LAB 7

```
class Generics <T, U> {
    T ob1,
    U ob2;

   Generics (T x, U y) {
      ob1 = x;
      ob2 = y;
   }

   T getob1() {
      return ob1;
   }

   U getob2() {
      return ob2;
   }

  void display() {
     System.out.println(" ob1: " + getob1());
     System.out.println(" ob2: " +getob2());
  }

  U join() {
   if (ob1 instance of Integer && ob2 instanceof Integer) {
      int i1= (Integer)getob1();
      int i2 = (Integer) getob2();
      return (U) new Integer ~~in~~(i1 + i2);
   }
else if (ob1 instanceof Double && ob2 instanceof~~of~~ Double) {
    double d1= (Double) getob1();
    double d2= (Double) getob2();
    return (U) new Double (d1+d2);
}

else if (ob1 instanceof String && ob2 instanceof String) {
    String s1= (String) getob1();
    String s2= (String) getob2();
    return (U) new String (s1+s2);
}

else {
  return (U) new String ("ERROR! ob1 & ob2 type mismatch");
 }
}
}

class Gens {
 public static void main (String[] args) {
 Generics <Integer, Integer> iObj= new Generics <Integer, Integer>(5,4);
  iObj. display();
 System.out. println ("Sum: "+ iObj. join());
 Generics <Double, Double > ~~iob~~ dObj = new Generics <Double, Double> (3.05, 4.02);
  dObj. display ();
  System. out. println ("sum: "+ dObj. join ());
 Generics <Strigg, String> sObj = new Generics <String, String>
 sObj. display ();                          ("Hello, ", " how are you? ");
  System.out.prinln ("Concatanation: " + sObj. join ());
 }
}
```