Interview Questions | MCQs

300+ [LATEST] Java Generics Interview Questions and Answers

1. Given The Following Classes:

class Shape { /* ... */ }
class Circle Extends Shape { /* ... */ }
class Rectangle Extends Shape { /* ... */ }
class Node { /* ... */ }
will Th

No. Because Node is not a subtype of Node.

110@Consider This Class:
class Node Implements Comparable {
Public Int Compareto(t Obj) { /* ... */ }

```
// ...
}
```

will The Following Code Compile? If Not, Why?

2. Consider This Class:

```
class Node Implements Comparable {
Public Int Compareto(t Obj) { /* ... */ }
// ...
}
```

will The Following Code Compile? If Not, Why?

Yes.

```
Node node = new Node<>();
Comparable comp = node;
```

120@What Is A Parameterized Or Generic Type?

3. How Do You Declare A Generic Class?

Note the declaration of class:Instead of T, We can use any valid identifier.

- class MyListGeneric

4. How Can We Restrict Generics To A Super Class Of Particular Class?

In MyListGeneric, Type T is defined as part of class declaration. Any Java Type can be used a type for this class. If we would want to restrict the types allowed for a Generic Type, we can use a Generic Restrictions. In declaration of the class, we specified a constraint “T super Number”. We can use the class MyListRestricted with any class that is a super class of Number class.

5. Write A Generic Method To Exchange The Positions Of Two Different Elements In An Array.

```
public final class Algorithm
{
public static void swap(T[] a, int i, int j)
{
T temp = a[i];
a[i] = a[j];
a[j] = temp;
}
}
```

6. What Is The Following Class Converted To After Type Erasure?

```
Public Class Pair {

Public Pair(k Key, V Value) {
This.key = Key;
This.value
```

```
public class Pair {
public Pair(Object key, Object value) {
this.key = key;
this.value = value;
}
public Object getKey()   { return key; }
public Object getValue() { return value; }
public void setKey(Object key)     { this.key = key; }
public void setValue(Object value) { this.value = value; }
private Object key;
private Object value;
}
```

7. Can We Use Generics With Array?

If you know the fact that Array doesn’t support Generics and that’s why Joshua bloach suggested to prefer List over Array because List can provide compile time type-safety over Array.

8. How To Write Parametrized Class In Java Using Generics ?

This is an extension of previous Java generics interview question. Instead of asking to write Generic method Interviewer may ask to write a type safe class using generics. again key is instead of using raw types you need to used generic types and always use standard place holder used in JDK.

9. What Is A Parameterized Or Generic Type?

A generic type is a type with formal type parameters. A parameterized type is an instantiation of a generic type with actual type arguments.

A generic type is a reference type that has one or more type parameters. These type parameters are later replaced by type arguments when the generic type is instantiated (or declared).

Example (of a generic type):

```
interface Collection  {
public void add (E x);
public Iterator iterator();
}
```

The interface Collection has one type parameter E . The type parameter E is a place holder that will later be replaced by a type argument when the generic type is instantiated and used. The instantiation of a generic type with actual type arguments is called a parameterized type

Example (of a parameterized type):

```
Collection coll = new LinkedList();
```

The declaration Collection denotes a parameterized type, which is an instantiation of the generic type Collection , where the place holder E has been replaced by the concrete type String .

10. What Are Generics?

Generics are used to create Generic Classes and Generic methods which can work with different Types(Classes).

11. Will The Following Class Compile? If Not, Why?

```
public Final Class Algorithm {
Public Static T Max(t X, T Y) {
Return X > Y ? X : Y;
}
}
```

No. The greater than (>) operator applies only to primitive numeric types.

12. What Is The Following Method Converted To After Type Erasure?

```
public Static >
Int Findfirstgreaterthan(t[] At, T Elem) {
// …
}
```

```
public static int findFirstGreaterThan(Comparable[] at, Comparable
elem) {
// …
}
```

13. Will The Following Method Compile? If Not, Why?

public Static Void Print(list Extends Number> List) {
For (number N : List)
System.out.print(n + ” “);
System.out.

Yes.

14. Will The Following Class Compile? If Not, Why?

```
public Class Singleton {

Public Static T Getinstance() {
If (instance == Null)
Instance = New Single
```

No. You cannot create a static field of the type parameter T.

100@Given The Following Classes:
class Shape { /* … */ }
class Circle Extends Shape { /* … */ }
class Rectangle Extends Shape { /* … */ }
class Node { /* … */ }
will The Following Code Compile? If Not, Why?
node Nc = New Node();
node Ns = Nc;

15. Whata Are The Type Parameters?

Type Parameters: The type parameters naming conventions are important to learn generics thoroughly.

The commonly type parameters are as follows:

- T – Type
- E – Element
- K – Key
- N – Number
- V – Value

16. What Are Advantages Of Using Generics?

Advantage of Java Generics:

There are mainly 3 advantages of generics. They are as follows:

Type-safety : We can hold only a single type of objects in generics. It doesn’t allow to store other objects.

Type casting is not required: There is no need to typecast the object.

Before Generics, we need to type cast.

```
List list = new ArrayList();
list.add(“hello”);
String s = (String) list.get(0);//typecasting
```

After Generics, we don’t need to typecast the object.

```
List list = new ArrayList();
list.add(“hello”);
String s = list.get(0);
```

Compile-Time Checking: It is checked at compile time so problem will not occur at runtime. The good programming strategy says it is far better to handle the problem at compile time than runtime.

```
List list = new ArrayList();
list.add(“hello”);
list.add(32);//Compile Time Error
```

17. Write A Generic Method To Find The Maximal Element In The Range [begin, End) Of A List.

```
import java.util.*;
public final class Algorithm {
public static >
T max(List<? extends T> list, int begin, int end) {
T maxElem = list.get(begin);
for (++begin; begin < end; ++begin)
if (maxElem.compareTo(list.get(begin)) < 0)
maxElem = list.get(begin);
return maxElem;
}
}
```

18. Write A Program To Implement Lru Cache Using Generics ?

One hint is that LinkedHashMap can be used implement fixed size LRU cache where one needs to remove eldest entry when Cache is full. LinkedHashMap provides a method called removeEldestEntry() which is called by put() and putAll() and can be used to instruct to remove eldest entry. you are free to come up with your own implementation as long as you have a written a working version along with Unit test.

19. How Can We Restrict Generics To A Subclass Of Particular Class?

In MyListGeneric, Type T is defined as part of class declaration. Any Java Type can be used a type for this class. If we would want to restrict the types allowed for a Generic Type, we can use a Generic Restrictions. Consider the example class below: In declaration of the class, we specified a constraint "T extends Number". We can use the class MyListRestricted with any class extending (any sub class of) Number – Float, Integer, Double etc.

```
class MyListRestricted {
private List values;
void add(T value) {
values.add(value);
}
void remove(T value) {
values.remove(value);
}
T get(int index) {
return values.get(index);
}
}
```

```
MyListRestricted restrictedListInteger = new MyListRestricted();
restrictedListInteger.add(1);
restrictedListInteger.add(2);
String not valid substitute for constraint "T extends Number".
//MyListRestricted restrictedStringList =
//              new MyListRestricted();//COMPILER ERROR
```

20. If The Compiler Erases All Type Parameters At Compile Time, Why Should You Use Generics?

You should use generics because:

- The Java compiler enforces tighter type checks on generic code at compile time.
- Generics support programming types as parameters.
- Generics enable you to implement generic algorithms

21. Can You Give An Example Of A Generic Method?

A generic type can be declared as part of method declaration as well. Then the generic type can be used anywhere in the method (return type, parameter type, local or block variable type).

Consider the method below:

```
static X doSomething(X number){
X result = number;
//do something with result
return result;
}
```

The method can now be called with any Class type extend Number.

```
Integer i = 5;
Integer k = doSomething(i);
```

---- >> Below Are The Related Posts Of Above Questions :::

------>>[MOST IMPORTANT]<<------

1. **300+ TOP JAVA Generics Interview Questions and Answers**
2. **300+ [LATEST] Java Programmer Interview Questions and Answers**
3. **250+ TOP MCQs on Fundamental of Generics and Answers**

4. 300+ [LATEST] Java Abstraction Interview Questions and Answers
5. 300+ [LATEST] Java Inheritance Interview Questions and Answers
6. 300+ [LATEST] Java Inner Classes Interview Questions and Answers
7. 250+ TOP MCQs on Generics and Answers
8. 300+ [LATEST] Java Collections Interview Questions and Answers
9. 300+ [LATEST] Java Constructor Interview Questions and Answers
10. 250+ TOP MCQs on Restrictions on Generics and Answers
11. 300+ [LATEST] Java.util Interview Questions and Answers
12. 300+ [LATEST] Java Hadoop Developer Interview Questions and Answers
13. 300+ [LATEST] Java Garbage Collection Interview Questions and Answers
14. 300+ [LATEST] Java Programming With Oops Concepts Interview Questions and Answers
15. 300+ [LATEST] Java String Interview Questions and Answers
16. 300+ [LATEST] Java.lang Interview Questions and Answers
17. 300+ [LATEST] Java 9 Interview Questions and Answers
18. 300+ [LATEST] Java Webdynpro Interview Questions and Answers
19. 300+ [LATEST] Java Collections Framework Interview Questions and Answers
20. 250+ TOP MCQs on Type Interface and Answers