# Unique Algorithm



Main:

```
val list = listOf(1, 3, 1, 4, 1, 3, 5, 4, 5)
val uniqueList = unique(list)
for(elem in uniqueList)
    println(elem)
```

↳ pode ser uma lista, array, range

```
for(i in 0 until 10)        // range
    println(i) // 0, 1, ..., 9
```

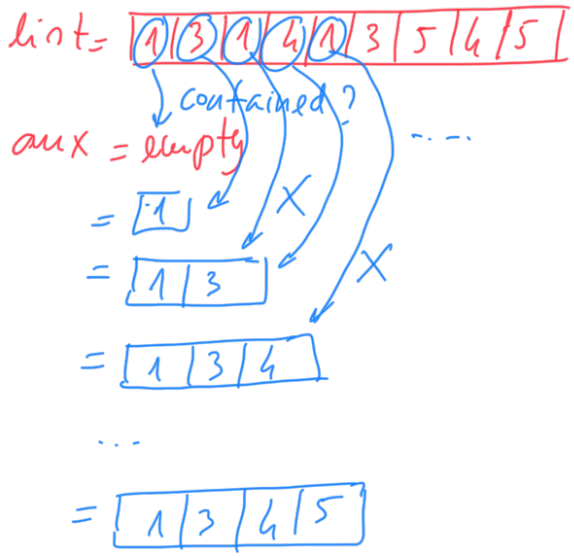
→ devido ao until ser Exclusive

```
for(i in 0..10)             // range
    println(i) // 0, 1, ..., 10
```

→ inclusive

```
for(i in 10 downTo 0)       // range
    println(i) // 10, 9, ..., 0
```

```
fun unique ( lint: Lint<Int> ) : Lint<Int> {
    // #1 - Use auxiliary list for repeated
    var aux = emptyLint<Int> ()
    var i = 0
    while ( i < lint.size ) {
        // check if repeated (using aux list)
        if ( lint[i] ! in aux )
            aux += lint[i]
        ++i
    }
    return aux
}
```

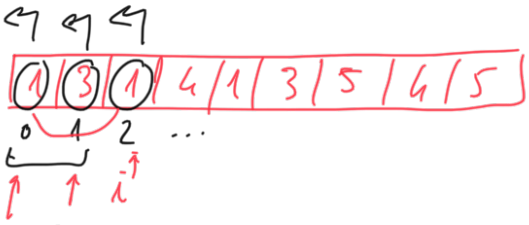


```
fun unique (...) -- {
    // #2 - search in the same list
    var aux = emptyList<Int>()
    var i=0 ; var found = false
    while (i < list.size)
        // search repeated in the same list
        // from beginning to the i-1
        for (j in 0..i-1)
            if (list[i] ==
                list[j]) {
                found = true
                break → sai do ciclo corrente
            }
        if (! found)
            aux += list[i]
        ++i
    }
    return aux
}
```

for(j in i-1 downTo 0)

<=>

for (j in 0 until i)




ou:

```
var j=0; var found=false
while (j <= i-1 && !found)
       ou j < i
    if(list[i] == list[j])
        found = true
        break // not necessary
    ++j
}
```