۱) تفاوت varargs با آرایه چیست ؟

The only **difference** is signature of a method that might have a variable number of arguments, as opposed to an **array** argument you can pass only one argument. ... On the other hand **vararg** arguments will be converted to an **array** and used there.

۲) unboxing و autoboxing چیست ؟

Autoboxing and **Unboxing**. **Autoboxing** is the automatic conversion that the **Java** compiler makes between the primitive types and their corresponding object wrapper classes. For example, converting an int to an Integer, a double to a Double, and so on. If the conversion goes the other way, this is called **unboxing**.

۳) چگونه یک شی Integer را توانستیم به ۲ تقسیم کنیم ؟

وقتی شی Integer را می خواهیم بر ۲ تقسیم کنیم ، unBoxing اتفاق می افتد و شی به primitive type تبدیل می شود که حالا توانایی تقسیم شدن بر ۲ را دارد.

۴) چگونه می توان از متد های مخصوص یک شکل خاص مثل دایره استفاده کرد ؟

در این سوال باید با استفاده از یک حلقه for_each و کلمه کلیدی instanceof دایره را پیدا کرده و متد خواص آنرا صدا می زنیم.

۵) انجام دهید ۳ :

```
Circle circle1 = new Shape(19);
```

در این خط ارور داریم زیرا شی shape یک کلاس abstract است و نمیتوان از آن شی ساخت.

صحیح :

```
Circle circle1 = new Circle(19);
```

```
Shape circle2 = new Circle(3);
```

در این خط شی ای از کلاس circle در reference ای از shape ذخیره شده است و صحیح است. (فرق آن با ساختن معمولی شی از کلاس circle این است که در این حالت دسترسی به متد های خاص کلاس circle نداریم.)

```
Rectangle rect1 = new Triangle(1,4,1);
```

غلط است چون شی از کلاس triangle را در reference ای از rectangle ذخیره کرده که این اشتباه است.

صحیح :

```
Triangle rect1 = new Triangle(1,4,1);
```

```
Polygon rect2 = new Rectangle(8,5,8,5);
```

این خط صحیح است ، زیرا شی از کلاس rectangle در reference ای از polygon ذخیره شده است که کلاس والد است.

```
Rectangle rec3 = new Shape(6,6,6,6);
```

در این خط ارور داریم زیرا شی shape یک کلاس abstract است و نمیتوان از آن شی ساخت.

صحیح :

```
Rectangle rec3 = new Rectangle(6,6,6,6);
```

```
Polygon tri1 = new Triangle(2,2,2);
```
این خط صحیح است ، زیرا شی از کلا triangle در reference ای از polygon ذخیره شده است که کلاس والد است.
```
Triangle tri2 = new Triangle(4,4,6);
```
صحیح است.
```
Shape tri3 = new Triangle(2,2,2);
```
این خط صحیح است ، زیرا شی از کلا triangle در reference ای از shape ذخیره شده است که کلاس والد است.

```
circle1 = circle2;
```
این خط اشتباه است ، زیرا شی circle2 با reference ای از shape در شی circle1 با reference ای از circle قرار داده شده که این کار مجاز نیست .(اما بر عکس آن صحیح است.)
```
rect2 = rect3;
```
صحیح است ، هر دو reference ای از rectangle دارند.
```
tri1 = tri3;
```
این خط اشتباه است ، زیرا شی tri3 با reference ای از shape در شی tri1 با reference ای از polygon قرار داده شده که این کار مجاز نیست .(اما بر عکس آن صحیح است.)
```
circle2 = tri3;
```
صحیح است ، هر دو reference ای از shape دارند.
```
tri3 = tri2;
```
این خط صحیح است ، زیرا شی tri2 با reference ای از triangle در شی tri3 با reference ای از shape قرار داده شده که این کار مجاز است. (چون کلاس shape کلاس والد برای triangle است)
```
rect3 = new Shape(2, 3, 2);
```
در این خط ارور داریم زیرا شی shape یک کلاس abstract است و نمیتوان از آن شی ساخت.
```
System.out.println(rect3.toString());
```
در این خط متد toString از کلاس rectangle صدا زده می شود .
خروجی: Rectangle ¯ sides : side 1 = 6 | side 2 = 6 | side 3 = 6 | side 4 = 6

۶) چرا به اشکال های نرم افزاری اصطلاحا bug گفته می شود؟

گریس موری هاپر، کارشناس زبردست کامپیوتر از پیشگامان برنامه نویسی نوین و عضو نیروی دریایی آمریکا وی نخستین کامپایلر را ابداع کرد.

مارک ۲ (کامپیوتر) برخلاف نسخه اولیه با استفاده از رله های الکترومغناطیسی با سرعت بالا ساخته شده بود اما پس از کار افتادن تکنسین های تحت نظر هاپر پس از باز کردن این سیستم در رله ۷۰ آن سوسکی را پیدا کردند. هاپر جسد حشره نگون بخت را در دفترچه خود چسباند و در متن کنار آن از پیدا شدن نخستین باگ واقعی خبر داد بر اساس وب سایت نیروی دریایی آمریکا این رویداد مقدمه ای برای استفاده از واژه باگ شد.