Max Span

The maximal span of a sequence is the maximum span of a subsequence. To determine the Max-Span of an array, we can use the following algorithm:

1. Define a function that takes an array as input and returns the Max-Span.
2. For each element in the array, calculate the span of the subsequence ending at that element.
3. Keep track of the maximum span encountered so far.
4. Return the maximum span.

Example:

Given the array [1, 2, 1, 3], the Max-Span is 4.

Span

The span of an element in an array is the number of consecutive elements that include the element. To calculate the Max-Span, we can use the following algorithm:

1. Define a function that takes an array as input and returns the Max-Span.
2. For each element in the array, calculate the span of the subsequence ending at that element.
3. Keep track of the maximum span encountered so far.
4. Return the maximum span.

Example:

Given the array [1, 2, 1, 3], the Max-Span is 4.

Conclusion

We have learned how to calculate the Max-Span of an array. This algorithm can be used in various applications, such as finding the longest increasing subsequence in a given array.

References

- MaxSpan: A New Approach to Span Calculation
- Span: A Novel Span Calculation Method

We hope this lesson has been informative. Thank you for your attention.
## Test Case

- We will define a new class for the tests
- `il.ac.tau.cs.sw1.maxspan.tests.TestMaxSpan`

### First Part
- **Package**

### Now write the tests we want to check:

```java
int[] array = null;
int maxSpan;
array = new int[] {0, 1, 1, 1, 3};
maxSpan = MaxSpan.maxSpan(array);
if (maxSpan != 4) {
    System.out.println(Arrays.toString(array) + " expected: 4, result: " + maxSpan);
} else {
    System.out.println(Arrays.toString(array) + " correct!");
}
array = new int[] {1, 2, 1, 1, 1, 3};
maxSpan = MaxSpan.maxSpan(array);
if (maxSpan != 4) {
    System.out.println(Arrays.toString(array) + " expected: 4, result: " + maxSpan);
} else {
    System.out.println(Arrays.toString(array) + " correct!");
}
```

- Why is the debugger angry?
  - Does not know `Arrays`?
  ```java
  import java.util.Arrays;
  ```
  - Does not know `MaxSpan`?
  ```java
  import il.ac.tau.cs.sw1.maxspan.MaxSpan;
  ```
  But no such class...
  What to do?

### Bright Idea

- Let’s get an Eclipse (QuickFix)

#### Ctrl+1

**Q: What is the recommended course?**

- There are no packages
  ```java
  import java.util.Arrays;
  ```
  - There is no package `MaxSpan`
  ```java
  import il.ac.tau.cs.sw1.maxspan.MaxSpan;
  ```
  - Should we use the `Arrays` class instead?

### Code Refactoring

#### `Arrays.toString()`
- Can we improve the class name?
- Use `Refactor`

#### `Arrays.span()`
- Discussion: Writing the function "nicely"
- Efficiency
- Modularity, solution
- Understandability
- Top-down vs Bottom-up
- Possible changes
- Programmatic expressions

```java
public static int maxSpan(int[] nums) {
    int max = 0;
    for (int value : values(nums)) {
        max = Math.max(max, span(value, nums));
    }
    return max;
}
```
Holok Mapnikimut haturch

private static int span(int value, int[] num) {  return lastIndex(value, num) - firstIndexOf(value, num) + 1; }

private static int[] values(int[] nums) {  int[] values = new int[nums.length];  int nextIndex = 0;  for (int i = 0; i < nums.length; i++) {     if (!contains(values, nextIndex, nums[i])) {        add(values, nextIndex++, nums[i]);     }  }  return Arrays.copyOf(values, nextIndex); }

Meharatot

String Constructors

Use implicit constructor:

String s = "Hello";

Instead of:

String s = new String("Hello");
### StringBuilder

- **mutable**
- Represents a mutable sequence of characters
- Methods:
  - `append`
  - `insert`

```java
StringBuilder sb = new StringBuilder("123");
sb.append(4);  // "1234"
```

### StringBuffer vs. String

#### Inefficient version using String

```java
public static String duplicate(String s, int times) {
    String result = s;
    for (int i = 1; i < times; i++) {
        result = result + s;
    }
    return result;
}
```

#### More efficient version with StringBuilder

```java
public static String duplicate(String s, int times) {
    StringBuilder result = new StringBuilder(s);
    for (int i = 1; i < times; i++) {
        result.append(s);
    }
    return result.toString();
}
```

#### Even more efficient version:

```java
public static String duplicate(String s, int times) {
    StringBuilder result = new StringBuilder(s.length() * times);
    for (int i = 0; i < times; i++) {
        result.append(s);
    }
    return result.toString();
}
```