**Algoritmos de Mudança de Base**

***Mário Leite***

***...***

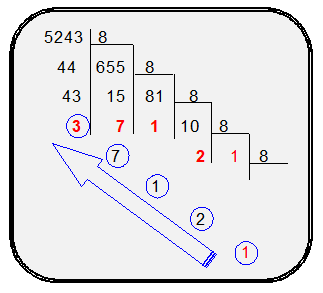
Mudança de base numérica é um assunto muito importante no ambiente da computação numérica, e todos os programadores deveriam saber como mostrar um valor numérico em uma determinada base desejada. Isto é importante em três casos bases :

* Raciocínio lógico: ajuda a entender diferentes formas de representar a mesma ideia.
* Tecnologia: essencial para programação, eletrônica e ciência da computação.
* Cultura matemática: mostra como sistemas diferentes resolvem problemas práticos.

Considerando a nossa base numérica, Decimal, o normal é mostrar o valor de um número dessa nossa base para outra qualquer; mas podemos fazer a mudança de uma base qualquer para outra.

1. **Da Base decimal para outra.**

A mudança da Base Decimal para uma outra base qualquer pode ser feita com o “*método do caminho inverso dos restos das divisões sucessivas*”. Neste método são feitas divisões sucessivas do número na base de origem (base **10**) pela base de destino (base **B**) até que o dividendo seja menor que esta base **B**; e quando isto acontecer o último resto será o primeiro dígito do número. Então, o número procurado será formado pelos restos das divisões escritos do fim para o início - *caminho inverso* - de modo que o primeiro resto será o último dígito do número na base desejada, e o último resto o primeiro dígito desse número. Por exemplo,mudar para a base **8** o número decimal **5243**. Empregando o “*método do caminho inverso dos restos das divisões sucessivas*”, observe como fica o processo no esquema da **figura 1**.



**Figura 1 - Esquema do método das divisões sucessivas**

Pelo esquema da **figura 1** pode ser observado os seguintes passos:

1ª Divisão: 5243/8

Dividendo: 655

Resto: **3**

2ª Divisão: 655/8

Dividendo: 81

Resto: **7**

3ª Divisão: 81/8

Dividendo: 10

Resto: **1**

4ª Divisão: 10/8

Dividendo: 1

Resto: **2**

5ª Divisão: 1/8 ==> como o dividendo neste caso é **1** (menor que a base 8), então este dividendo passa a ser o último resto que faltava. Assim, usando o *caminho inversos dos restos* teremos **1 2 1 7 3**; e pode-se afirmar que 5243(**10**) = 12173(**8**).

Outro exemplo: Fazer a mudança de base do número **164** para a base **2**. Empregando o método acima descrito, obtém-se **1 0 1 0 0 1 0 0**, como demonstrado no esquema abaixo.

164 2

**0** 82 2

**0** 41 2

**1** 20 2

**0** 10 2

**0** 5 2

**1** 2 2

**0**

**(1)**

Seguindo o caminho inverso das divisões sucessivas, e considerando o último dividendo (1) como o primeio dígito o resultado é: **164**(**10**) = **1 0 1 0 0 1 0 0**(**2**).

1. **De uma base qualkquer para outra**

Agora, vamos fazer a mudança de base: **FACA(16)** para a base **2**. Neste caso é melhor converter **FACA(16)** - base hezadecimal - para a base decimal e em seguida para a base binária para obter o que foi solicitado. Assim, usando o conceito de notação posicional de um dígito em um número, e considerando que:

* F(16) = 15
* A(16) = 10
* C(16) = 12

Pode ser considerada a seguinte soma de produtos:

**15\*163 + 10\*162 + 12\*161 + 10\*160 = 64202(10)**

Agora é possível converter **64202(10)** para a base **2** através do método do “*caminho inverso dos restos das divisões sucessivas*”, como é apresentado a seguir...

64202 2

**0** 32101 2

**1** 16050 2

**0** 8025 2

**1**  4012 2

**0** 2006 2

**0** 1003 2

**1**  501 2

**1** 250 2

**0** 125 2

**1** 62 2

**0** 31 2

**1** 15 2

**1** 7 2

**1** 3 2

**1**  **(1)**

Como o último dividendo é menor que a base: o número procurado é:

**1 1 1 1 1 0 1 0 1 1 0 0 1 0 1 0** (**2**)

Neste caso foram necessários dois *bytes* para escrever o número pedido na Base Binária:

**FACA(16) = 64202(10) = 1 1 1 1 1 0 1 0 1 1 0 0 1 0 1 0 (2)**

Como pode ser observado, a mudança para a base **2** através do “*método do caminho inverso dos restos das divisões sucessivas*” pode ser muito trabalhoso; no caso foram necessárias muitas divisões, pois o número a ser convertido era muito “grande”. Entretanto, quando a base de saida for potência da base de chegada podemos usar o fato de que, por exemplo, **16 = 24**. Deste modo, pode-se considerar que cada dígito da base **16** equivale a quatro dígitos da base **2**; então, assim pode ser escrito:

**F = 1111; A = 1010; C = 1100; A =1 010 ==> FACA**

Portanto, é possível “montar” o número pesquisado concatenando diretamente os blocos de dígitos da base **2** acima descritos como: **1111 1010 1100 1010**, que é o resultado encontrado nos restos das divisões sucessivas. Por outro lado, é importante saber que um número hexadecimal também pode ser representado seguido de um **H**: 160**H**, 123**H**, 23AB5**H**, FACA**H**, etc.

O programa **“BaseXParaBaseY”**, em Visualg, é uma solução para mudar de uma base qualquer para outra quaquer na faixa: **<=2 B <= 16**, cuja saída pode ser vista na **figura 2**.

-----------------------------------------------------------------------------------------------------------------------------

**Nota**: Postagem baseada no livro: “*Curso Básico de Programação: Teoria e Prática*”.

Publicado pelo autor na “Amazon”, “Clube de Autores” e “Editora Ciência Moderna”.

Cliquen o *link* abaixo para a cessar explicações sobre o livro:

<https://www.amazon.com.br/Curso-B%C3%A1sico-Programa%C3%A7%C3%A3o-Teoria-Pr%C3%A1tica/dp/8539908700>

**Algoritmo** **"BaseXParaBaseY"**

**//Programa para converter um número de uma base X para outra base Y**

**//Autor : Mário Leite**

**//E-mail : marleite@gmail.com**

//---------------------------------------------------------------------

**Var** i, j, Q, T, N, ND, B1, B2, Dif: **inteiro**

BS, NB, NS, NumB1, Resto: **caractere**

VetNX: **vetor**[1..32] de **caractere**

VetD, VetRN, VetNN: **vetor**[1..32] **de** **inteiro**

R, Resp, Acabou: **logico**

**Inicio**

**LimpaTela**

B1 <- 0

B2 <- 0

*{Loop para garantir entradas de bases válidas para o programa}*

**Enquanto** ((B1<2) **ou** (B1>16) **ou** (B1=10)) **Faca**

**Escreva**("Entre com a base numérica de origem: ")

**Leia**(B1)

**FimEnquanto**

**Enquanto** ((B2<2) **ou** (B2>16) **ou** (B2=10)) **Faca**

**Escreva**("Entre com a base numérica de destino: ")

**Leia**(B2)

**FimEnquanto**

*{Entrada do número para as conversões}*

BS <- **NumpCarac**(B1)

**Escreva**("Entre com o número na base ",BS, ": ")

**Leia**(NS)

**Escreval**("")

NumB1 <- **Maiusc**(NS) *//converte letra para maiúscula*

NS <- **Maiusc**(NS)

T <- **Compr**(NS) *//pega o tamanho da string*

*{Verifica coerência do número com sua base}*

**Se**(B1<10) **Entao** //(2<=B1<10)

**Para** j **De** 1 **Ate** T **Faca**

VetNX[j] <- **Copia**(NS,j,1);

**Se**((**Asc**(VetNX[j])>=48) **e** (**Asc**(VetNX[j])<=57)) **Entao** //é número

VetNN[j] <- **CaracpNum**(VetNX[j])

**Se**(VetNN[j]<B1) **Entao**

R <- **Verdadeiro**

**Senao**

R <- **Falso**

**FimSe**

**Senao**

R <- **Falso**

**FimSe**

**FimPara**

**Se**(R) **Entao** *//todos os elementos do número coerentes com a base* ***B1***

{Define o número expandindo a "Notação Posicional"}

ND <- 0

**Para** j **De** 1 **Ate** T **Faca**

VetNX[j] <- **Copia**(NS,j,1);

VetNN[j] <- **CaracpNum**(VetNX[j])

ND<-ND + **Int**(VetNN[j]\*B1^(T-j)) //monta número na base decimal

**FimPara**

**FimSe**

**Senao** //(**10<B1<=16**)

**Para** j **De** 1 **Ate** T **Faca** //verifica se tem elemento estranho à base

VetNX[j] <- **Copia**(NS,j,1)

**Se**((**Asc**(VetNX[j])>=48) **e** (**Asc**(VetNX[j])<=57)) **Entao** //é número

R <- **Verdadeiro**

**Senao**

**Se**((**Asc**(VetNX[j])>=65)**e**(**Asc**(VetNX[j])<=90)) **Entao** //é letra

**Se**((**Asc**(VetNX[j])>=65) **e** (**Asc**(VetNX[j])<=70)) **Entao**

R <- **Verdadeiro** //letra válida

**Senao**

R <- **Falso** //letra inválida

**FimSe**

**FimSe**

**FimSe**

**FimPara**

**Se**(R) **Entao** //número é coerente com a base

ND <- 0

**Para** j **De** 1 **Ate** T **Faca**

**Escolha** VetNX[j]

**Caso** "A"

VetNN[j] <- 10

**Caso** "B"

VetNN [j]) <- 11

**Caso** "C"

VetNN[j] <- 12

**Caso** "D"

VetNN[j] <- 13

**Caso** "E"

VetNN[j] <- 14

**Caso** "F"

VetNN[j] <- 15

**OutroCaso**

VetNN[j] <- **CaracPNum**(VetNX[j])

**FimEscolha**

ND <- ND + **Int**(VetNN[j]\*B1^(T-j))

**FimPara**

**Senao**

R <- Falso

**FimSe**

**FimSe**

**Se**(R=Falso) **Entao**

**Escreval**("O número ", NS, " não existe na base ", BS)

**Senao**

{Converte **ND(10)** na base de destino **B2**}

N <- ND

NS <- **NumpCarac**(N)

T <- **Compr**(NS)

Resp <- **Verdadeiro**

{Loop para verificar os dígitos do número digitado}

**Para** j **De** 1 **Ate** T **Faca**

VetNX[j] <- **Copia**(NS,j,1);

**Se**(**Asc**(VetNX[j])<48) o**u** (**Asc**(VetNX[j])>57) **Entao**

Resp <- **Verdadeiro**

**Fimse**

**FimPara**

{Verifica base e número e decide se faz as divisões sucessivas}

**Se**((Resp) **e** (B2>10) **e** ((N>9) **e** (N<B2))) **Entao** //não precisa

Dif <- B2-N

**Escolha** Dif

**Caso** 1

NB <- "F"

**Caso** 2

NB <- "E"

**Caso** 3

NB <- "D"

**Caso** 4

NB <- "C"

**Caso** 5

NB <- "B"

**Caso** 6

NB <- "A"

**FimEscolha**

**Senao**

{É preciso fazer as divisões sucessivas}

j <- 1

Acabou <- **Falso**

**Enquanto** (**Nao**(Acabou)) **Faca**

VetD[j] <- **Int**((N/B2))

Q <- VetD[j]

VetRN[j] <- (N **Mod** B2)

**Se**(VetD[j]<B2) **Entao**

j <- j + 1

VetRN[j] <- Q

Acabou <- **Verdadeiro** //não precisa mais dividir

**Senao**

N <- VetD[j]

j <- j + 1

**FimSe**

**FimEnquanto** //fim do loop das divisões sucessivas

{Define o número na base **B2**}

NB <- " "

**Para** i **De** j **Ate** 1 **Passo** -1 **Faca** //loop inverso para os restos

**Escolha** VetRN[i]

**Caso** 10

**Resto** <- "A"

**Caso** 11

Resto <- "B"

**Caso** 12

Resto <- "C"

**Caso** 13

Resto <- "D"

**Caso** 14

Resto <- "E"

**Caso** 15

Resto <- "F"

**OutroCaso**

Resto <- **NumpCarac**(VetRN[i])

**FimEscolha**

NB <- NB + Resto //monta o número como os restos

**FimPara**

{Resultado do processamento}

**Se**(Resp) **Entao**

**Escreval**("Mudança de base")

**Escreval**(NumB1,"(",**NumpCarac**(B1),") =", NB, "(",**NumpCarac**(B2),")")

**Senao**

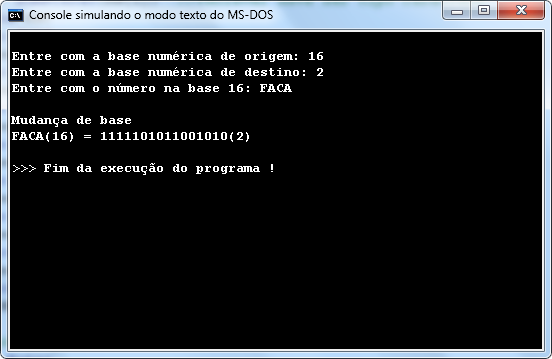
**Escreval**("O número ",NS," não é válido para esta mudança de base")

**Fimse**

**Fimse**

**FimSe**

**FimAlgoritmo** **//Fim do programa "BaseXParaBaseY"**



**Figura 2 - Saída do programa BaseXParaBaseY: Mudança de Base 16 para Base 2**