Solução da Exponenciação Eficiente: Projeto Euler

Prof. Jeferson Souza, MSc. (thejefecomp)

thejefecomp@neartword.com

NeartWord

Exponenciação Eficiente - Projeto Euler

Descrição do Problema

A forma mais simples de realizar o cálculo de n^{15} necessita de quatorze (14) multiplicações:

$$n \times n \ldots \times n = n^{15}$$

Ao usar um método "binário", pode-se realizar o mesmo cálculo em seis (6) multiplicações:

$$n \times n = n^2$$
$$n^2 \times n^2 = n^4$$
$$n^4 \times n^4 = n^8$$
$$n^8 \times n^4 = n^{12}$$
$$n^{12} \times n^2 = n^{14}$$
$$n^{14} \times n = n^{15}$$

Exponenciação Eficiente - Projeto Euler

Descrição do Problema - Continuação

Entretanto, ainda é possível realizar esse mesmo cálculo em cinco (5) multiplicações:

$$n \times n = n^2$$
$$n^2 \times n = n^3$$
$$n^3 \times n^3 = n^6$$
$$n^6 \times n^6 = n^{12}$$
$$n^{12} \times n^3 = n^{15}$$

Pode-se definir, portanto, $m(k)$ como o número mínimo de multiplicações necessárias para calcular n^k; por exemplo $m(15) = 5$.

Exponenciação Eficiente - Projeto Euler

Descrição do Problema - Continuação

Logo, escreva um programa que encontre o $\sum m(k)$ para $1 \leq k \leq 200$.

O enunciado original do problema da exponenciação eficiente pode ser encontrado no sítio do Projeto Euler - (`https://projecteuler.net/problem=122`).

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Descrição Geral da solução

A solução do problema passa pela análise dos exemplos disponibilizados na descrição do problema, onde pode-se notar que:

1. dentro do limite do exponente utilizado como base de cálculo do número mínimo de multiplicações, **todo o expoente dobra de tamanho a cada execução até quanto for possível**. A exceção fica na transição entre o expoente de número 2 e o exponente de número 3;
2. **todo o exponente ímpar**, utilizado como base de cálculo do número mínimo de multiplicações, **necessita ter representada a transição entre o expoente de número 2 e o expoente de número 3**.

Exponenciação Eficiente - Solução Autor: prof. Jeferson Souza (thejefecomp)

Aspectos peculiares da solução

1. O algoritmo está escrito na linguagem de programação Scilab;
2. A função *m(k)* ganhou um parâmetro adicional, i.e. *m(k,depurar)*, parâmetro este utilizado no controle de depuração da execução do algoritmo, a imprimir cada multiplicação realizada caso o valor da variável depurar seja verdadeiro, i.e. $depurar = \%t$;
3. O algoritmo utiliza operações simples de multiplicação, divisão, e soma. Entretanto, possui variação em uma das suas rotinas que pode ser implementada com a função de arredondamento *ceil()*, a qual encontra o inteiro mais próximo que seja maior ou igual ao valor a ser arredondado;
4. Solução disponível no repositório scilab-codes do perfil do prof. Jeferson Souza (thejefecomp) no Github - https://github.com/thejefecomp.

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Funcionamento do algoritmo para $k > 1$

1. Identifica se o exponente k utilizado como base de cálculo é par ou ímpar;
2. No caso de k ser par, o número de multiplicações recebe o valor inicial de 1 [1. n x n $= n^2$] $\longrightarrow expoenteCorrente = 2$;
3. No caso de k ser ímpar, o número de multiplicações recebe o valor inicial de 2 [2. n^2 x n $= n^3$] $\longrightarrow expoenteCorrente = 3$;
4. Inicia-se o laço definido pela condição $expoenteCorrente < k$, multiplica-se o valor do expoenteCorrente por 2, e armazena-se em expoenteResultante $\longrightarrow expoenteResultante = expoenteCorrente * 2$;
5. Caso $expoenteResultante > k$, inicia-se o processo inverso de divisão;
6. Na primeira execução do processo inverso de divisão, atribui-se o valor do expoenteCorrente para uma variável chamada de expoenteComplementar $\longrightarrow expoenteComplementar = expoenteCorrente$;

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Funcionamento do algoritmo para $k > 1$

7. Divide-se o exponenteComplementar por 2 [$exponenteComplementar = exponenteComplementar/2$], e atribui-se a soma entre expoenteCorrente e expoenteComplementar ao expoenteResultante
$\longrightarrow expoenteResultante = expoenteCorrente + expoenteComplementar$;
8. Repete-se o processo de divisão do expoenteComplementar até que o $expoenteResultante \leq k$;
9. Ao final de cada iteração, o valor do expoenteCorrente recebe o valor do expoenteResultante calculado, e o número de multiplicações é incrementado em 1;
10. Continua-se a execução do laço até que o $expoenteCorrente == k$.

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Funcionamento do algoritmo para $k > 1$

11. Ao final, obtém-se o número mínimo de multiplicações necessárias para obter a exponenciação de n^k.
12. O programa principal define um laço de 1 até 200, a calcular o somatório de m(k) para o referido intervalo. **PS: Veja a beleza do algoritmo, e descubra por si só o resultado ;-)**.

Detalhes importantes

1. Quando $k == 0$ ou $k == 1$, não é necessário realizar multiplicações, i.e., o valor retornado, correspondente ao número mínimo de multiplicações, é zero (0);
2. Quando $k < 0$, $k = k * -1$, a implicar na inversão de sinal do valor de k para efetuar os cálculos.

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

A multiplicação n x n = n^2 está sempre presente

```
expoenteCorrente = 2
numeroMultiplicacoes = 1

if depurar then

   mprintf('\n\n 1. n x n = n^2\n\n')
end
```

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Identificação de expoente ímpar por meio da função modulo()

```
if modulo(k,2) == 1 then

  expoenteCorrente = 3
  numeroMultiplicacoes = 2

  if depurar then

    mprintf('2. n^2 x n = n^3\n\n')
  end
end
```

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Começa a execução do laço

```
expoenteComplementar = 0


/*
  Executado enquanto expoente corrente não for igual ao expoente alvo, i.e. k.
*/
while expoenteCorrente < k

  expoenteResultante = expoenteCorrente * 2

  /*
  Somente entra na rotina de busca de expoente calculado anteriormente
  se o valor da duplicação do expoenteCorrente for maior que o expoente
  alvo (i.e. k).
  */
  if expoenteResultante > k then
```

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Processo inverso de divisão

```
    continuar = %t
    while continuar
        if expoenteComplementar == 2 then
            expoenteComplementar = 1
        elseif expoenteComplementar == 3
            expoenteComplementar = 2
        elseif expoenteComplementar > 3
            expoenteComplementar = expoenteComplementar/2
        end
```

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Processo inverso de divisão - alternativa com a função ceil()

```
    continuar = %t
while continuar

    if expoenteComplementar > 1 then

        expoenteComplementar = ceil(expoenteComplementar/2)

    end
```

Exponenciação Eficiente - Solução
Autor: prof. Jeferson Souza (thejefecomp)

Saída do processo inverso de divisão

```
        expoenteResultante = expoenteCorrente + expoenteComplementar

        if expoenteResultante <= k then

            continuar = %f
        end

    end
```

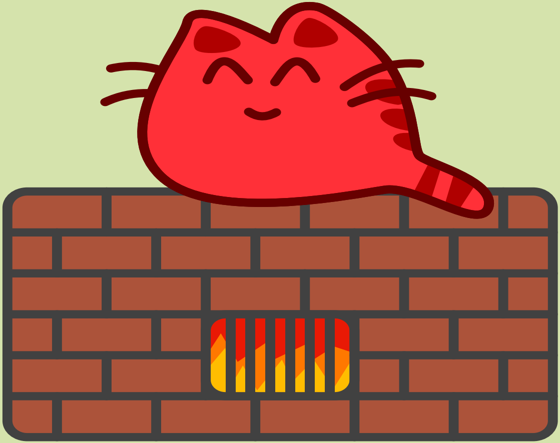
Exponenciação Eficiente - Solução Autor: prof. Jeferson Souza (thejefecomp)

Somatório com depuração

```
somatorio = 0

for i = 1:200

  somatorio = somatorio + m(i,%t)
end


mprintf('somatorio = %d', somatorio)
```

PS: Temos o número 11 como numeração na transparência, a recordar o grande Romário, excepcional jogador da seleção brasileira. Romário era o “rei” da área; mesmo marcado, jogava a movimentar-se com inteligência :-D...

That's it folks!



Thank you for your attention!