Mudança de Base

Mário Leite

...

Métodos para representar números e efetuar operações são conhecidos desde a Antiguidade; portanto, o homem já vem adotando sistemas de numeração há muito tempo. Por exemplo, de acordo com relatos históricos o **Sistema de Base 60** é creditado aos babilônios em função da hora ser dividida em 60 minutos e o minuto em 60 segundos.

O **Sistema Romano** - baseado nas letras **I**, **V**, **X**, **L**, **C**, **D** e **M** - não foi muito utilizado por apresentar muitas dificuldades operacionais; sua prática se tornou inviável, pois, para representar unidades de milhares era preciso colocar tracinhos acima da letra **M**, correspondendo a cada milhar.

O **Sistema Hindu** foi o mais funcional e prático, pois, utilizava nove símbolos para representar os dígitos; e mais tarde deu um salto histórico entre os sistemas numéricos ao introduzir o *zero*.

O emprego do *zero* no sistema numérico foi fundamental para a matemática; uma arma poderosíssima para os matemáticos e para toda a humanidade, pois, foi a partir do conceito do "nada" é que se estabeleceu o conceito de "notação posicional", permitindo definir o chamado "valor de posição" de um dígito dentro do número. Este conceito traduz-se no seguinte: "*a posição de um dígito em um número determina o seu valor quantitativo nesse número*". Deste modo, quando uma pessoa diz que possui uma coleção de **401** discos de vinil, **1** significa uma unidade, **0** indica que não existem dezenas e o **4** quatro centenas de discos; por isto, é importante o conceito de "notação posicional" para compreender bem o **Sistema Decimal**. A literatura sobre mudanças de base é vasta; na Internet existem sistemas automáticos que permitem a mudança de base numérica entre as três mais empregadas: *decimal*, *hexadecimal* e *binária*. A **tabela 1** mostra alguns valores numéricos nessas três tabelas: de **0** até **21**, mas, essa sequência pode ser estendida infinitamente, bastando aplicar algoritmos simples de conversão de uma base para outra.

Os chamados "conversores automáticos" encontrados na Internet mostram os resultados das conversões nestas três bases citadas (não em várias); entretanto, permanece a pergunta: COMO FAZEM ISTO!? Embora mostrem o resultado, esses "conversores" não explicam como chegar a eles. Nestes casos, é preciso mais do que uma simples aplicação da "Notação Posicional" da base decimal, ou mesmo da noção de "Progressão Geométrica" de razão **2** para o caso da base binária.

O programa "**BaseXParaBaseY**" (em pseudocódigo) é uma solução para a mudança de uma base **X** qualquer para uma outra base **Y** qualquer, com X e Y limitadas de **2** a **16**, apenas por conveniência. O programa recebe um valor numa base original e o converte para a base deseja.

```
Programa "ConvereteBaseXParaBaseY"
//Programa para converter um número de uma base X para outra base Y
//Autor: Mário Leite-
//-------------------------------------------------------------------------------
    Declare VetNX: array[1..32] de caractere
       VetD, VetRN, VetNN: array [1..32] de inteiro
       i, j, Q, T, N, ND, B1, B2, Dif: inteiro
       BS, NB, NS, NumB1, Resto: caractere
       R, Resp, Acabou: lógico
Início
   LimpaTela
   B1 ← 0
   B2 ← 0
   Enquanto ((B1<2) ou (B1>16)) Faça
      Escreva("Entre com a base numérica de origem: ")
      Leia(B1)
   FimEnquanto
   Enquanto ((B2<2) ou (B2>16)) Faça
      Escreva("Entre com a base numérica de destino: ")
      Leia(B2)
   FimEnquanto
   BS ← NumCarac(B1)
   Escreva("Entre com o número na base ",BS, ": ")
   Leia(NS)
   EscrevaLn("")  //salta linha
   NumB1 ← Maiusc(NS)  //converte letra para maiúscula
   NS ← Maiusc(NS)
   T  ← Tamanho(NS)  //pega o tamanho da string
   Se(B1<10) Então  //(2<=B1<10)
      Para j De 1 Até T Faça
         VetNX[j] ← Copia(NS,j,1);
         Se((Asc(VetNX[j])>=48) e (Asc(VetNX[j])<=57)) Então  //é número
            VetNN[j] ← CaracNum(VetNX[j])
            Se(VetNN[j]<B1) Então
               R ← .V.
            Senão
               R ← .F.
            FimSe
         Senão
            R ← .F. //não é número
         FimSe
      FimPara
      Se(R) Então  //todos os elementos do número coerentes com a base B1
         ND ← 0
         Para j De 1 Até T Faça
            VetNX[j] ← Copia(NS,j,1);
            VetNN[j] ← CaracNum(VetNX[j])
            ND <- ND + Int(VetNN[j]*B1^(T-j))  //monta o número na base decimal
         FimPara
      FimSe
   Senão  //(10=<B1<=16)
      Para j De 1 Até T Faça //verifica se tem elemento estranho à base
         VetNX[j] ← Copia(NS,j,1)
         Se((Asc(VetNX[j])>=48) e (Asc(VetNX[j])<=57)) Então  //é número
            R ← .V.
         Senão
            Se((Asc(VetNX[j])>=65) e (Asc(VetNX[j])<=90)) Então  //é letra
               Se((Asc(VetNX[j])>=65) e (Asc(VetNX[j])<=70)) Então
                 R ← .V. //letra válida
```

```
            Senão
               R ← .F. //letra inválida
            FimSe
         FimSe
       FimSe
    FimPara
    Se(R) Então  //número é coerente com a base
       ND ← 0
       Para j De 1 Até T Faça
          Selecione 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
             CasoContrário
               VetNN[j] ← CaracNum(VetNX[j])
          FimSelecione
          ND ← ND + Int(VetNN[j]*B1^(T-j))
       FimPara
    Senão
       R ← .F.
    FimSe
 FimSe
 EscrevaLn("")
 EscrevaLn("")
 Se(R=.F.) Então
    EscrevaLn("O número ", NS, " não existe na base ", BS)
 Senão  //converte ND(10) na base de destino B2
    N <- ND
    NS ← NumCarac(N)
    T  <- Tamanho(NS)
    Resp ← .V.
    Para j De 1 Até T Faça
       VetNX[j] <- Copia(NS,j,1);
       Se(Asc(VetNX[j])<48) ou (Asc(VetNX[j])>57) Então
         Resp ← .V.
       Fimse
    FimPara
    Se((Resp) e (B2>10) e ((N>9) e (N<B2))) Então //não precisa fazer divisões
       Dif ← B2-N
       Selecione 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"
      FimSelecione
   Senão  //é preciso fazer as divisões sucessivas
      j ← 1
      Acabou <- .F.
      Enquanto (Nao(Acabou)) Faça
         VetD[j] ← Int((N/B2))
         Q ← VetD[j]
         VetRN[j] ← (N mod B2)
         Se(VetD[j]<B2) Então
            j ← j + 1
            VetRN[j] ← Q
            Acabou ← .V.  //não precisa mais dividir
         Senão
            N ← VetD[j]
            j ← j + 1
         FimSe
      FimEnquanto  //fim do loop das divisões sucessivas
      NB <- " "
      Para i De j Até 1 Passo -1 Faça  //loop inverso para os restos
         Selecione 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"
           CasoContrário
             Resto ← NumCarac(VetRN[i])
         FimSelecione
         NB ← NB + Resto  //monta o número como os restos
      FimPara
      Se(Resp) Então
         EscrevaLn("Mudança de base")
         EscrevaLn(NumB1,"(",NumCarac(B1),")=",NB,"(",NumCarac(B2),")")
      Senão
         EscrevaLn(" O número ",NS," não é válido para esta mudança de base")
      Fimse
   Fimse
FimSe
FimPrograma  //Fim do programa
```

Decimal	Hexadecimal	Binária
0	0	00000000
1	1	00000001
2	2	00000010
3	3	00000011
4	4	00000100
5	5	00000101
6	6	00000110
7	7	00000111
8	8	00001000
9	9	00001001
10	A	00001010
11	B	00001011
12	C	00001100
13	D	00001101
14	E	00001110
15	F	00001111
16	10	00010000
17	11	00010001
18	12	00010010
19	13	00010011
20	14	00010100
21	15	00010101

Tabela 1 - Alguns valores nas três bases mais conhecidas