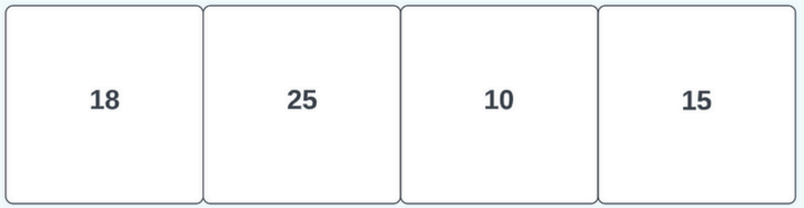
BIG O - ANÁLISIS COMPLEJIDAD TEMPORAL Y ESPACIAL

BIG O - EJERCICIOS

Daniel Blanco Calviño

EJEMPLO 1

```java
void reverseArray(int[] array) {
  for (int i = 0; i < array.length / 2; i++) {
    int reversePosition = array.length - 1 - i;
    int tmpVal = array[i];
    array[i] = array[reversePosition];
    array[reversePosition] = tmpVal;
  }
}
```



EJEMPLO 1

```java
void reverseArray(int[] array) {
  for (int i = 0; i < array.length / 2; i++) {
    int reversePosition = array.length - 1 - i;
    int tmpVal = array[i];
    array[i] = array[reversePosition];
    array[reversePosition] = tmpVal;
  }
}
```

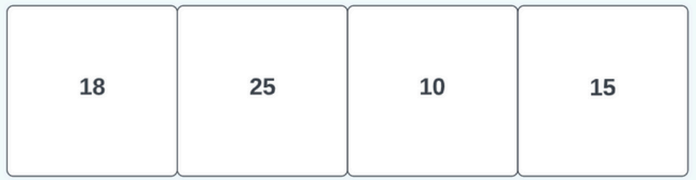
Complejidad temporal O(N)

EJEMPLO 2

```java
void printUnorderedPairs(int[] array) {
  for (int i = 0; i < array.length - 1; i++) {
    for (int j = i + 1; j < array.length; j++) {
      System.out.println(i + " " + j);
    }
  }
}
```

EJEMPLO 2

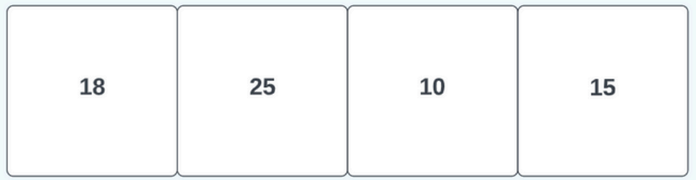
```java
void printUnorderedPairs(int[] array) {
  for (int i = 0; i < array.length - 1; i++) {
    for (int j = i + 1; j < array.length; j++) {
      System.out.println(i + " " + j);
    }
  }
}
```



EJEMPLO 2

```java
void printUnorderedPairs(int[] array) {
  for (int i = 0; i < array.length - 1; i++) {
    for (int j = i + 1; j < array.length; j++) {
      System.out.println(i + " " + j);
    }
  }
}
```

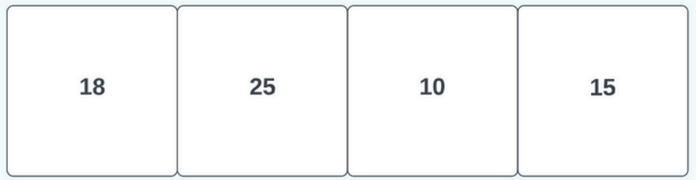
- 18,25
- 18,10
- 18,15



EJEMPLO 2

```java
void printUnorderedPairs(int[] array) {
  for (int i = 0; i < array.length - 1; i++) {
    for (int j = i + 1; j < array.length; j++) {
      System.out.println(i + " " + j);
    }
  }
}
```

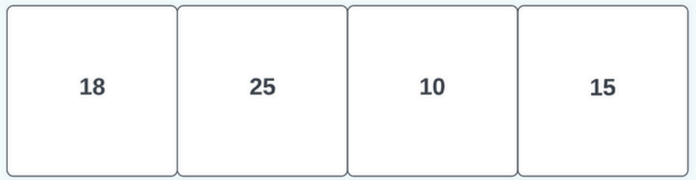
- 18,25
- 18,10
- 18,15
- 25,10
- 25,15



EJEMPLO 2

```java
void printUnorderedPairs(int[] array) {
  for (int i = 0; i < array.length - 1; i++) {
    for (int j = i + 1; j < array.length; j++) {
      System.out.println(i + " " + j);
    }
  }
}
```

- 18,25
- 18,10
- 18,15
- 25,10
- 25,15
- 10,15



EJEMPLO 2

```java
void printUnorderedPairs(int[] array) {
  for (int i = 0; i < array.length - 1; i++) {
    for (int j = i + 1; j < array.length; j++) {
      System.out.println(i + " " + j);
    }
  }
}
```

(N - 1) + (N - 2) + ... + 2 + 1 = N * (N - 1) / 2 **= O(N^2)**

EJEMPLO 3

```
1  int product(int a, int b) {
2    int result = 0;
3    for (int i = 0; i < b; i++) {
4      result += a;
5    }
6    return result;
7  }
```

EJEMPLO 3

```
1  int product(int a, int b) {
2    int result = 0;
3    for (int i = 0; i < b; i++) {
4      result += a;
5    }
6    return result;
7  }
```

Complejidad temporal O(b)

EJEMPLO 4

```java
int[] copyArray(int[] array) {
  int[] copy = new int[0];
  for (int val : array) {
    copy = appendToNew(copy, val);
  }
  return copy;
}

int[] appendToNew(int[] array, int val) {
  int[] bigger = new int[array.length + 1];
  for (int i = 0; i < array.length; i++) {
    bigger[i] = array[i];
  }

  bigger[bigger.length - 1] = val;
  return bigger;
}
```

EJEMPLO 4

```java
int[] copyArray(int[] array) {
  int[] copy = new int[0];
  for (int val : array) {
    copy = appendToNew(copy, val);
  }
  return copy;
}

int[] appendToNew(int[] array, int val) {
  int[] bigger = new int[array.length + 1];
  for (int i = 0; i < array.length; i++) {
    bigger[i] = array[i];
  }

  bigger[bigger.length - 1] = val;
  return bigger;
}
```

O(N)

O(N)

EJEMPLO 4

```java
int[] copyArray(int[] array) {
  int[] copy = new int[0];
  for (int val : array) {
    copy = appendToNew(copy, val);
  }
  return copy;
}

int[] appendToNew(int[] array, int val) {
  int[] bigger = new int[array.length + 1];
  for (int i = 0; i < array.length; i++) {
    bigger[i] = array[i];
  }

  bigger[bigger.length - 1] = val;
  return bigger;
}
```

O(N)

O(N)

O(N^2)

EJEMPLO 5

```java
void printPairsTwoArrays(int[] arrayA, int[] arrayB) {
  for (int valA : arrayA) {
    for (int valB : arrayB) {
      System.out.println(valA + " " + valB);
    }
  }
}
```

EJEMPLO 5

```
1  void printPairsTwoArrays(int[] arrayA, int[] arrayB) {
2    for (int valA : arrayA) {
3      for (int valB : arrayB) {
4        System.out.println(valA + " " + valB);
5      }
6    }
7  }
```

O(A)

O(B)

O(A * B)

EJEMPLO 6

```
1  int intPowerOfTwo(int val) {
2    int result = 0;
3    while (val > 1) {
4      result++;
5      val /= 2;
6    }
7    return result;
8  }
```

EJEMPLO 6

```
1  int intPowerOfTwo(int val) {
2    int result = 0;
3    while (val > 1) {
4      result++;
5      val /= 2;
6    }
7    return result;
8  }
```

O(log val)