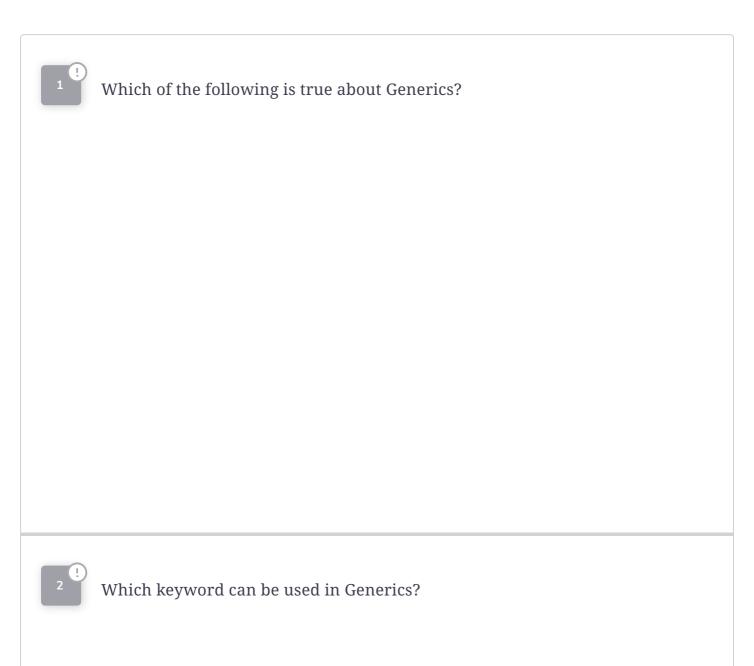
## Quick Quiz!

Here is a quick quiz to check your understanding of Generics.



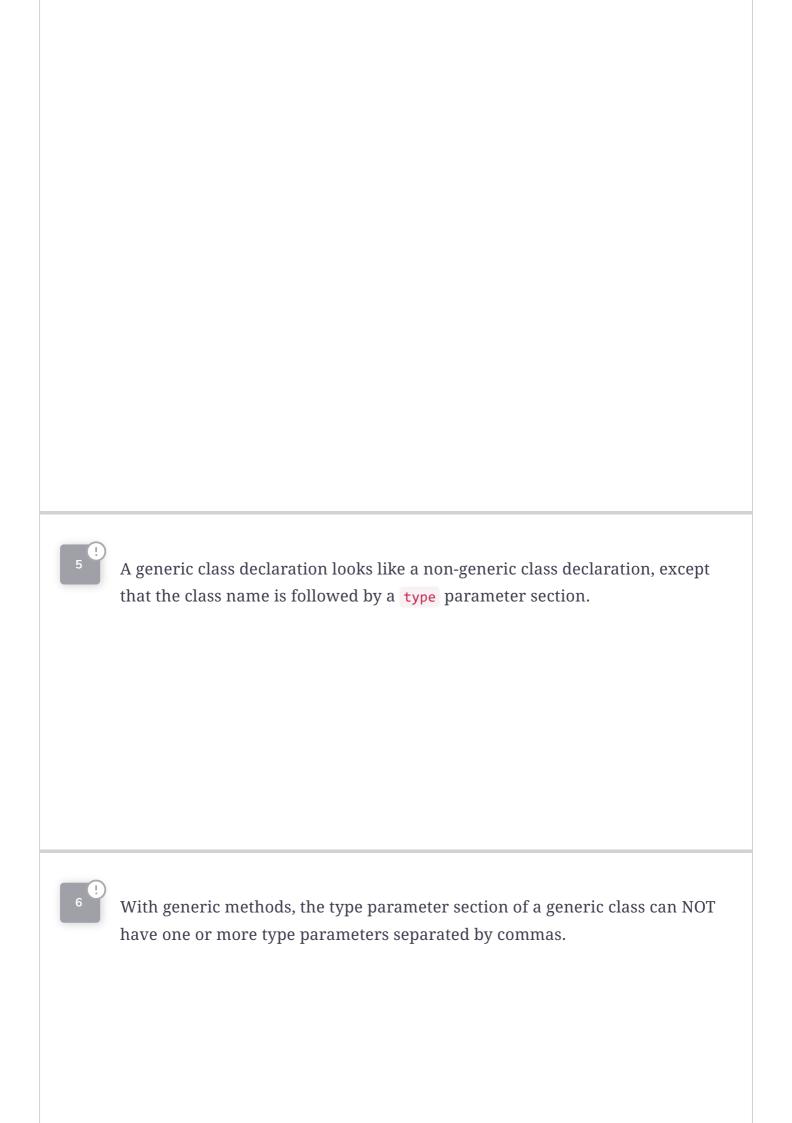
What is the output of the following program?

```
class GetMax {
  public static <T extends Comparable<T>> T maximum (T a, T b)
  {
    if(a.compareTo(b) > 0)
      return a;
    else
      return b;
  }
  public static void main( String args[] )
  {
    System.out.println(maximum(4, 6));
    System.out.println(maximum(4.0, 6.1));
    System.out.println(maximum(4, 6.7));
  }
}
```

4

## What is the output of the following program?

```
class Score <T>
{
  T value;
  public int counter = 0;
  public Score() {counter++;}
  public Score(T v) { value = v; counter++;}
};
class Main
 public static void main(String args[])
 {
   Score <Integer> x = new Score <Integer>();
   Score <Integer> y = new Score <Integer>();
   Score <Double> z = new Score <Double> ();
   System.out.println(x.counter);
   System.out.println(y.counter);
   System.out.println(z.counter);
```



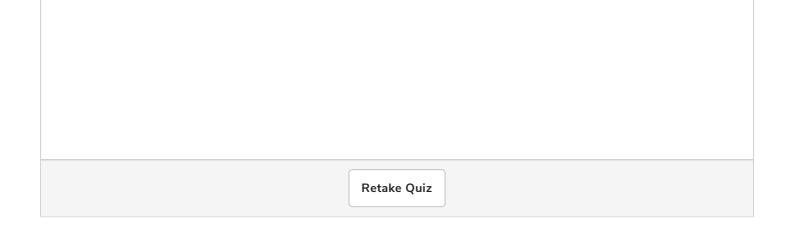


What are the benefits of Generics in Java?



Generics can be used for more than one types defined in the form of comma separated list in the class / function header.

```
public static <A,B,T> void MultipleTypesFunction(A x, B y, C z)
{
    System.out.println( x + " is Type A");
    System.out.println( y + " is Type B");
    System.out.println( z + " is Type C");
}
```



In the next chapter, we will discuss array lists.