**Decimal para Binário**

/\*\*

\* Converte decimal para binário. A regra é ficar dividindo o valor por 2, pegar o resto de cada divisão e inserir o valor da direita para a esquerda na String de

\* retorno. O algoritmo é executado até que o valor que foi sucessivamente dividido se torne 0. Obs.: assume que o valor passado é inteiro positivo.

\* Exemplo: 13

\* 13/2 = 6 -> resto 1 -> Resultado: 1

\* 6/2 = 3  -> resto 0 -> Resultado: 01

\* 3/2 = 1  -> resto 1 -> Resultado: 101

\* 1/2 = 0  -> resto 1 -> Resultado: 1101

\*

\* Resultado: 1101

\*

\* @param valor número decimal a ser convertido

\* @return String contendo o valor em binário

\*/

public static String converteDecimalParaBinario(int valor) {

   int resto = -1;

   StringBuilder sb = new StringBuilder();

   if (valor == 0) {

      return "0";

   }

   // enquanto o resultado da divisão por 2 for maior que 0 adiciona o resto ao início da String de retorno

   while (valor > 0) {

      resto = valor % 2;

      valor = valor / 2;

      sb.insert(0, resto);

   }

   return sb.toString();

}

## Decimal para Hexadecimal

/\*\*

\* Converte decimal para hexadecimal. A regra é ficar dividindo o valor por 16, pegar o resto de cada divisão e inserir o valor da direita para a esquerda na String \* de retorno. Se o resultado da divisão for maior que 15, chamo o método através de recursão. O algoritmo é executado até que o valor que foi sucessivamente

\* dividido se torne 0. Obs.: assume que o valor passado é inteiro positivo.

\* Exemplo: 1279

\* 1279/16 = 79 -> resto 15 -> Resultado: F

\* 79/16 = 4    -> resto 15 -> Resultado: F

\* 4/16 = 0     -> resto 4  -> Resultado: 4

\*

\* Resultado: 4FF

\*

\* @param valor número decimal a ser convertido

\* @return String contendo o valor em hexadecimal

\*/

public static String converteDecimalParaHexadecimal(int valor) {

   char[] hexa = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};

   int resto = -1;

   StringBuilder sb = new StringBuilder();

   if (valor == 0) {

      return "0";

   }

   // enquanto o resultado da divisão por 16 for maior que 0 adiciona o resto ao início da String de retorno

   while (valor > 0) {

      resto = valor % 16;

      valor = valor / 16;

      sb.insert(0, hexa[resto]);

   }

   return sb.toString();

}

## Decimal para Octal

/\*\*

\* Converte decimal para octal. A regra é ficar dividindo o valor por 8, pegar o resto de cada divisão e inserir o valor da direita para a esquerda na String de

\* retorno. Se o resultado da divisão for maior que 7, chamo o método através de recursão. O algoritmo é executado até que o valor que foi sucessivamente dividido

\* se torne 0. Obs.: assume que o valor passado é inteiro positivo.

\* Exemplo: 1279

\* 1279/8 = 159 -> resto 7 -> Resultado: 7

\* 159/8 = 19   -> resto 7 -> Resultado: 7

\* 19/8 = 2     -> resto 3 -> Resultado: 3

\* 2/8 = 0       -> resto 2 -> Resultado: 2

\*

\* Resultado: 2377

\*

\* @param valor número decimal a ser convertido

\* @return String contendo o valor em octal

\*/

public static String converteDecimalParaOctal(int valor) {

   int resto = -1;

   StringBuilder sb = new StringBuilder();

   if (valor == 0) {

      return "0";

   }

   // enquanto o resultado da divisão por 8 for maior que 0 adiciona o resto ao início da String de retorno

   while (valor > 0) {

      resto = valor % 8;

      valor = valor / 8;

      sb.insert(0, resto);

   }

   return sb.toString();

}

## Binário para Decimal

/\*\*

\* Converte binário para decimal. A regra é ficar multiplicando, da direita para a esquerda, o valor binário por 2 elevado a um índice (começa em 0).

\* Obs.: assume que o valor passado é diferente de null e tem pelo menos um bit. Exemplo: '1101'

\* 1\*2^0 + 0\*2^1 + 1\*2^2 + 1\*2^3 = 13.

\*

\* Resultado: 13

\*

\* @param valorBinario String contendo o valor em binário

\* @return valor convertido em decimal

\*/

public static int converteBinarioParaDecimal(String valorBinario) {

   int valor = 0;

   // soma ao valor final o dígito binário da posição \* 2 elevado ao contador da posição (começa em 0)

   for (int i=valorBinario.length(); i>0; i--) {

      valor += Integer.parseInt(valorBinario.charAt(i-1)+"")\*Math.pow(2, (valorBinario.length()-i));

   }

   return valor;

}

## Binário para Hexadecimal

Obs.: o método converteBinarioParaDecimal(String) chamado na lógica está mais acima.

/\*\*

\* Converte binário para hexadecimal. A regra é converter grupos de 4 bits - da direita para a esquerda - em um caractere hexadecimal. Para facilitar,

\* converto o grupo de 4 caracteres em binário para decimal e o valor resultante será a posição do array de valores hexadecimais, cujo caractere será

\* retornado. Obs.: assume que o valor passado é diferente de null e tem pelo menos um bit. Exemplo: 11011011

\* Grupo 1: 1011 = 11 em decimal -> posição 11 do array de hexadecimais: 'b' -> Resultado: b

\* Grupo 2: 1101 = 13 em decimal -> posição 13 do array de hexadecimais: 'd' -> Resultado: db

\*

\* Resultado: db

\*

\* @param binario valor binário a ser convertido

\* @return valor em hexadecimal

\*/

public static String converteBinarioParaHexadecimal(String binario) {

   char[] hexa = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};

   StringBuilder sb = new StringBuilder();

   int posicaoInicial = 0;

   int posicaoFinal = 0;

   char caractereEncontrado = '-';

   // começa a pegar grupos de 4 dígitos da direita para a esquerda, por isso posicaoInicial e posicaoFinal ficam na posição de fim da String

   posicaoInicial = binario.length();

   posicaoFinal = posicaoInicial;

   while (posicaoInicial > 0) {

      // pega de 4 em 4 caracteres da direita para a esquerda. Se o último grupo à esquerda tiver menos de 4 caracteres, pega os restantes (1, 2 ou 3)

      posicaoInicial = ((posicaoInicial - 4) >= 0) ? posicaoInicial - 4 : 0;

      /\*

      \*  Transforma o grupo de 4 caracteres em um dígito hexadecimal. Primeiro converte o grupo de 4 (ou menos) caracteres em decimal e depois pega

      \*  o caractere correspondente no array de hexadecimais. Utilize o método converteBinarioParaDecimal(String) mais acima.

      \*/

      caractereEncontrado = hexa[converteBinarioParaDecimal(binario.substring(posicaoInicial, posicaoFinal))];

      // insere no início da String de retorno

      sb.insert(0, caractereEncontrado);

      posicaoFinal = posicaoInicial;

   }

   return sb.toString();

}

## Binário para Octal

Obs.: o método converteBinarioParaDecimal(String) chamado na lógica está mais acima.

/\*\*

\* Converte binário para octal. A regra é converter grupos de 3 bits - da direita para a esquerda - em um caractere octal. Para facilitar,

\* converto o grupo de 3 caracteres em binário para decimal, pois a lógica é a mesma. Obs.: assume que o valor passado é diferente de null

\* e tem pelo menos um bit. Exemplo: 11111011

\* Grupo 1: 011 = 3 em decimal e em octal -> resultado: 3

\* Grupo 2: 111 = 7 em decimal e em octal -> resultado: 73

\* Grupo 3: 11  = 3 em decimal e em octal -> resultado: 373

\*

\* Resultado: 373

\*

\* @param binario valor binário a ser convertido

\* @return valor em octal

\*/

public static String converteBinarioParaOctal(String binario) {

   StringBuilder sb = new StringBuilder();

   int posicaoInicial = 0;

   int posicaoFinal = 0;

   // começa a pegar grupos de 3 dígitos da direita para a esquerda, por isso posicaoInicial e posicaoFinal ficam na posição de fim da String

   posicaoInicial = binario.length();

   posicaoFinal = posicaoInicial;

   while (posicaoInicial > 0) {

      // pega de 3 em 3 caracteres da direita para a esquerda. Se o último grupo à esquerda tiver menos de 3 caracteres, pega os restantes (1 ou 2)

      posicaoInicial = ((posicaoInicial - 3) >= 0) ? posicaoInicial - 3 : 0;

      /\*

      \* Converte binário para decimal (de 0 a 7) que vai ser o mesmo caractere em octal e insere no início da String de retorno. Utilize o método

      \* converteBinarioParaDecimal(String) mais acima.

      \*/

      sb.insert(0, converteBinarioParaDecimal(binario.substring(posicaoInicial, posicaoFinal)));

      posicaoFinal = posicaoInicial;

   }

   return sb.toString();

}

## Hexadecimal para Binário

Obs.: o método converteDecimalParaBinario(int) chamado na lógica está mais acima.

/\*\*

\* Converte hexadecimal para binário. A regra é converter cada caractere em um grupo de 4 bits - da esquerda para a direita. Para facilitar, converto

\* o caractere em decimal pegando o índice dele no array de hexadecimais. Depois converto o decimal em um grupo de 4 bits e insiro no início da

\* String de retorno. Obs.: assume que o valor passado é diferente de null e tem pelo menos um dígito.

\* Exemplo: 7d7

\* 7 -> 111  -> Ajuste: 0111 -> Resultado: 0111

\* d -> 1101 -> Ajuste: 1101 -> Resultado: 01111101

\* 7 -> 111  -> Ajuste: 0111 -> Resultado: 011111010111

\*

\* Resultado: 11111010111

\*

\* @param valorHexa valor hexadecimal a ser convertido

\* @return valor em binário

\*/

public static String converteHexadecimalParaBinario(String valorHexa) {

   int posicao = 0;

   int posicaoArrayHexa = -1;

   StringBuilder sb = new StringBuilder();

   String valorConvertidoParaBinario = null;

   char[] hexa = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};

   // enquanto tem caracteres em hexa para converter

   while (posicao < valorHexa.length()) {

      // pega a posição do caractere no array de caracteres hexadecimais

      posicaoArrayHexa = Arrays.binarySearch(hexa, valorHexa.charAt(posicao));

      // pega o valor em decimal (correspondente à posição do caractere no array de hexadecimais) e converte para binário

      valorConvertidoParaBinario = converteDecimalParaBinario(posicaoArrayHexa);

      /\*

      \*  Se o valor convertido para binário tem menos de 4 dígitos, complementa o valor convertido com '0' à esquerda até

      \*  ficar com 4 dígitos, se não for o caractere mais à esquerda do valor em hexadecimal

      \*/

      if (valorConvertidoParaBinario.length() != 4 && posicao > 0) {

         for (int i=0; i < (4-valorConvertidoParaBinario.length()); i++) {

            sb.append("0");

         }

      }

      sb.append(valorConvertidoParaBinario);

      posicao++;

   }

   return sb.toString();

}

## Hexadecimal para Decimal

/\*\*

\* Converte hexadecimal para decimal. A regra é ficar multiplicando, da direita para a esquerda, o valor binário por 16 elevado a um índice (começa em 0).

\* Obs.: assume que o valor passado é diferente de null e tem pelo menos um dígito.

\* Exemplo: 'a91'

\* 1\*16^0 + 9\*16^1 + 10\*16^2 = 2705.

\*

\* Resultado: 2705

\*

\* @param valorHexa String contendo o valor em hexadecimal

\* @return valor convertido em decimal

\*/

public static int converteHexadecimalParaDecimal(String valorHexa) {

   int valor = 0;

   int posicaoCaractere = -1;

   char[] hexa = {'0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'a', 'b', 'c', 'd', 'e', 'f'};

   // soma ao valor final o dígito binário da posição \* 16 elevado ao contador da posição (começa em 0)

   for (int i=valorHexa.length(); i>0; i--) {

      posicaoCaractere = Arrays.binarySearch(hexa, valorHexa.charAt(i-1));

      valor += posicaoCaractere\*Math.pow(16, (valorHexa.length()-i));

   }

   return valor;

}

## Hexadecimal para Octal

Obs.: os métodos converteHexadecimalParaDecimal(String) e converteDecimalParaOctal(int) chamados na lógica estão mais acima.

/\*\*

\* Converte hexadecimal para octal. Não existe algoritmo direto, portanto deve-se fazer

\* uma conversão para decimal e depois de decimal para octal. Utilize os métodos

\* converteHexadecimalParaDecimal(String) e converteDecimalParaOctal(int) mostrados

\* anteriormente.

\* Exemplo:

\* hexadecimal 'a91' -> convertido para decimal: 2705

\* decimal 2705  -> convertido para octal: 5221

\*

\* Resultado: 5221

\*

\* @param valorHexa String contendo o valor em hexadecimal

\* @return valor convertido em octal

\*/

public static String converteHexadecimalParaOctal(String valorHexa) {

   int valorDecimal = converteHexadecimalParaDecimal(valorHexa);

   return ConversaoOctal.converteDecimalParaOctal(valorDecimal);

}

## Octal para Binário

/\*\*

\* Converte octal para binário. A regra é converter cada caractere em um grupo de 3 bits - da esquerda para a direita. Converto o decimal em um

\* grupo de 3 bits e insiro no início da String de retorno. Obs.: assume que o valor passado é diferente de null e tem pelo menos um dígito.

\* Exemplo: 1275

\* 1 -> 001 -> Resultado: 001

\* 2 -> 010 -> Resultado: 001010

\* 7 -> 111 -> Resultado: 001010111

\* 5 -> 101 -> Resultado: 001010111101

\*

\* Resultado: 1010111101

\*

\* @param valorOctal valor octal a ser convertido

\* @return valor em binário

\*/

public static String converteOctalParaBinario(String valorOctal) {

   int posicao = 0;

   StringBuilder sb = new StringBuilder();

   String valorConvertidoParaBinario = null;

   while (posicao < valorOctal.length()) { // pega o valor em decimal (correspondente ao caractere da posição) e converte para binário

      valorConvertidoParaBinario = converteDecimalParaBinario(Integer.parseInt(valorOctal.charAt(posicao)+""));

      /\* Se o valor convertido para binário tem menos de 3 dígitos, complementa o valor convertido com '0' à esquerda até

       \* ficar com 3 dígitos, se não for o caractere mais à esquerda do valor em octal

       \*/

      if (valorConvertidoParaBinario.length() != 3  && posicao > 0) {

         for (int i=0; i < (3-valorConvertidoParaBinario.length()); i++) {

            sb.append("0");

     }

      }

      sb.append(valorConvertidoParaBinario);

      posicao++;

   }

   return sb.toString();

}

## Octal para Decimal

/\*\*

\* Converte octal para decimal. A regra é ficar multiplicando, da direita para a esquerda, o valor binário por 8 elevado a um índice (começa em 0).

\* Obs.: assume que o valor passado é diferente de null e tem pelo menos um dígito.

\* Exemplo: '127'  \* 7\*8^0 + 2\*8^1 + 1\*8^2 = 87.

\*

\* Resultado: 87

\*

\* @param valorOctal String contendo o valor em octal

\* @return valor convertido em decimal

\*/

public static int converteOctalParaDecimal(String valorOctal) {

   int valor = 0;

   // soma ao valor final o dígito binário da posição \* 8 elevado ao contador da posição (começa em 0)

   for (int i=valorOctal.length(); i>0; i--) {

      valor += Integer.parseInt(valorOctal.charAt(i-1)+"")\*Math.pow(8, (valorOctal.length()-i));

   }

   return valor;

}

## Octal para Hexadecimal

Obs: os métodos converteOctalParaDecimal(String) e converteDecimalParaHexadecimal(int) chamados na lógica estão mais acima.

/\*\*

\* Converte octal para hexadecimal. Não existe algoritmo direto, portanto deve-se fazer

\* uma conversão para decimal e depois de decimal para hexadecimal. Utilize os métodos

\* converteOctalParaDecimal(String) e converteDecimalParaHexadecimal(int) mostrados

\* anteriormente.

\* Exemplo:

\* octal 5025 -> convertido para decimal: 2581

\* decimal 2581  -> convertido para hexadecimal: 'a15'

\*

\* Resultado: a15

\*

\* @param valorOctal String contendo o valor em octal

\* @return valor convertido em hexadecimal

\*/

public static String converteOctalParaHexadecimal(String valorOctal) {

   int valorDecimal = converteOctalParaDecimal(valorOctal);

   return converteDecimalParaHexadecimal(valorDecimal);

}