PIRÂMIDES DE NÚMEROS PRIMOS PALÍNDROMOS

Prof. André L.B. Cavalcante, DSc

UPIS Faculdades Integradas – Faculdade de Tecnologia
Dept. Sistemas de Informação (andre02592@upis.br)

RESUMO: Este artigo apresenta de forma didática uma explicação sobre a existência dos números primos palíndromos por meio de uma abordagem geométrica e computacional, isto é, apresenta a interdisciplinaridade entre a matemática e as técnicas e linguagens de programação. No artigo são apresentadas pirâmides construídas com números primos palíndromos, utilizando algoritmos desenvolvidos em linguagem C.

1. INTRODUÇÃO

Há mais de 4500 anos, homens construíram, moraram e dormiram em pirâmides de pedras no Antigo Egito (Figura 1).

Segundo o relato de Heródoto, pai da História, a pirâmide de Quéops foi construída num espaço de vinte anos (Mctutor, 2006). Estima-se que cerca de cem mil homens trabalharam nessa obra durante três meses por ano, provavelmente durante a época das cheias do rio Nilo, quando a agricultura ficava paralisada.

Figura 1 – Pirâmide de Quéops

Ainda existem vestígios de rampas que os egípcios utilizavam para transportar as pedras. Uma possível teoria sobre a construção, é que eles usavam uma série de pequenas rampas em torno da pirâmide, para empurrar os blocos iniciais até uma altura de 30 metros. Outra rampa lateral maior, apoiada em apenas uma das faces, servia para transportar o restante das pedras até o topo (Mctutor, 2006).

Quéops, construída cerca de 2500 anos antes de Cristo, é a pirâmide de maiores dimensões. Ocupa uma área de mais de seis hectares, contém quase cinco milhões de toneladas de pedra, mede cerca de 145 metros de altura e possui base quadrada de lados medindo aproximadamente 230 m.

Localizada em Downtown Memphis, a Pirâmide Arena não é apenas uma verdadeira e exclusiva obra de arquitetura estrutural, mas também, uma arena completamente funcional, com 98 m de altura, concluída em 1991 (Figura 2).

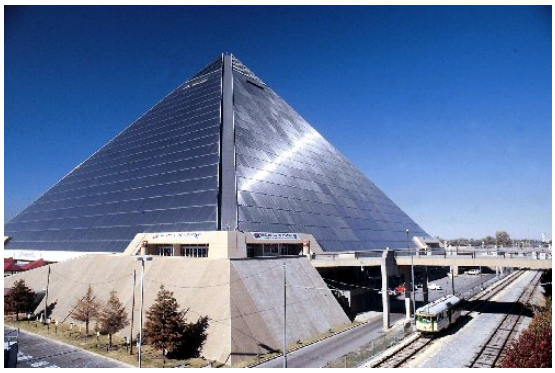

Figura 2 – Pirâmide de Memphis

O nome Memphis é devido ao Antigo Egito e daí a escolha da construção da arena sob a forma de uma pirâmide, só que desta vez construída com metais e vidros ao invés de pedras, sob um rigoroso projeto arquitetônico e de engenharia civil.

As pirâmides exercem um fascínio tão grande nas pessoas que até nas grandes ficções cinematográficas, como em "O Código Da Vinci", são usadas como cenários que guardam consigo numerosos mistérios (Figura 3).

Figura 3 – Pirâmide do Louvre

Este artigo também trata sobre a construção de pirâmides, mas das construídas com números primos palíndromos. Apresenta de forma didática uma explicação sobre a existência dos números primos palíndromos por meio de uma abordagem geométrica e computacional, isto é, apresenta a interdisciplinaridade entre a matemática e as técnicas e linguagens de programação.

2. NÚMEROS PRIMOS PALÍNDROMOS

Os números primos palíndromos são aqueles que possuem simultaneamente as propriedades de números primos e números palíndromos, isto é, possuem apenas dois divisores naturais, o número um e ele próprio e, além disso, o número não se altera quando os algarismos que o compõem são escritos na ordem inversa.

A linguagem C apresenta-se como uma ferramenta viável para implementação de algoritmos capazes de gerar números com estas características (Kernigham & Ritchie, 1988; Cavalcante, 2004).

2.1 Números Primos

Um número inteiro n ($n > 1$) possuindo somente dois divisores positivos n e 1 é chamado primo. Se $n > 1$ não é primo diz-se que é composto.

Muito já se sabe sobre os números primos, entre os principais resultados pode-se citar o Teorema Fundamental da Aritmética e a demonstração de Euclides sobre a infinitude dos números primos. As demonstrações destes resultados já estão consagradas na literatura

(Keng, 1982; Cavalcante, 1997 e Coutinho, 2003), e os teoremas são apresentados a seguir.

Teo 2.1: (Teorema Fundamental da Aritmética) Todo inteiro maior do que 1 pode ser representado de maneira única (a menos da ordem) como um produto de fatores primos.

Teo 2.2: (Euclides) A seqüência dos números primos é infinita.

Um resultado surpreendente é que apesar de infinitos, à medida que crescem surgem entre eles "saltos" que tornam a busca por encontrar tais números cada vez mais difícil e demorada.

Teo 2.3: Para qualquer inteiro positivo *k*, existem *k* inteiros consecutivos todos compostos. Em outras palavras, existem "saltos" arbitrariamente grandes na seqüência dos números primos.

Dem.: Como $(k + 1)!$ é divisível por todos os *k* números entre 2 e $k + 1$, então a seqüência:

$$(k + 1)! + 2,$$
$$(k + 1)! + 3,$$
$$\ldots$$
$$(k + 1)! + k,$$
$$(k + 1)! + (k + 1)$$

é toda composta por *k* números consecutivos compostos, concluíndo a demonstração.

Um importante critério para verificar se um determinado número é primo foi apresentado por Eratóstenes e a demonstração deste resultado também já está consagrada na literatura (Keng, 1982; Cavalcante, 1997 e Coutinho, 2003), e o teoremas é apresentado a seguir.

Teo 2.4: (Crivo de Eratóstenes) Se *n* não é primo, então *n* possui, necessariamente, um fator primo menor do que ou igual a $\sqrt{n}$.

A Figura 4 apresenta uma função escrita em linguagem C capaz de verificar se um determinado número inteiro *p* é primo ou composto.

```
int primo(int p)
{
   int i, pri=0, div=0;
   for (i=1; i<=p; i++)
      if (p%i == 0) div++;
   if (div == 2) pri = 1;
   return(pri);
}
```

Figura 4 – Algoritmo verificador de números primos

Usando o algoritmo da Figura 4 é possível encontrar os números primos listados na Figura 5.

8000053
8000071
8000087
100000037
100000073

Figura 5 – Números primos

2.2 Números Palíndromos

Um número inteiro n $(n \geq 1)$ é dito um número palíndromo quando não se altera se os algarismos que o compõem são escritos na ordem inversa.

Numa primeira abordagem, pode-se tentar trabalhar com números inteiros positivos e bombardeá-los, até descobrir todos os algarismos que o compõem. Neste caso, apenas é necessário utilizar os operadores aritméticos da linguagem C (/ e %), já que o algarismo das unidades de qualquer número inteiro *N* é obtido a partir do resto da divisão do número *N* por 10, isto é:

$$N\%10 \qquad (1)$$

Além disso, o próximo número em análise é dado por:

$$N/10 \qquad (2)$$

Com sucessivas etapas deste algoritmo é possível encontrar todos os algarismos que compõem o número *N*. Uma vez estabelecido todos os algarismos que compõem os números é possível construir um novo número formado com os mesmos algarismos e escrito na ordem inversa.

A Figura 5 apresenta uma função escrita em linguagem C capaz de escrever um número dado, com os algarismo na ordem inversa.

```
int inverso(int k)
{
int i=0, dividendo=k, resto;
while (dividendo > 0)
{
   resto = dividendo%10;
   dividendo = dividendo/10;
   i = i * 10 + resto;
}
return(i);
}
```

Figura 5 – Algoritmo de inversão do número

De posse dos dois números, o informado à função e o gerado por ela, pode-se proceder a comparação dos mesmos e a verificação se são palíndromos ou não. A Figura 6 apresenta uma função escrita em linguagem C, capaz de verificar se um determinado numéro e palíndromo.

```
int palindromo(int p)
{
   int i, pal=0;
   int inverso (int k);
   i = inverso(p);
   if (p == i) pal = 1;
   return(pal);
}
```

Figura 6 – Algoritmo verificador de números palíndromos

Usando o algoritmo da Figura 6 é possível encontrar os números palíndromos listados na Figura 7.

941149
949949
29888892
29988992
29999992

Figura 7 – Números palíndromos

2.3 Números Primos Palíndromos

Como neste artigo interessam apenas os números que são primos e palíndromos, pode-se proceder verificando dentro da lista dos números palíndromos aqueles que possuem apenas dois divisores naturais, isto é, são números primos.

A Figura 8 apresenta um algoritmo escrito em linguagem C capaz de gerar todos os números primos palíndromos dentro de um determinado intervalo. Utilizando o algoritmo apresentado (Figura 8) é possível encontrar os números primos palíndromos apresentados na Figura 9.

2	181	95959	9324239
3	383	96769	9397939
5	787	97579	9400049
7	929	98689	9504059

Figura 9 – Números primos palíndromos

```
#include <stdio.h>
#include <stdlib.h>
int main(int)
{
   int min, max;
   int primo(int p);
   int palindromo(int p);
   printf("\t\tVerificador de Números Primos Palindromos\n\n");
   printf("Digite o intervalo onde deseja verificar: ");
   scanf("%d%d",&min,&max);
   while (min < max)
   {
      if (palindromo(min) ==1)
         if (primo(min)==1)printf("%d e palindromo e primo\n", min);
      min = min + 1;
   }
   system("pause");
   return 0;
}
```

Figura 8 – Algoritmo gerador de números primos palíndromos

3. PIRÂMIDES DE NÚMEROS PRIMOS PALÍNDROMOS

Denomina-se pirâmide de números primos palíndromos, a toda seqüência de números primos palíndromos, tal que a cada novo termo tem-se um novo primo palíndromo com os dígitos centrais iguais ao termo anterior.

Define-se como altura de uma pirâmide de números primos palíndromos a quantidade de termos que esta seqüência possui. No entanto, ao contrário dos antigos egípcios, as pirâmides de números primos palíndromos são construídas do topo para a base. Por exemplo, começando com o número primo 2, pode-se construir a pirâmide de altura 5 (Figura 9).

2
929
39293
7392937
373929373

Figura 9 – Pirâmide de números primos palíndromos de altura 5 começando em 2

Segundo Honaker & Caldwell (2000), começando com apenas um dígito primo e adicionando ao novo termo um dígito de cada lado, encontra-se que as maiores pirâmides possíveis possuem altura cinco. Isto porque existem apenas 4 possíveis dígitos que podem ser adicionados a cada termo: 1, 3, 7, e 9. A Figura 10 apresenta algumas pirâmides que podem ser construídas a partir de um único dígito primo.

		5	5	5	
3	3	151	353	757	7
131	131	31513	33533	37573	373
11311	71317	3315133	1335331	9375739	93739

Figura 10 – Pirâmides de números primos palíndromos construídas a partir de um único dígito primo

Honaker & Caldwell (2000) afirmam também que começar com números primos maiores não ajuda muito, no entanto, existem muitas soluções.

Russo (2000) encontrou uma pirâmide de números primos palíndromos truncada de altura nove (Figura 11).

7159123219517
371591232195173
33715912321951733
7337159123219517337
973371591232195173379
39733715912321951733793
3397337159123219517337933
933973371591232195173379339
39339733715912321951733793393

Figura 11 – Pirâmide de números primos palíndromos truncada de altura nove

Contudo, ao se adicionar dois dígitos a cada lado do termo central, existem 40 pares de dígitos que se pode adicionar ao novo termo da seqüência. Iniciando com o número primo 2, a maior pirâmide que foi construída com esta característica possui altura 26 (Honaker & Caldwell, 2000).

A Figura 12 apresenta uma pirâmide de números primos palíndromos de dois dígitos a cada lado do termo central de altura seis.

2
30203
903020309
3790302030973
98379030203097389
969837903020309738969

Figura 12 - Pirâmide de números primos palíndromos de dois dígitos a cada lado do termo central de altura seis

Ainda existe a possibilidade de construção de pirâmides de números primos palíndromos sendo que a cada termo ocorre uma variação alternada da quantidade de dígitos a cada lado do termo central (Figura 13).

2
727
37273
333727333
93337273339
309333727333903

Figura 13 – Pirâmide de números primos palíndromos de altura seis

4. ALGORITMO GERADOR DE PIRÂMIDES DE NÚMEROS PRIMOS PALÍNDROMOS

Conforme dito anteriormente, para as pirâmides que a cada novo termo são adicionados apenas um número lateral, as únicas possibilidades são:

$$K_i = \begin{bmatrix} 1 \\ 3 \\ 7 \\ 9 \end{bmatrix} \quad (3)$$

onde i varia de 1 até 4.

Seja a seqüência de números primos palíndromos formadores da pirâmide de altura k+1:

$$\{p_1, p_2, ..., p_k, p_{k+1}\} \quad (4)$$

Neste caso, dado p_1 pode-se utilizar a seguinte fórmula de recorrência para gerar o próximo primo palíndromo:

$$p_k = K_i \cdot 10^{2(k-1)} + p_{k-1} \cdot 10 + K_i \quad (5)$$

onde k é maior ou igual a 2 e K_i é escolhido de forma que o número se torne primo e palíndromo.

Fazendo uso destes fatos é possível criar um procedimento, escrito em linguagem C, para

encontrar pirâmides de números primos palíndromos (Figura 14).

```
void piramide(int k2, int alt2)
{
int i, j, num;
int p[4]={1,3,7,9};
printf("\n\n%d\n",k2);
for (j=1;j<=alt2-1;j++)
{
 for (i=0;i<4;i++)
 {
   num=p[i]*pow(10,j*2)+k2*10+p[i];
  if (palindromo(num)==1)
    if (primo(num)==1)
    {
      printf("%d\n", num);
      k2=num;
      break;
    }
  }
 }
}
```

Figura 14 – Algoritmo gerador de pirâmides de números primos palíndromos

Os parâmetros k2 e alt2 do procedimento apresentado na Figura 14 correspondem respectivamente, ao número primo inicial e altura da pirâmide desejada.

O procedimento da Figura 14 apesar de eficiente possui convergência lenta e limitada pelo tamanho das variáveis. É possível suprir um pouco desta limitação substituindo todas as variáveis do tipo `int` pelo tipo `unsigned int`, mas mesmo assim os números gerados são pequenos.

5. ALGORITMO DE FRASES PALÍNDROMAS

Para suprir a limitação do tamanho da variável numérica, a Figura 15 apresenta um algoritmo, escrito em C, verificador de números palíndromos, que utiliza o conceito de string.

Por trabalhar com variáveis literais, este algoritmo também pode ser utilizado para verificar se uma determinada palavra ou frase é ou não um palíndromo. Um exemplo de palavra palíndromo seria REVIVER.

O algoritmo da Figura 15 faz uso de dois vetores unidimensionais de caracteres (string). O primeiro vetor *W* carrega a cadeia de caracteres a ser analisada, enquanto o segundo, vetor inversor *V*, carrega os mesmos caracteres, segundo o critério apresentado a seguir:

$$V[i] = W[(tot - 1) - i] \qquad (6)$$

onde *tot* corresponde ao número total de caracteres do vetor *W*, fornecidos para a análise.

De posse dos dois vetores unidimensionais de caracteres (string) realiza-se a comparação dos mesmos fazendo uso de um contador de caracteres distintos.

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#define max 100
int main(int)
{
    char texto1[max], texto2[max];
    int i, aux=0;
    printf("Digite um número ou um texto: ");
    gets(texto1);
    for(i=0; i<strlen(texto1); i++)
       texto2[i] = texto1[(strlen(texto1)-1)-i];
    printf("\n");
    for(i=0; i<strlen(texto1); i++)
       if (texto1[i] != texto2[i]) aux ++;
    if(aux == 0)
        printf("\nO numero/texto %s e panlindromo", texto1);
    else
        printf("\nO numero/texto %s nao e palindromo");
    printf("\n\n\n");
    system("pause");
    return 0;
}
```

Figura 15 - Algoritmo verificador de números palíndromos que utiliza o conceito de string

Utilizando o algoritmo (Figura 15) é possível verificar que as frases da Figura 16 são exemplos de palíndromos.

"O Pedro me morde, pô"

"I was stressed, desserts saw I'

"O galo nada no lago"

"A mamá Roma lê aviva el amor a papá y a papá Roma le aviva el amor a mamá"

Figura 16 – Frases palíndromas

Este algoritmo apesar de eficiente para verificação de palíndromos grandes trabalha apenas com caracteres e, portanto, não é capaz de realizar cálculos numéricos, sem que a conversão de literal para números seja realizada, como os necessários em algoritmos que verificam se um determinado número é primo ou composto.

6. CONCLUSÕES

Ainda hoje não se sabe ao certo como os antigos egípcios construíram as pirâmides, mas é fato que a interdisciplinaridade entre a matemática e as técnicas e linguagens de

programação surge como uma alternativa viável para o estudo e desenvolvimento de algoritmos que envolvam técnicas construtivas de pirâmides de números primos palíndromos.

Para verificar se um determinado número é primo, o melhor é trabalhar com variáveis numéricas, no entanto, para verificar se este número é palíndromo, o melhor é trabalhar com variáveis literais.

À medida que o número cresce é necessário utilizar critérios mais eficientes para verificar se este número é primo ou composto, devido ao grande esforço computacional observado.

Como pesquisa futura pode-se citar o estabelecimento de critérios mais eficientes, tanto do ponto de vista da matemática, quanto da complexidade do algoritmo, que possibilitem a construção de pirâmides de números primos palíndromos.

REFERÊNCIAS BIBLIOGRÁFICA

Cavalcante, A.L.B. Tópicos em Teoria dos Números. Relatório PIBIC/CNPq. Universidade de Brasília (1997).

Cavalcante, A.L.B. Linguagem e Técnicas de Programação em C. Notas de Aula. Editora UPIS, (2004).

Coutinho, S.C. Números Inteiros e Criptografia RSA. Rio de Janeiro: IMPA. Série de Computação e Matemática (2003).

Honaker, G.L. & Caldwell, C. Palindromic Prime Pyramids (2000).

Keng, H.L. Introduction to Number Theory. Springer-Verlag (1982).

Kernigham, B. & Ritchie, D. C: The Programming Language, (2nd edition), Prentice-Hall (1988).

Mctutor, History of Mathematics Archive The. Disponível em: http://www_history.mcs.st_ andrews.ac.uk/history/index.html (2006).

Russo, F. Private correspondence to Honaker (2000), citado em Honaker & Caldwell (2000).