Problem1:

```
public class Utils {
          public static <T extends Comparable<T> > int search ( T[] data , int n , T e) {
             for ( int i=0 ; i<n ; i++)
                if ( data[i].compareTo(e)== 0)
                   return i;

             return -1;
          }
 }
```

Problem2:

2.1:

```

 public class Utils {
       public static <T> int search (T[] data , int n , Condition<T> cond) {
             for ( int i=0 ; i<n ; i++)
                if (cond.test(data[i]))
                   return i;

             return -1;
          }


```

2.2:

```
          public static int searchEven(Integer[] data, int n ){
             Condition<Integer> cond = new Even();
             int index = search(data,n,cond);
             if(index!= -1)
                return data[index];

             return -1;

          }
 }

```

```

public class Even implements Condition<Integer> {


      public boolean test( Integer data) {
            return data%2==0;
      }
}
```

Problem3:

3.1:

```

public class GenericArray<T> {
      private T[] list;

      public GenericArray ( int n ) {
         list = (T[])new Object [n];
      }

      public T get ( int i ) {
         if ( i>=0 && i< list.length)
            return list[i];

         return null;
      }

      public void set ( int i , T val) {
         if ( i>=0 && i< list.length)
            list[i] = val;
      }

}

```

3.2:

When I compile the code an error message appears saying there is a problem at creating generic array:

```
ArrayOfBox.java:5: error: generic array creation
                              Box<String>[] b = new Box <String>[3];
```

So I solve this problem by using Generic Array class:

```
public class ArrayOfBox {

          public static void main (String [] args ) {
             /*Box<String>[] b = new Box <String>[3];

             b[0] = new Box<String>("A");
             b[0].update("b");*/

             GenericArray<Box<String>> b = new GenericArray <Box<String>> (3);
             Box<String> b1 = new Box<String>("A");
             b.set(0,b1);
             b1.update("b");
          }
       }
```

Problem4:

```

public class TestPile<T> {
   public static<T extends Comparable<T> > boolean compare(Pile<T> p1, Pile<T> p2){
      Pile<T> p3 = new Pile<T> (1000);
      Pile<T> p4 = new Pile<T> (1000);
      int p1Siza =0, p2Size = 0;
      T temp1, temp2;
      boolean flag = true;

      while (!p1.empty()){
         p3.add(p1.remove());
         p1Siza++;
      }
      while (!p2.empty()){
         p4.add(p2.remove());
         p2Size++;
      }

      if ( p2Size != p1Siza )//since they have different number of elements so they aren't equals
         return false;

      for (int i=0 ; i<p1Siza ; i++){
         temp1 = p3.remove();
         temp2 = p4.remove();
         if (!(temp1.compareTo(temp2)==0) )
            flag = false;
         p1.add(temp1);
         p2.add(temp1);
      }

      return flag;



   }
}

```