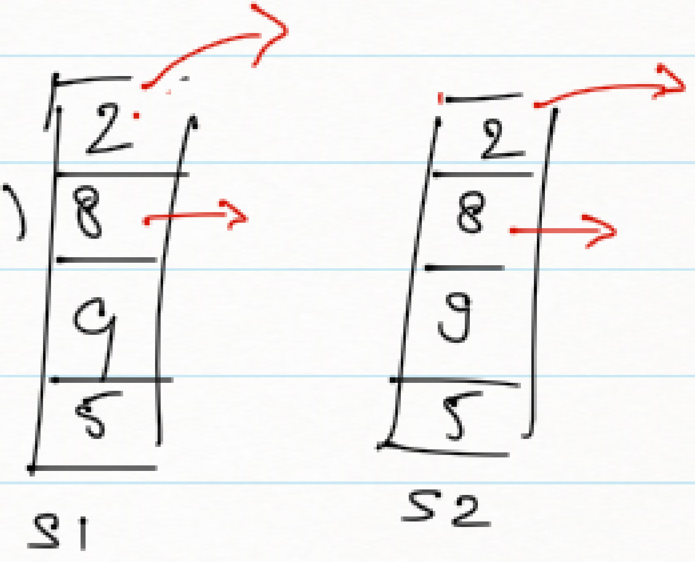
Compare 2 stacks.

```
boolean AreStacksEqual ( Stack<Integer> s1 , Stack<Integer> s2)
{
    if(s1. size() != s2. size())
        return false;

    while ( s1. size()>0)
    {
        int val1 = s1. pop();
        int val2 = s2.pop();

        if(val1 != val2)
            return false;
    }
    return true;
}
```



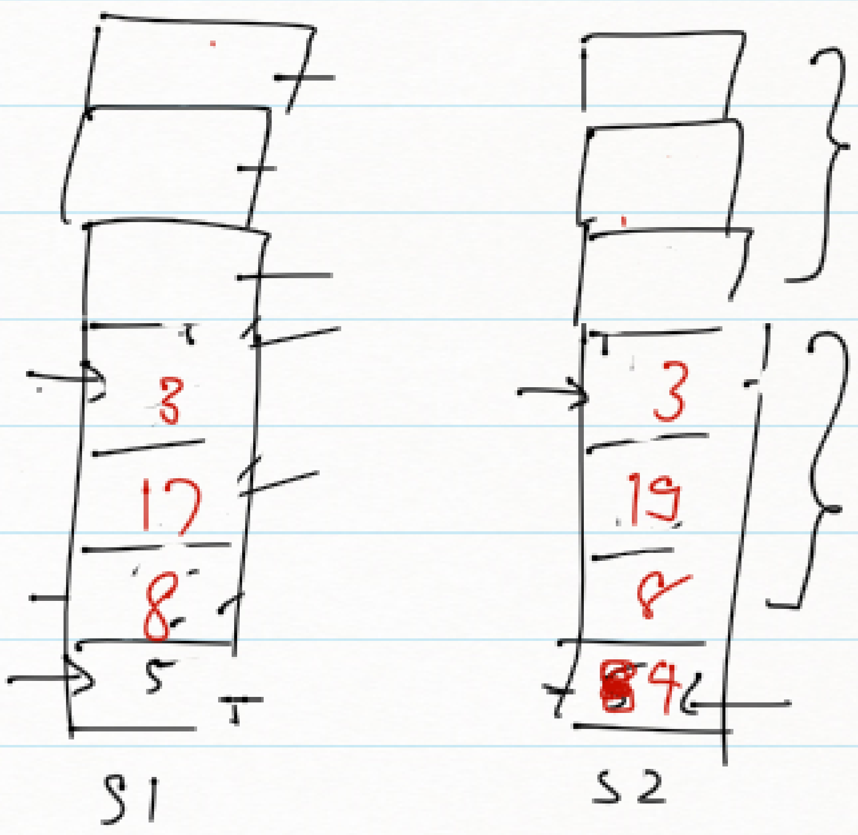
```
→  while ( ! s1. IsEmpty())
   {
   }
```

```
int a[]={1, 2, 3};
for(i=0; i<=2; i++)
{
    s.push(a[i]);
}
```

```
boolean AreStacksEqual(S1, S2)
{     boolean ans = true;
   if(s1.size() != s2.size())
        return false;

   int   n =  s1.size();

  for(int i=1; i<=n; i++)
  {
        shift(S1, S2, (n-i));
         int val= S1.top();     →   5

        shift(S2, S1, 2*(n-i));
         if(val != S2.top())
return false;      ans= false      ;
        shift(S1, S2, (n-i));     →  if(ans == false)
                                         return ans;
  }
} return true;
```

```
void shift (Stack(Integer)S1, Stack(Integer)S2
                                        , int k)
{
      int i=1;
      while( i<=k)
      {
           int val= S1.pop();
           S2.push(val);
           i++;
      }
}
```

ans = false.



Given an array, find next greater element to the right.

i

a =

5	4	9	12
0	1	2	3

5 → 9 ✓

4 → 9 ✓

9 → -1 ✓

12 → -1 ✓

i = 3

found = false.

```
int a[] = {5, 4, 9, 12}
for(i=0; i<=3; i++)
{       boolean found = false;
    for(j=i+1; j<=3; j++)
    {
        if( a[j] > a[i])
        {
                print   a[i] → a[j];
                found = true;
                break;
        }
    }
    if( found == false) { print   a[i] → -1 ; }
}
```

$O(n^2)$

2*n

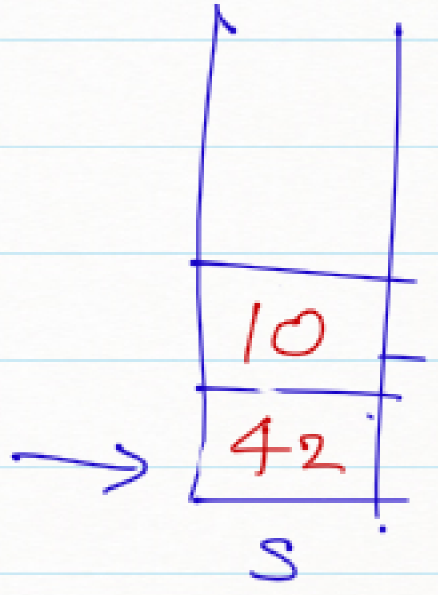
```
         0                 ↓        ↓
a[] = {  7 ,  5 ,  42  ,  10 };
Stack<Integer>  s  = new  Stack<>();

✓ s.push( a[0] );

for( int i=1; i<=3; i++)
{
      while(!s.IsEmpty()  &&  a[i] > s.top())
  →  {       int val = s.pop();
             print (   val   →   a[i]);
      }                                     } → pop
  → s.push( a[i] );
}
while(!s.IsEmpty())
  {       int val = s.pop();     print val → -1; }
```

5 → 42
7 → 42
10 → -1
42 → -1

2*n

O(n)

×



7 → 15
15 → 42
10 → -1
42 → -1

```
             0    1    2    3    4    5    6
int  a[] = { 2,   5,   7,   1,   9,   6,   3 };
stack<Integer>  s  = new  stack<>();
int   r[a.length];
s.push(a[a.length() - 1]);
r[a.length() - 1] =  -1;
                a.length - 2
for( int i=             ;  i >= 0; i--)
{
      while(!s.IsEmpty()  &&  a[i] > s.top())
      {
              s.pop();
      }
      if(s.IsEmpty())
      {       r[i] = -1;  }
      else
      {       r[i] =  s.top();  ✓
      }
}  push(a[i]);
```

r

5	7	9	9	-1	-1	-1
0	1	2	3	4	5	6