Purpose	Practice type parameter usage
What to Submit	You don't have to submit this problem.

Problem 1: Generics

1. Write a class named ku.util.ListUtil that has a static append method:

```
static <E> void append(List<E> list, List<E> args)
```

the method should append each element of the `args` List to the end of `list` (first parameter).

2. Write some code to verify that the method works if you have two lists of type Number:

```
List<Number> list = new ArrayList<Number>( );
// TODO add a BigInteger to list
List<Number> appendee = new ArrayList<Number>( );
// actual values in appendee are doubles
appendee.add( new Double(0.5) );
appendee.add( new Double(2.5) );
ListUtil.append( list, appendee );
//TODO print the list to verify it worked
```

3. It is OK to add doubles to the list in problem 2 because Double is a subclass of Number. So, we should be able to write:

```
List<Double> appendee = new ArrayList<Double>( );
// actual values in appendee are doubles
appendee.add( new Double(0.5) );
appendee.add( new Double(2.5) );
ListUtil.append( list, appendee );
```

Try this code to verify that it **won't compile**. What is the reason?

4. Modify the method signature on `append( )` to mean "args *is a List of elements from any subclass of E*". Hint: use a *wildcard* in the type parameter.

```
public static <E> void append( List<E> list, List<_______> args )
```

and verify that now problem 3 compiles and works as expected.

5. In Write a generic "max" method in ListUtil that returns the "maximum" of its arguments, and accepts a variable number of arguments (the ". . ." notation). The arguments can be any type that implements `Comparable<E>`

```
Double max = ListUtil.max( 1.5, 2.98E8, Math.PI );
     // max is Double(2.98E8)
String last = ListUtil.max( "ant", "cat", "zebra", "monkey");
     // max is "zebra"
```

6. The "max" method in the previous problem works for String because String implements Comparable<String>, and similarly for Double. But what about Coin from the Purse lab? Coin implements Comparable<Valuable> not Comparable<Coin>.

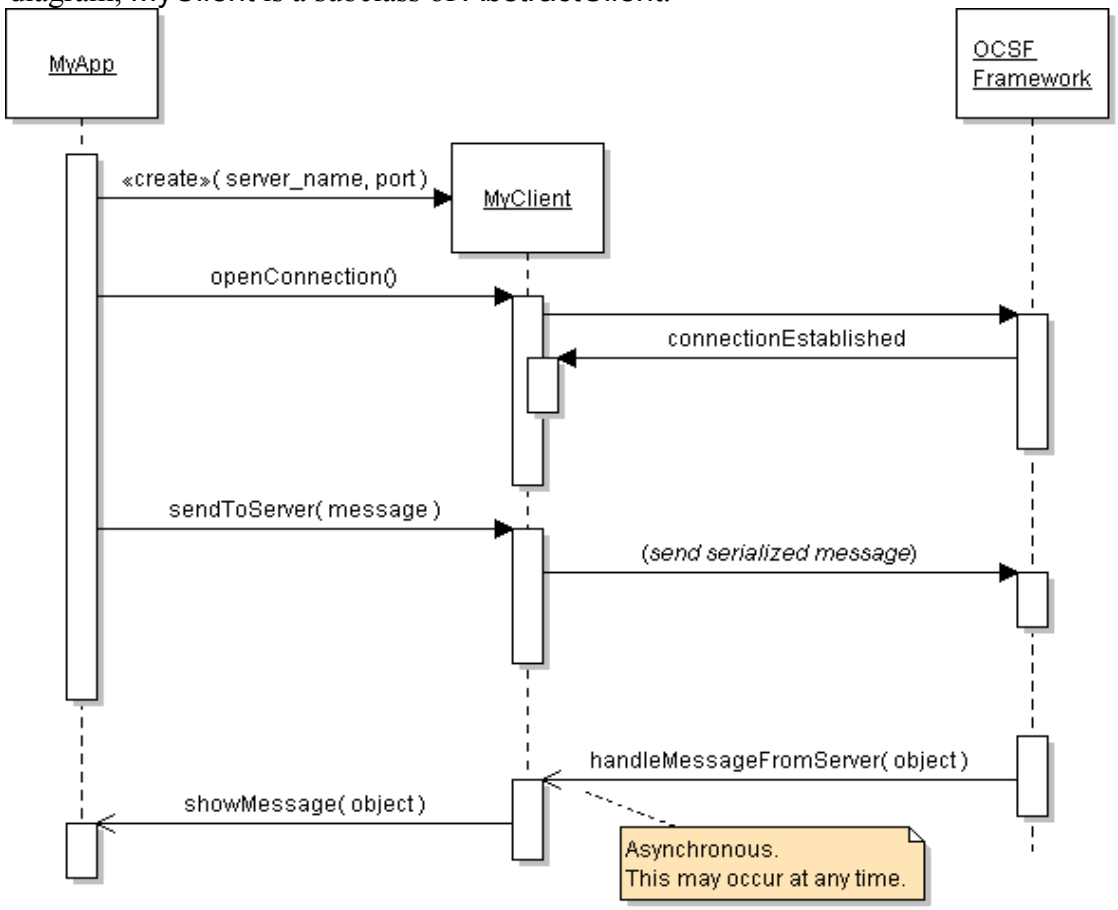
Try to invoke max(new Coin(5), new Coin(10), new Coin(100)) to verify this. Your Coin class must implement Comparable<Valuable> for tis problem.

Then correct the problem by completing this type declaration:

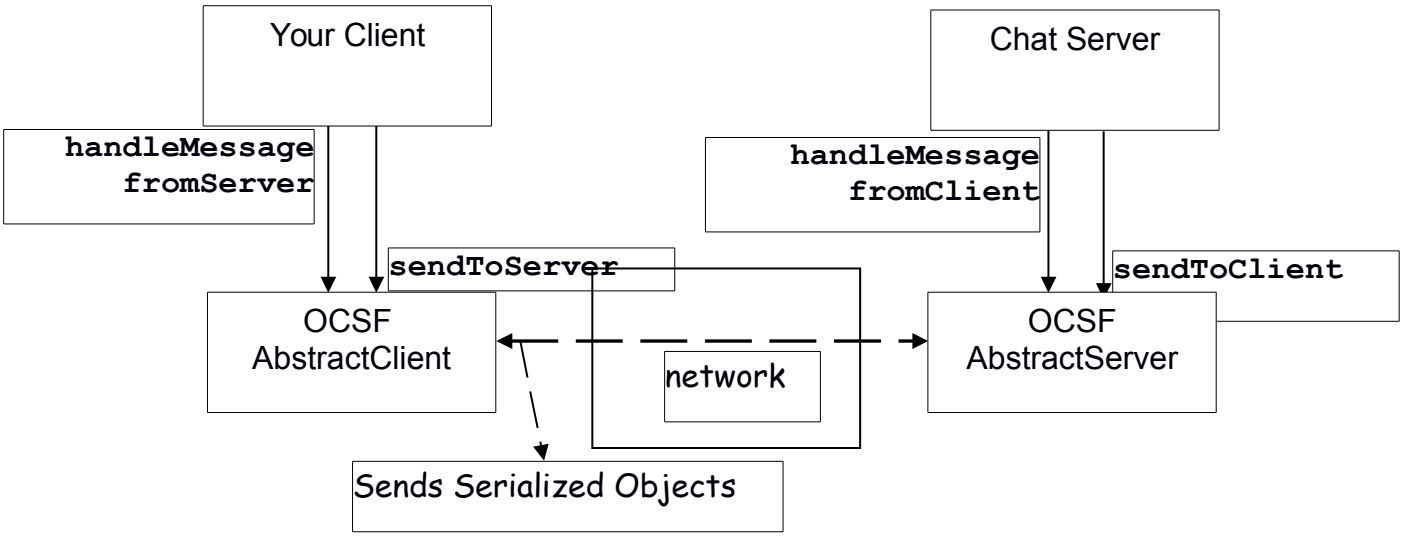
```
public static <E extends<? _______ E>> E max( E … args )
```

Typical Usage

In this diagram, MyClient is a subclass of AbstractClient.



Conceptual View of OCSF Operation



References

Lethbridge and Lagariere, *Object-Oriented Software Engineering,* 2E. Textbook describes use of OCSF and a chat project.

A standard, high-performance framework for chat and other applications is XMPP.

XMPP is a standard protocol for real-time messaging; XMPP was originally called *Jabber*. Google Talk uses XMPP. You can use XMPP to write your own Chat client or other Internet application. There are many several free XMPP servers (such as *Jabberd* and *OpenFire*), clients, and libraries. XMPP can be used for more than just chat.

SMACK is an open-source XMPP library for Java. It is used by several chat applications.
http://www.igniterealtime.org/projects/smack/

- How to use SMACK to write a Java client: http://www.javacodegeeks.com/2010/09/xmpp-im-with-smack-for-java.html
- Other two articles in the same series describe infrastructure for using XMPP.

XEP-0045 Multi-User Chat. Protocol for a multi-user chat using XMPP.
http://xmpp.org/extensions/xep-0045.html#bizrules-message