

MMC de vários números

Mário Leite

...

Eu já havia comentado sobre o **MMC**, mostrando uma aplicação prática com este elemento da Matemática, que nos é apresentado, inicialmente, no Curso Fundamental, na soma de frações. E quando se trata de frações com o mesmo denominador, basta somar os numeradores e manter o denominador, como por exemplo: $1/3 + 2/3 = 3/3 = 1$. Mas, quando se trata de somar frações com denominadores diferentes fica um pouquinho mais complexo; e mais ainda, quando estão envolvidas mais de duas frações. Entretanto, nestes casos, basta achar o MMC dos denominadores, dividi-lo por cada denominador, multiplicar o resultado de cada divisão pelo respectivo numerador e considerá-lo como o denominador de cada fração; e finalmente, aplicar a regra da soma de frações com denominadores iguais.

O programa "**ProgMMC-VariosNumeros**" (codificado em Visualg - que pode ser convertido para qualquer linguagem) é uma solução geral para somar várias frações; não apenas duas. Este programa usa a função "**FunMMC**" que calcula e retorna o MMC de cada dupla de números lidos no programa principal. Note que para calcular o MMC foi utilizado o MDC, pois, estes dois elementos mantêm uma certa "simbiose" entre si. E a solução, *pensada, planejada e programada* (ANTES DA CODIFICAÇÃO), baseou no seguinte raciocínio lógico: para qualquer quantidade de números, o MMC pode ser calculado em duplas de números.

MMC(n1, n2, n3, n4) = **MMC**(**MMC**(n1, n2), (**MMC**(3, 4))) ==> para quantidade par de números.

MMC(n1, n2, n3, n4, n5) = **MMC**(**MMC**(X), n5), onde X = **MMC**(n1, n2, n3, n4) ==> para quantidade impar de números.

Este, e outros 1000 (mil) programas estão disponíveis no meu livro "**1001 Programas Prontos Para Você Codificar Na Sua Linguagem Preferida**".

Para adquirir o *pdf/e-book* deste livro ou o *pdf* de outros livros sobre programação, entre em contato pelo e-mail: **marleite@gmail.com**

Algoritmo "ProgMMC-VariosNumeros"

//Calcula e mostra o MMC de vários números

//Em Visualg

//Autor: Mário Leite

//-----

//Programa principal

//Elementos globais

Const MAXELE=10 //limita a quantidade de números

Var VetNum, VetMMC: **vetor**[1..MAXELE] **de inteiro**
j, k, n, Qte, QteNum, UltNum, MMC, MMC1, MMC2: **inteiro**

Inicio

Repita

Escreva("Digite a quantidade de números: [min 2 -max",MAXELE,"]: ")

Leia(QteNum)

Ate((QteNum>=2) **e** (QteNum<=MAXELE))

Qte <- QteNum //preserva a quantidade original de números lidos

Para j **De** 1 **Ate** QteNum **Faca** //inicialmente zera os elementos do vetor

VetNum[j] <- 0

FimPara

Escreval("") //apenas salta uma linha

Para j **De** 1 **Ate** QteNum **Faca**

VetNum[j] <- 0

Repita

Escreva("Digite o número #", j, ": ")

Leia(VetNum[j])

Ate(VetNum[j]>0) //garante um número não negativo

FimPara

UltNum <- 0

Se(Qte Mod 2 <> 0) **Entao** //quantidade de números lida é impar

UltNum <- VetNum[Qte] //último número do vetor **VetNum**[]

QteNum <- Qte - 1 //garante pares de números para os **MMCs** parciais

FimSe

Se(QteNum=2) **Entao**

MMC <- **FunMMC**(VetNum[1],VetNum[2]) //chama função para calcular o **MMC**

Senao //calcula o **MMC** para cada par de números da lista lida

Para j **De** 1 **Ate** QteNum **Passo** 2 **Faca**

n <- j + 2

Se(n>QteNum) **Entao** //índice ultrapassou a quantidade de números

Interrompa //sai do loop incondicionalmente

FimSe

k <- j + 1

Se(k>=4) **Entao** //corte para atualizar **MMC1** com **MMC**: a partir da segunda dupla

MMC1 <- MMC

Senao

MMC1 <- **FunMMC**(VetNum[j],VetNum[k])

FimSe

MMC2 <- **FunMMC**(VetNum[n],VetNum[k+2])

MMC <- **FunMMC**(MMC1,MMC2) //MMC de dupla sequencial dos números

FimPara

FimSe //fim do teste de decisão de como calcular o **MMC**

Se(UltNum<>0) **Entao** //resta um número a ser computado}

MMC <- **FunMMC**(MMC,UltNum) //calcula o **MMC** final

FimSe

Escreval("")

Escreva("MMC(")

{Mostra o **MMC** da lista de números lidos}

Para j **De** 1 **Ate** Qte **Faca**

Se(j<Qte) **Entao**

Escreva(VetNum[j], ",")

Senao

Escreva(VetNum[j], ") =")

FimSe

FimPara

Escreval(MMC)

FimAlgoritmo //fim do programa principal

```

//-----
Funcao FunMMC (Num1, Num2: inteiro): inteiro
//Calcula o MMC de uma dupla de números em função do MDC
    Var MDC, MMCx, N1, N2, N1x, N2x, Aux: inteiro
Inicio
    {Inicializações convenientes para entrar no loop de validação}
    N1 <- Abs(Int(Num1)) //garante parâmetro inteiro e positivo
    N2 <- Abs(Int(Num2))
    N1x <- N1
    N2x <- N2
    {Loop para calcular o MDC de uma dupla de números}
    Enquanto (N2<>0) Faca
        Aux <- N1
        N1 <- N2
        N2 <- (Aux Mod N2)
    FimEnquanto
    MDC <- N1
    MMCx <- Int(N1x*N2x/MDC)
    Retorne MMCx
FimFuncao //fim da função "FunMMC"
//-----

```

```

C:\ Console simulando o modo texto do MS-DOS

Digite a quantidade de números: [min 2 -max 10]: 4

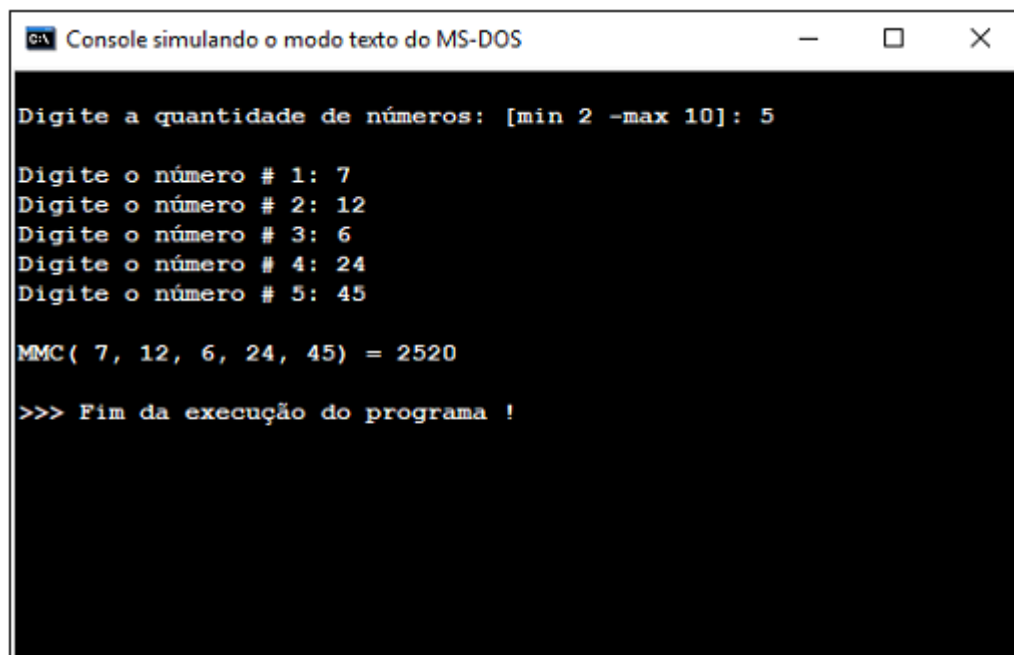
Digite o número # 1: 6
Digite o número # 2: 7
Digite o número # 3: 12
Digite o número # 4: 24

MMC( 6, 7, 12, 24) = 168

>>> Fim da execução do programa !

```

Figura 1 - MMC de uma quantidade par de números



```
c:\> Console simulando o modo texto do MS-DOS

Digite a quantidade de números: [min 2 -max 10]: 5

Digite o número # 1: 7
Digite o número # 2: 12
Digite o número # 3: 6
Digite o número # 4: 24
Digite o número # 5: 45

MMC( 7, 12, 6, 24, 45) = 2520

>>> Fim da execução do programa !
```

Figura 2 - MMC de uma quantidade ímpar de números