UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

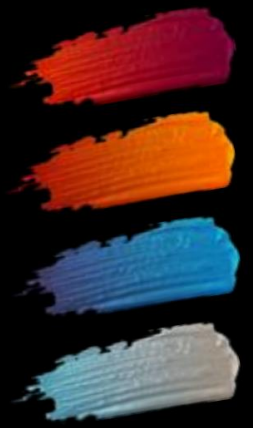
SERLAB
Software Engineering Research

Integrazione e Test di Sistemi Software

Esercizio con il cuscinetto sinistro

Azzurra Ragone

Dipartimento di Informatica - Università degli Studi di Bari
Via Orabona, 4 - 70125 - Bari
Tel: +39.080.5443270 | Fax: +39.080.5442536
serlab.di.uniba.it



Test basati sulle specifiche + test strutturali: un esempio

RICHIESTE:

Completa una stringa con una stringa specificata. Completa fino a raggiungere la dimensione di size. - str—Stringa da completare; può essere null.

- dimensione: la dimensione da raggiungere.

- padStr: la stringa con cui aggiungere il riempimento. Null o vuoto vengono trattati come un singolo spazio.

Il metodo restituisce una **stringa con riempimento a sinistra,** la stringa originale se non è necessario alcun riempimento oppure null se viene immessa una stringa nulla.

EX.

- str = 'abc'

-dimensione = 5

- padStr = 'X'

RISULTATO: 'XXabc'

LeftPad(): implementazione

```
public static String leftPad(final String str, final int size,
  String padStr) {

  if (str == null) {
    return null;
  }

  if (padStr==null || padStr.isEmpty()) {
    padStr = SPACE;
  }

  final int padLen = padStr.length();
  final int strLen = str.length();
  final int pads = size - strLen;

  if (pads <= 0) {
    // returns original String when possible
    return str;
  }

  if (pads == padLen) {
    return padStr.concat(str);
  } else if (pads < padLen) {
    return padStr.substring(0, pads).concat(str);
```

If the string to pad is null, we return null right away.

If the pad string is null or empty, we make it a space.

There is no need to pad this string.

If the number of characters to pad matches the size of the pad string, we concatenate it.

If we cannot fit the entire pad string, we add only the part that fits.

LeftPad(): implementazione

```
public static String leftPad(final String str, final int size,
  String padStr) {

  if (str == null) {
    return null;
  }

  if (padStr==null || padStr.isEmpty()) {
    padStr = SPACE;
  }

  final int padLen = padStr.length();
  final int strLen = str.length();
  final int pads = size - strLen;

  if (pads <= 0) {
    // returns original String when possible
    return str;
  }

  if (pads == padLen) {
    return padStr.concat(str);
  } else if (pads < padLen) {
    return padStr.substring(0, pads).concat(str);
```

If the string to pad is null, we return null right away.

There is no need to pad this string.

If the number of characters to pad matches the size of the pad string, we concatenate it.

If we cannot fit the entire pad string, we add only the part that fits.

Ex:

str = 'abc'
dimensione = 2
padStr = 'X' pad =
2-3<= 0
RISULTATO = 'abc'

LeftPad(): implementazione

```
public static String leftPad(final String str, final int size,
  String padStr) {

  if (str == null) {
    return null;
  }

  if (padStr==null || padStr.isEmpty()) {
    padStr = SPACE;
  }

  final int padLen = padStr.length();
  final int strLen = str.length();
  final int pads = size - strLen;

  if (pads <= 0) {
    // returns original String when po
    return str;
  }

  if (pads == padLen) {
    return padStr.concat(str);
  } else if (pads < padLen) {
    return padStr.substring(0, pads).concat(str);
```

If the string to pad is null, we return null right away.

If the pad string is

Ex:

str = 'abc'
dimensione = 4
padStr = 'X' pad = 4-3 = 1 padLen = 1

RISULTATO = 'Xabc'

s to
he
pad string, we concatenate it.

If we cannot fit the entire pad string, we add only the part that fits.

LeftPad(): implementazione

```
public static String leftPad(final String str, final int size,
  String padStr) {

  if (str == null) {
    return null;
  }

  if (padStr==null || padStr.isEmpty()) {
    padStr = SPACE;
  }

  final int padLen = padStr.length();
  final int strLen = str.length();
  final int pads = size - strLen;

  if (pads <= 0) {
    // returns original String when p
    return str;
  }

  if (pads == padLen) {
    return padStr.concat(str);
  } else if (pads < padLen) {
    return padStr.substring(0, pads).concat(str);
```

If the string to pad is null, we return null right away.

If the pad string is null or empty, we make it a space.

Ex:

str = 'abc' size = 4

padStr = 'XXX'

pads = 4-3 = 1 padLen = 3

RISULTATO = 'Xabc'

rs to he te it.

If we cannot fit the entire pad string, we add only the part that fits.

LeftPad(): implementazione

```
  } else {
    final char[] padding = new char[pads];
    final char[] padChars = padStr.toCharArray();

    for (int i = 0; i < pads; i++) {
      padding[i] = padChars[i % padLen];
    }

    return new String(padding).concat(str);
  }
}
```

We have to add the pad string more than once. We go character by character until the string is fully padded.

Es: pads>padLen

str = 'abc'

dimensione =

7 padStr = 'XY' pad =

7-3 = 4 padLen = 2

RISULTATO = 'XYXYabc'

Iniziamo con i test basati sulle specifiche

leftPad()

Tre input:
str = 'abc'
dimensione = 5
padStr = 'X'

Un'uscita: Stringa

Identificare le partizioni e la suite di test (20 minuti)

Azzurra Ragone

Dipartimento di Informatica - Piano VI - Stanza 616 Email: azzurra.ragone@uniba.it