Programação 2

Strings

Prof. Domingo Santos

domingos.santos@upe.br

UNIVERSIDADE DE PERNAMBUCO

Resumo

- Considerações iniciais
- Classe String
- Classes empacotadoras
- Tokenização de Strings

Conteúdo baseado no capítulo 14 do livro de Deitel

Considerações iniciais

- Um programa pode conter literais de caractere. Um literal de caractere é um valor inteiro representado como caractere entre aspas simples
- O valor de um literal de caractere é o valor inteiro do caractere no conjunto de caracteres Unicode
- String é uma sequência de caracteres tratada como uma única unidade entre aspas duplas

Representação ASCII:

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Classe String

Construtores String:

- Possibilidade de inicializar a string de formas diferentes

```
public class StringConstructors
{
   public static void main(String[] args)
   {
       char[] charArray = {'b', 'i', 'r', 't', 'h', ' ', 'd', 'a', 'y'};
       String s = new String("hello");

       // utiliza os construtores String
       String s1 = new String();
       String s2 = new String(s);
       String s3 = new String(charArray);
       String s4 = new String(charArray, 6, 3);

       System.out.printf("s1 = %s\ns2 = %s\ns3 = %s\ns4 = %s\n" , s1, s2, s3, s4);
   }
} // fim da classe StringConstructors
```

```
s1 =
s2 = hello
s3 = birth day
s4 = day
BUILD SUCCESSFUL (total time: 0 seconds)
```

Classe String

Métodos da classe:

- length: retornam o comprimento de uma String
- charAt: obtém o caractere em uma localização específica em uma String
- getChars: conjunto de caracteres de uma String como um array char, respectivamente.

```
public class StringMiscellaneous  {

   public static void main(String[] args) {

       String s1 = "java eh top";
       char[] charArray = new char[5];
       System.out.printf("s1: %s", s1);
       System.out.printf("%nLength of s1: %d", s1.length()); // testa o método length
       // faz loop pelos caracteres em s1 com charAt e os exibe na ordem inversa
       System.out.printf("%nThe string reversed is: " );
       for (int count = s1.length() - 1; count >= 0; count--) {
           System.out.printf("%c ", s1.charAt(count));
       }
       s1.getChars(0, 5, charArray, 0); // copia caracteres a partir de string para charArray
       System.out.printf("%nThe character array is: " );
       for (char character : charArray) {
           System.out.print(character);
       }
       System.out.println();
   }
}
```

```
s1: java eh top
Length of s1: 11
The string reversed is: p o t   h e   a v a j
The character array is: java
BUILD SUCCESSFUL (total time: 0 seconds)
```

Classe String

Comparando Strings

```
public class StringCompare {

   public static void main(String[] args) {
       String s1 = new String("hello"); // s1 é uma cópia de "hello"
       String s2 = "goodbye";
       String s3 = "Happy Birthday";
       String s4 = "happy birthday";
       System.out.printf(
               "s1 = %s%ns2 = %s%ns3 = %s%ns4 = %s%n%n", s1, s2, s3, s4);
       // teste para igualdade
       if (s1.equals("hello")) {
           System.out.println("s1 equals \"hello\"");
       } else {
           System.out.println("s1 does not equal \"hello\"");
       }

       if (s1 == "hello") {
           System.out.println("s1 is the same object as \"hello\"");
       } else {
           System.out.println("s1 is not the same object as \"hello\"");
       }
    }
}
```

```
s1 = hello
s2 = goodbye
s3 = Happy Birthday
s4 = happy birthday

s1 equals "hello"
s1 is not the same object as "hello"
```

Classe String

comparar strings menor e maior que

- Quando o computador compara Strings, na verdade ele compara os códigos numéricos dos caracteres nas Strings.

- compareTo:
  - retorna um valor inteiro que indica a ordem relativa das strings na ordem lexicográfica.
  - a ordem lexicográfica é a ordem alfabética baseada nos valores Unicode dos caracteres.
  - Ele retorna um valor negativo se a string atual for lexicograficamente menor que a string de comparação,
  - zero se forem iguais e
  - um valor positivo se a string atual for lexicograficamente maior.

Classe String

Comparando Strings

```
public class StringCompare {

   public static void main(String[] args) {
       String s1 = new String("hello"); // s1 é uma cópia de "hello"
       String s2 = "goodbye";
       String s3 = "Happy Birthday";
       String s4 = "happy birthday";
       // testa quanto à igualdade (ignora maiúsculas e minúsculas)
       if (s3.equalsIgnoreCase(s4))
       {
           System.out.printf("%s equals %s with case ignored%n", s3, s4);
       } else {
           System.out.println("s3 does not equal s4");
       }

       System.out.printf("%ns1.compareTo(s2) is %d", s1.compareTo(s2));
       System.out.printf("%ns2.compareTo(s1) is %d", s2.compareTo(s1));
       System.out.printf("%ns1.compareTo(s1) is %d", s1.compareTo(s1));
       System.out.printf("%ns3.compareTo(s4) is %d", s3.compareTo(s4));
       System.out.printf("%ns4.compareTo(s3) is %d%n%n", s4.compareTo(s3));
   }
}
```

Happy Birthday equals happy birthday with case ignored

s1.compareTo(s2) is 1
s2.compareTo(s1) is -1
s1.compareTo(s1) is 0
s3.compareTo(s4) is -32
s4.compareTo(s3) is 32

Classe String

Comparando Strings

```
public class StringCompare {

   public static void main(String[] args) {
       String s1 = new String("hello"); // s1 é uma cópia de "hello"
       String s2 = "goodbye";
       String s3 = "Happy Birthday";
       String s4 = "happy birthday";
       // testa regionMatches (distingue maiúsculas e minúsculas)
       if (s3.regionMatches(0, s4, 0, 5)) {
           System.out.println("First 5 characters of s3 and s4 match");
       } else {
           System.out.println(
                   "First 5 characters of s3 and s4 do not match");
       }
       // testa regionMatches (ignora maiúsculas e minúsculas)
       if (s3.regionMatches(true, 0, s4, 0, 5)) {
           System.out.println("First 5 characters of s3 and s4 match with case ignored");
       } else {
           System.out.println(
                   "First 5 characters of s3 and s4 do not match");
       }
   }
}
```

First 5 characters of s3 and s4 do not match

First 5 characters of s3 and s4 match with case ignored

Classe String

Métodos String, startsWith e endsWith

```
public class StringStartEnd {

   public static void main(String[] args) {
       String[] strings = {"started", "starting", "ended", "ending"};
       // testa o método startsWith
       for (String string : strings) {
           if (string.startsWith("st")) {
               System.out.printf("\"%s\" starts with \"st\"%n", string);
           }
       }
    }
}
```

"started" starts with "st"
"starting" starts with "st"

Classe String

Métodos String, startsWith e endsWith

```
public class StringStartEnd {

   public static void main(String[] args) {
       String[] strings = {"started", "starting", "ended", "ending"};

       // testa o método startsWith iniciando da posição 2 de string
       for (String string : strings) {
           if (string.startsWith("art", 2)) {
               System.out.printf(
                       "\"%s\" starts with \"art\" at position 2%n", string);
           }
       }
   }
}
```

```
"started" starts with "art" at position 2
"starting" starts with "art" at position 2
```

Classe String

Métodos String, startsWith e endsWith

```
public class StringStartEnd {

    public static void main(String[] args) {
        String[] strings = {"started", "starting", "ended", "ending"};

        // testa o método endsWith
        for (String string : strings) {
            if (string.endsWith("ed")) {
                System.out.printf("\"%s\" ends with \"ed\"%n", string);
            }
        }
    }
}
```

```
"started" ends with "ed"
"ended" ends with "ed"
```

Classe String

Localizando caracteres e substrings em strings

```
public class StringIndexMethods {

   public static void main(String[] args) {
      String letters = "abcdefghijklmabcdefghijklm";
      // testa indexOf para localizar um caractere em uma string
      System.out.printf("'c' is located at index %d%n", letters.indexOf('c'));
      System.out.printf("'a' is located at index %d%n", letters.indexOf('a', 1));
      System.out.printf("'$' is located at index %d%n%n", letters.indexOf('$'));

      // testa lastIndexOf para localizar um caractere em uma string
      System.out.printf("Last 'c' is located at index %d%n", letters.lastIndexOf('c'));
      System.out.printf("Last 'a' is located at index %d%n", letters.lastIndexOf('a', 25));
      System.out.printf("Last '$' is located at index %d%n%n", letters.lastIndexOf('$'));
   }
}
```

```
'c' is located at index 2
'a' is located at index 13
'$' is located at index -1

Last 'c' is located at index 15
Last 'a' is located at index 13
Last '$' is located at index -1
```

Classe String

Localizando caracteres e substrings em strings

```
public class StringIndexMethods {

   public static void main(String[] args) {
      String letters = "abcdefghijklmabcdefghijklm";
      // testa indexOf para localizar uma substring em uma string
      System.out.printf("\"def\" is located at index %d%n", letters.indexOf("def"));
      System.out.printf("\"def\" is located at index %d%n", letters.indexOf("def", 7));
      System.out.printf("\"hello\" is located at index %d%n%n", letters.indexOf("hello"));

      // testa lastIndexOf para localizar uma substring em uma string
      System.out.printf("Last \"def\" is located at index %d%n", letters.lastIndexOf("def"));
      System.out.printf("Last \"def\" is located at index %d%n", letters.lastIndexOf("def", 25));
      System.out.printf("Last \"hello\" is located at index %d%n" ,          letters.lastIndexOf("hello"));

   }
}
```

```
"def" is located at index 3
"def" is located at index 16
"hello" is located at index -1

Last "def" is located at index 16
Last "def" is located at index 16
Last "hello" is located at index -1
```

Classe String

Extraindo substrings de strings

```
public class SubString {

   public static void main(String[] args) {
       String letters = "abcdefghijklmabcdefghijklm";
       System.out.printf("Substring from index 20 to end is \"%s\"%n",
               letters.substring(20));
       System.out.printf("%s \"%s\"%n",
               "Substring from index 3 up to, but not including 6 is",
               letters.substring(3, 6));

   }
}
```

```
Substring from index 20 to end is "hijklm"
Substring from index 3 up to, but not including 6 is "def"
```

Classe String

Concatenando strings

```
public class StringConcatenation {
   public static void main(String[] args) {
       String s1 = "Amar o java é crime? ";
       String s2 = "Me prenda agora";
       System.out.printf("s1 = %s%ns2 = %s%n%n", s1, s2);
       System.out.printf("Result of s1.concat(s2) = %s%n", s1.concat(s2));
       System.out.printf("s1 after concatenation = %s%n", s1);
   }
}
```

```
s1 = Amar o java é crime?
s2 = Me prenda agora

Result of s1.concat(s2) = Amar o java é crime? Me prenda agora
s1 after concatenation = Amar o java é crime?
```

Classe String

Métodos de String diversos

```
public class StringMiscellaneous2 {

   public static void main(String[] args) {
       String s1 = "hello";
       String s2 = "GOODBYE";
       String s3 = "    spaces        ";

       System.out.printf("Replace 'l' with 'L' in s1: %s%n%n", s1.replace('l', 'L'));

       System.out.printf("s1.toUpperCase() = %s%n", s1.toUpperCase());
       System.out.printf("s2.toLowerCase() = %s%n%n", s2.toLowerCase());

       System.out.printf("s3 after trim = \"%s\"%n%n", s3.trim());

       char[] charArray = s1.toCharArray();
       System.out.print("s1 as a character array = ");

       for (char character : charArray) {
           System.out.print(character);
       }
       System.out.println();
   }
}
```

```
Replace 'l' with 'L' in s1: heLLo

s1.toUpperCase() = HELLO
s2.toLowerCase() = goodbye

s3 after trim = "spaces"

s1 as a character array = hello
```

Classe String

Método String ValueOf

```
public class StringValueOf {
   public static void main(String[] args) {
       char[] charArray = {'a', 'b', 'c', 'd', 'e', 'f'};
       boolean booleanValue = true;
       int integerValue = 7;
       float floatValue = 2.5f;
       double doubleValue = 33.333;

       System.out.printf("char array = %s%n", String.valueOf(charArray));
       System.out.printf("part of char array = %s%n", String.valueOf(charArray, 3, 3));
       System.out.printf("boolean = %s%n", String.valueOf(booleanValue));
       System.out.printf("int = %s%n", String.valueOf(integerValue));
       System.out.printf("float = %s%n", String.valueOf(floatValue));
       System.out.printf("double = %s%n", String.valueOf(doubleValue));
   }
}
```

```
char array = abcdef
part of char array = def
boolean = true
int = 7
float = 2.5
double = 33.333
```

Classes empacotadoras de tipo

O Java fornece oito classes empacotadoras de tipo: Boolean, Character, Double, Float, Byte, Short, Integer e Long

Permite que os valores de tipo primitivo sejam tratados como objetos

Possui uma série de métodos que facilitam a manipulação do dado

Classes empacotadoras de tipo

Classe Character

```
public class StaticCharMethods {

   public static void main(String[] args) {
       char c = '4';

       System.out.printf("is digit: %b%n", Character.isDigit(c));
       System.out.printf("is letter: %b%n", Character.isLetter(c));
       System.out.printf("is letter or digit: %b%n", Character.isLetterOrDigit(c));
       System.out.printf("is lower case: %b%n", Character.isLowerCase(c));
       System.out.printf("is upper case: %b%n", Character.isUpperCase(c));
       System.out.printf("to upper case: %s%n", Character.toUpperCase(c));
       System.out.printf("to lower case: %s%n", Character.toLowerCase(c));
   }
}
```

```
is digit: true
is letter: false
is letter or digit: true
is lower case: false
is upper case: false
to upper case: 4
to lower case: 4
```

Tokenização de Strings

```java
public class TokenTest {

    public static void main(String[] args) {
        String sentence = "Luffy, Zoro, Nami, Usopp, Sanji, Chopper, Robin, Franky, Brook, Jinbe";
        String[] tokens = sentence.split(",");
        System.out.printf("Number of elements: %d%nThe tokens are:%n", tokens.length);

        for (String token : tokens) {
            System.out.println(token.trim());
        }
    }
}
```

```
Number of elements: 10
The tokens are:
Luffy
Zoro
Nami
Usopp
Sanji
Chopper
Robin
Franky
Brook
Jinbe
```

exemplos de manipulação de string

Contagem de caracteres: Escreva um programa que conte o número de caracteres em uma string fornecida pelo usuário.

```java
import java.util.Scanner;

public class ContagemCaracteres {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Digite uma string: ");
        String input = scanner.nextLine();

        int count = input.length();
        System.out.println("Número de caracteres: " + count);
        scanner.close();
    }
}
```

exemplos de manipulação de string

Inversão de string: Crie um programa que inverta uma string, ou seja, se o usuário digitar "hello", o programa deve exibir "olleh".

```java
import java.util.Scanner;

public class InversaoString {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Digite uma string: ");
        String input = scanner.nextLine();

        String reversed = new StringBuilder(input).reverse().toString();
        System.out.println("String invertida: " + reversed);
        scanner.close();
    }
}
```

exemplos de manipulação de string

Remoção de espaços: Faça um programa que remova todos os espaços em branco de uma string fornecida pelo usuário

```java
import java.util.Scanner;

public class RemocaoEspacos {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Digite uma string com espaços: ");
        String input = scanner.nextLine();

        String noSpaces = input.replaceAll("\\s+", "");
        System.out.println("String sem espaços: " + noSpaces);
        scanner.close();
    }
}
```

exemplos de manipulação de string

Concatenação de strings: Escreva um programa que solicite duas strings do usuário e as concatene em uma única string.

```java
import java.util.Scanner;

public class ConcatenacaoStrings {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Digite a primeira string: ");
        String str1 = scanner.nextLine();
        System.out.print("Digite a segunda string: ");
        String str2 = scanner.nextLine();

        String concatenada = str1.concat(str2);
        System.out.println("Strings concatenadas: " + concatenada);
        scanner.close();
    }
}
```

exemplos de manipulação de string

Capitalização de palavras: Faça um programa que converta todas as palavras de uma string em letras maiúsculas ou minúsculas, conforme escolha do usuário.

```
import java.util.Scanner;

public class CapitalizacaoPalavras {
   public static void main(String[] args) {
       Scanner scanner = new Scanner(System.in);
       System.out.print("Digite uma frase: ");
       String frase = scanner.nextLine();

       System.out.println("Frase em maiúsculas: " + frase.toUpperCase());
       System.out.println("Frase em minúsculas: " + frase.toLowerCase());
       scanner.close();
   }
}
```