增加了新特性：可变参数：适用于参数个数不确定，类型确定的情况，java把可变参数当做数组处理。可变参数必须位于最后一项。当可变参数个数多余一个时，必将有一个不是最后一项，所以只支持有一个可变参数。因为参数个数不定，所以当其后边还有相同类型参数时，java无法区分传入的参数属于前一个可变参数还是后边的参数，所以只能让可变参数位于最后一项。

可变参数的特点：

（1）只能出现在参数列表的最后；

（2）...位于变量类型和变量名之间，前后有无空格都可以

（3）调用可变参数的方法时，编译器为该可变参数隐含创建一个数组，在方法体中以数组的形式访问可变参数。

2、下列代码能否编译通过？说明原因。

public class VarArgsTest{

/\*\*

\* @param args

\*/

public static void main(String[] args) {

// TODO Auto-generated method stub

// 向上转型

Base base = new Sub();

base.print("hello");

// 不转型

Sub sub = new Sub();

sub.print("hello");//compile error

}

}

// 基类

class Base {

void print(String... args) {

System.out.println("Base......test");

}

}

// 子类，覆写父类方法

class Sub extends Base {

@Override

void print(String[] args) {

System.out.println("Sub......test");

}

答案：

能编译通过，base对象把子类对象sub做了向上转型，形参列表是由父类决定的，所以能通过。而看看子类直接调用的情况，这时编译器看到子类覆写了父类的print方法，因此肯定使用子类重新定义的print方法，尽管参数列表不匹配也不会跑到父类再去匹配下，因为找到了就不再找了，因此有了类型不匹配的错误

这是个特例，覆写的方法参数列表竟然可以与父类不相同，这违背了覆写的定义，并且会引发莫名其妙的错误

3、请简述形式参数与实际参数的主要特点（5分）

答案：（1）. 形参：全称为“形式参数”，是在定义方法名和方法体的时候使用的参数，用于接收调用该方法时传入的实际值

（2）. 全称为“实际参数”，是在调用方法时传递给该方法的实际值

4、可变参数的特点：

（1）只能出现在参数列表的最后；

（2）...位于变量类型和变量名之间，前后有无空格都可以；

（3）调用可变参数的方法时，编译器为该可变参数隐含创建一个数组，在方法体中以数组的形式访问可变参数

5、在定义方法时，在最后一个形参后加上三点“…“，就表示该形参可以接受多个参数值，多个参数值被当成数组传入。请叙述需要注意的地方：。(写出2条即可)（5分）

答案：

1.可变参数只能作为函数的最后一个参数，但其前面可以有也可以没有任何其他参数

2.由于可变参数必须是最后一个参数，所以一个函数最多只能有一个可变参数

3.Java的可变参数，会被编译器转型为一个数组

4.变长参数在编译为字节码后，在方法签名中就是以数组形态出现的。这两个方法的签名 是一致的，不能作为方法的重载。如果同时出现，是不能编译通过的。可变参数可以兼容 数组，反之则不成立

6、什么时候可以使用可变长参数？

答案：在不确定方法需要处理的对象的数量时可以使用可变长参数，会使得方法调用更简单，无 需手动创建数组

7、运行以下程序会输出的结果

package com;

// 这里使用了静态导入

import static java.lang.System.out;

public class VarArgsTest {

public void print(String... args) {

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

out.println(args[i]);

}

}

public void print(String test) {

out.println("----------");

}

public static void main(String[] args) {

VarArgsTest test = new VarArgsTest();

test.print("hello");

test.print("hello", "alexia");

}

}

运行结果：

-----------

hello

alexia

8、以下代码运行结果：

package com;

// 这里使用了静态导入

import static java.lang.System.out;

public class VarArgsTest1 {

public void print(String... args) {

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

out.println(args[i]);

}

}

public void print(String test,String...args ){

out.println("----------");

}

public static void main(String[] args) {

VarArgsTest1 test = new VarArgsTest1();

test.print("hello");

test.print("hello", "alexia");

}

}

答案： 编译错误

9、什么是可变参数

答案：在 Java 5 中提供了变长参数，允许在调用方法时传入不定长度的参数。变长参数是 Java 的一个语法糖，本质上还是基于数组的实现

10,可变参数与泛型的关系

答案：J2SE 1.5 中新增了**"泛型"**的机制，可以在一定条件下把一个类型参数化。例如，可以在编写一个类的时候，把一个方法的形参的类型用一个标识符（如T）来代表， 至于这个标识符到底表示什么类型，则在生成这个类的实例的时候再行指定。这一机制可以用来提供更充分的代码重用和更严格的编译时类型检查。不过泛型机制却不能和个数可变的形参配合使用。如果把一个能和不确定个实参相匹配的形参的类型，用一个标识符来代表，那么编译器会给出一个 **"generic array creation"** 的错误

11、可变参数和数组有什么区别？

可变参数是兼容数组类参数的，但是数组类参数却无法兼容可变参数。

12以下代码编译结果是什么\_\_\_\_\_ (编译错误)

public class TestVarArgus {

public static void dealArray(int[] intArray) {

for (int i : intArray)

System.out.print(i + " ");

System.out.println();

}

public static void main(String args[]) {

dealArray(1, 2, 3);

}

13 可变参数的特点有哪些？

1可变参数只能作为函数的最后一个参数，但其前面可以有也可以没有任何其他参数

2由于可变参数必须是最后一个参数，所以一个函数最多只能有一个可变参数

3Java的可变参数，会被编译器转型为一个数组

4变长参数在编译为字节码后，在方法签名中就是以数组形态出现的。这两个方法的签名是一致的，不能作为方法的重载。如果同时出现，是不能编译通过的。可变参数可以兼容数组，反之则不成立

14以下代码能编译通过的是？

A public static void dealArray(int.. intArray, int count)

B public static void dealArray(int count, int.. intArray)

C public static void dealArray(int... intArray, int count)

D public static void dealArray(int count, int... intArray)

15以下代码编译是否能通过？（X）

public class TestVarArgus {

public static void dealArray(int... intArray) {

for (int i : intArray)

System.out.print(i + " ");

System.out.println();

}

public static void dealArray(int[] intArray) {

for (int i : intArray)

System.out.print(i + " ");

System.out.println();

}

public static void main(String args[]) {

dealArray();

dealArray(1);

dealArray(1, 2, 3);

}

}

16可变参数类型要放在参数列表的最后一项，也能放在定长参数的最前面（ X ）

17请使用可变参数来完善下面的代码，

public class TestVarArgus {

public static void main(String args[]) {

dealArray(1);

dealArray(1,2);

dealArray(1, 2, 3);

}

}

输出： 1

1 2

1 2 3

六、程序实现题

1、用可变参数求多个数的和（个数不确定）

|  |
| --- |
| public class Varable {  public static void main(String[] args) {  System.out.println(add(2, 3));  System.out.println(add(2, 3, 5));  }    public static int add(int x, int... args) {  int sum = x;  for (int i = 0; i < args.length; i++) {  sum += args[i];  }  return sum;  }  } |

2、参数方法调用：创建一个带2个以上不同类型参数方法，并在主方法中实际调用

（10分）

|  |
| --- |
| public class Test01{  public static void main0(String f,int cc){  System.out.println(f);  }  public static void main(String[] args) {  // TODO Auto-generated method stub  String aac=null;  main0(aac,22);    }  } |

3、用可变参数求多个数的和（至少一个数）（10分）

答案：

|  |
| --- |
| public class Varable {      public static void main(String[] args) {          System.out.println(add(2, 3));          System.out.println(add(2, 3, 5));      }        public static int add(int x, int... args) {          int sum = x;          for (int i = 0; i < args.length; i++) {              sum += args[i];          }          return sum;      }  } |

4、求若干个整型数中的最大值

|  |
| --- |
| import java.util.ArrayList;  import java.util.List;      public class VariableParameter {    //求若干个整型数中的最大值  public int getMax(int... items){ //定义可变参数items    int max = Integer.MIN\_VALUE; //次数为int能表示的最小值，值为-2147483648  for(int item : items){  max = item > max? item : max; //取大值  }  return max;  }  //返回比number大的数的个数  public int getLagerNum(int number,int... items){ //若除了可变长参数还有其它参数，可变长参数一定要放在最后  int sum = 0;  for(int item : items){  if(item > number){  sum++;  }  }  return sum;  }    public static void main(String[] args) {  VariableParameter vp = new VariableParameter();  System.out.println("最大值："+vp.getMax()); //可以不写参数  System.out.println("最大值："+vp.getMax(2,1,4,7,2,-1,3,3)); //可以在参数中直接写入若干个参数  System.out.println("最大值："+vp.getMax(-1,4,5,0,1)); //可以在参数中直接写入若干个参数    System.out.println("比number大的数的个数："+vp.getLagerNum(9, 1,11,5,14,-1,9,21)); //9为number，后边的是items  int a[] = new int[] {1,11,5,14,-1,9,21};  System.out.println("比number大的数的个数："+vp.getLagerNum(9,a)); //items也可以用数组替代    }    } |

5、金额转换：编写一个可以接收可变参数方法

（10分）

|  |
| --- |
| public class TestVarArgus {  public static void dealArray(int... intArray){  for (int i : intArray)  System.out.print(i +" ");  System.out.println();  } |

6、编 写add方法，求任意个整数求和

（10分）

|  |
| --- |
| Public class Varable {      public static void main(String[] args) {          System.out.println(add(2, 3));          System.out.println(add(2, 3, 5));      }        public static int add(int x, int... args) {          int sum = x;          for (int i = 0; i < args.length; i++) {              sum += args[i];          }          return sum;      }  } |

7、定义可变参数计算多个参数的总和

（10分）

|  |
| --- |
| public class Main {  static int sumvarargs(int... intArrays){  int sum, i;  sum=0;  for(i=0; i< intArrays.length; i++) {  sum += intArrays[i];  }  return(sum);  }  public static void main(String args[]){  int sum=0;  sum = sumvarargs(new int[]{10,12,33});  System.out.println("数字相加之和为: " + sum);  }  } |

8、使用可变参数编写方法，判断输入的参数是否为空：例如：调用方法isNotEmpty(null)能够返回false,isNotEmpty(agr1)能够返回true。

|  |
| --- |
| 调用方法类似：isNotEmpty(v1,v2,v3,v4,v5)  /\*\*  \* 判断所有的参数是否为空  \* @param objects  \* @return  \*/  public static boolean isNotEmpty(Object ...objects){  if(objects!=null){  for(Object o:objects){  if(o==null||"".equals(o)){  return false;  }  }  }  return true;  } |