# Recursive Linear Search self-referencing

**Input:** integers $a_1, \ldots, a_n$
integer $n$
integer $i := 1$

good!

## Search (~~$a_i$~~ $x$)

integer $x$

```
{ if (a_i = x)
        return(i)

  if (i = n)
        return(0)
  i := i+1
  search (~~a_i+1~~, x)
}
```

Questions:

Is it necesary to assume a strictly increasing sequence $a_1 < a_2 < a_3 < \cdots < a_n < \cdots$ ?

What happens if for some $i, j$
$a_j < a_i$ but $i < j$?

What happens if for some $i$ $a_i = a_{i+1} = a_{i+2}$?

# Recursive self referencing
# Binary Search

**Input:** integers $a_1, a_2, a_3, \ldots, a_n, \ldots$ in increasing numeric order

integer $n$
integers $i := 1$, $j := n$

```
binary-search (x)
integer x
{ integer m
    m = ⌊(i+j)/2⌋

    if (x = am), return (m)
    if (x < am and i<m)
    { j := m-1
      binary-search(x)
    }
    if (x > am and j>m)
    { i := m+1
      binary-search (x)
    }
}
```

## Questions

1. Run the algorithm with $a_1 = 1, a_2 = 3, a_4 = 6, a_5 = 8$
$a_6 = 9, a_7 = 10, a_8 = 12$
2. Is it necessary to assume an increasing sequence $a_i < a_j$, for $i < j$? example when for some $i, j$ $i < j$ but $a_j < a_i$?
3. Does the algorithm work for $a_i = a_{i+1}$?

## Recursive-upward (iterative) for Fibonacci

```
fibonacci (n)
integer n
{ integers z, x:=0, y:=1, i
  if (n = 0), return(0)
  for (i:=1 , to n-1)
  {   z := x+y
      x := y
      y := z
      i = i+1
  }
  return (y)
```

### Running for n = 6

i=1
$z = 1$
$x = 1$
$y = 1$

i=2
$z = 2$
$x = 1$
$y = 2$

i=3
$z = 3$
$x = 2$
$y = 3$

i=4
$z = 5$
$x = 3$
$y = 5$

i=5
$z = 8$
$x = 5$
$y = 8$

i=6
$z = 13$
$x = 18$
$\boxed{y = 13}$