1. short s1 = 1; s1 = s1 + 1;有错吗?short s1 = 1; s1 += 1;有错吗？（编程试试。）

答：

对于short s1 = 1; s1 = s1 + 1;由于1是int类型，因此s1+1运算结果也是int 型，需要强制转换类型才能赋值给short型。

而short s1 = 1; s1 += 1;可以正确编译，因为s1+= 1;相当于s1 = (short)(s1 + 1);其中有隐含的强制类型转换。

2. float f=3.4;是否正确？（编程试试。）

答：

不正确。

3.4是双精度数，将双精度型（double）赋值给浮点型（float）属于下转型（down-casting，也称为窄化）会造成精度损失，因此需要强制类型转换float f =(float)3.4; 或者写成float f =3.4F;。

3. 重载（Overload）和重写（Override）的区别。重载的方法能否根据返回类型进行区分？

答：

方法的重载和重写都是实现多态的方式，区别在于前者实现的是编译时的多态性，而后者实现的是运行时的多态性。

重载发生在一个类中，同名的方法如果有不同的参数列表（参数类型不同、参数个数不同或者二者都不同）则视为重载；重写发生在子类与父类之间，重写要求子类重写方法与父类被重写方法有相同的返回类型，有更宽泛的访问权限。

重载对返回类型没有特殊的要求。

4. 抽象类和接口有什么异同？

答：

抽象类和接口都不能够实例化，但可以定义抽象类和接口类型的引用。

一个类如果继承了某个抽象类或者实现了某个接口都需要对其中的抽象方法全部进行实现，否则该类仍然需要被声明为抽象类。

接口比抽象类更加抽象，因为抽象类中可以定义构造器，可以有抽象方法和具体方法，而接口中不能定义构造器，而且其中只能有抽象方法，default方法和static方法。

抽象类中可以定义成员变量，而接口中定义的成员变量实际上都是常量。

有抽象方法的类必须被声明为抽象类，而抽象类未必要有抽象方法。

5. 抽象的方法是否可同时是静态的？（编个小程序试试。）

**答：**

不能。

抽象类是不能实例化的，即不能被分配内存;而static修饰的方法在类实例化之前就已经别分配了内存，这样一来矛盾就出现了：抽象类不能被分配内存，而static方法必须被分配内存。所以抽象类中不能有静态的抽象方法。

抽象方法需要子类重写，而静态的方法是无法被重写的。

6. 阐述静态变量和实例变量的区别。

答：

静态变量是被static修饰符修饰的变量，也称为类变量，它属于类，不属于类的任何一个对象，一个类不管创建多少个对象，静态变量在内存中有且仅有一个拷贝。

实例变量必须依存于某一实例，需要先创建对象然后通过对象才能访问到它。

静态变量可以实现让多个对象共享内存。

7. 是否可以从一个静态方法内部发出对非静态方法的调用？请具体解释一下。

答：

不可以，静态方法只能访问静态成员，因为非静态方法的调用要先创建对象，在调用静态方法时可能对象并没有被初始化。

8. 接口是否可继承（extends）接口？抽象类是否可实现（implements）接口？抽象类是否可继承具体类（concrete class）？

答：

接口可以继承接口，而且支持多重继承。

抽象类可以实现(implements)接口，抽象类可继承具体类也可以继承抽象类。

9. Java 中的final关键字有哪些用法？

答：

（1）修饰类：表示该类不能被继承.

（2）修饰方法：表示方法不能被重写。

（3）修饰变量：表示变量只能一次赋值以后值不能被修改（常量）。

10. switch 是否能作用在byte 上，是否能作用在long 上，是否能作用在String上？

答：

在Java 5以前，switch(expr)中，expr只能是byte、short、char、int。从Java 5开始，Java中引入了枚举类型，expr也可以是enum类型，从Java 7开始，expr还可以是字符串（String），但是长整型（long）在目前所有的版本中都是不可以的。

11. 用最有效率的方法计算2乘以8？

答：

2 << 3（左移3位相当于乘以2的3次方，右移3位相当于除以2的3次方）。

12. Java 中会存在内存泄漏吗，请简单描述。

答：

理论上Java因为有垃圾回收机制（GC）不会存在内存泄露问题（这也是Java被广泛使用于服务器端编程的一个重要原因）；然而在实际开发中，可能会存在无用但可达的对象，这些对象不能被GC回收，因此也会导致内存泄露的发生。

例如Hibernate的Session（一级缓存）中的对象属于持久态，垃圾回收器是不会回收这些对象的，然而这些对象中可能存在无用的垃圾对象，如果不及时关闭（close）或清空（flush）一级缓存就可能导致内存泄露。下面例子中的代码也会导致内存泄露。