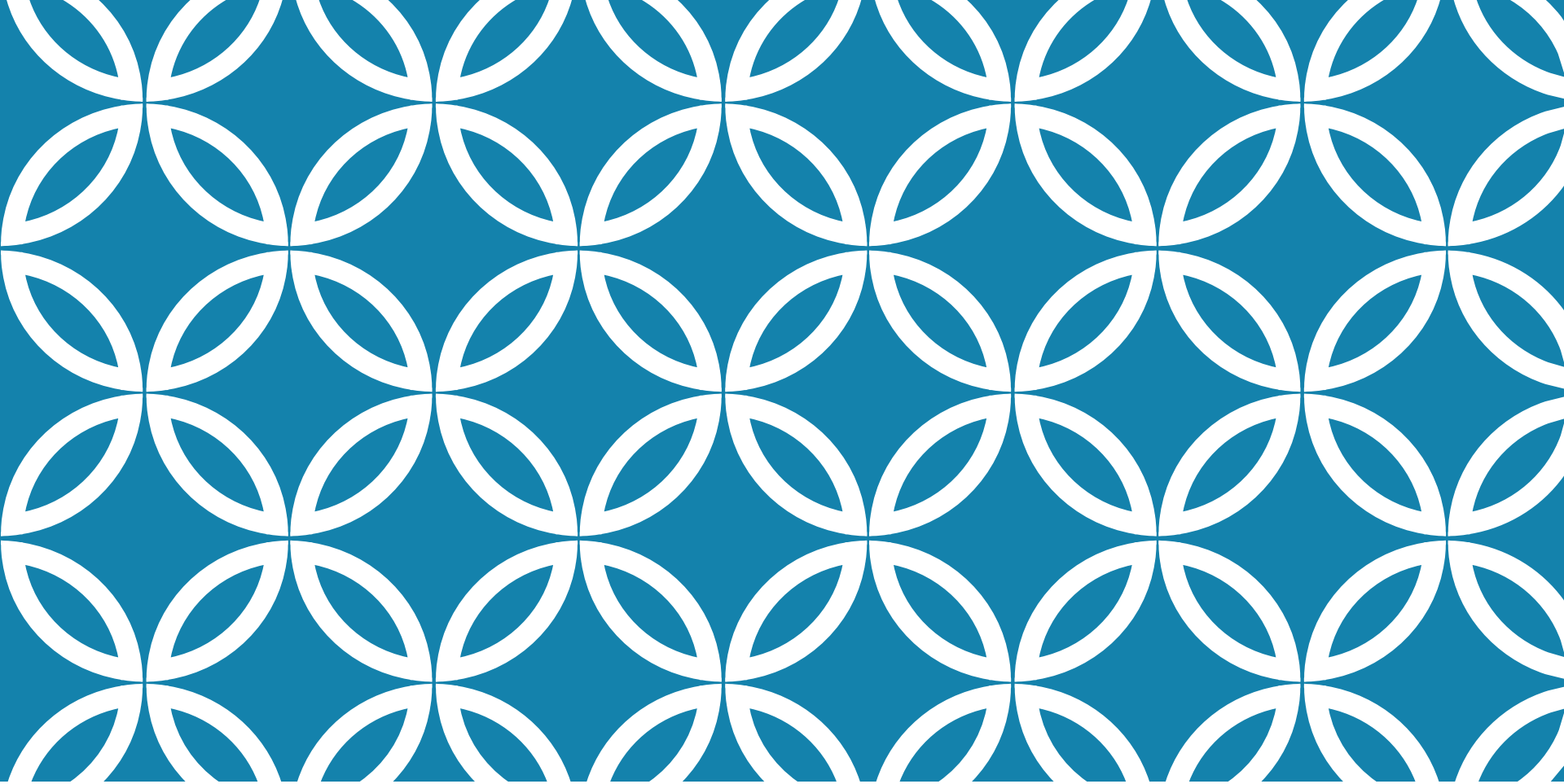


TÉCNICAS ELEMENTARES

Prof. André Vignatti

NÚMEROS PRIMOS

Problema: Dado um número natural *n*, determinar se ele é primo.

```
program ehprimo;
var n, cont, i: integer;
begin
      read (n);
      cont:= 0; (* contador de divisores de n *)
      i:= 2;
      while i <= n-1 do
      begin
            if n mod i = 0 then
                  cont:= cont + 1; (* achou um divisor *)
            i:= i + 1;
      end;
      if cont = 0 then
            writeln (n,' eh primo')
end.
```

PRIMOS: 2ª VERSÃO - VARIÁVEL BOOLEAN

```
program ehprimo_v2;
var n, i: integer;
    eh_primo: boolean;
begin
      read (n);
      eh_primo:= true; (* inicia chutando que n eh primo *)
      i:= 2;
      while (i <= n-1) and eh_primo do
      begin
            if n mod i = 0 then
                  eh_primo:= false; (* se nao for, corrige *)
            i:= i + 1;
      end;
      if eh_primo then
            writeln (n,' eh primo')
end.
```

PRIMOS: 3ª VERSÃO

```
program eh_primo_v3;
var n, i: integer;
    eh_primo: boolean;
begin
    read (n);
    eh_primo:= true; (* inicia chutando que n eh primo *)
    if n mod 2 = 0 then (* n eh par *)
        if n <> 2 then
            eh_primo:= false (* n nao eh 2 *)
        else eh_primo:= true
    else (* n nao eh par, testar todos os impares *)
    begin
        i:= 3;
        while (i <= n-1) and eh_primo do
        begin
            if n mod i = 0 then
                eh_primo:= false; (* achamos um divisor impar *)
            i:= i + 2;
        end;
    end;
    if eh_primo then
        writeln (n,' eh primo')
end.
```

PRIMOS: 4ª VERSÃO

```
program eh_primo_v3;
var n, i: integer;
    eh_primo: boolean;
begin
    read (n);
    eh_primo:= true;                (* inicia chutando que n eh primo *)
    if n mod 2 = 0 then             (* n eh par *)
        if n <> 2 then
            eh_primo:= false        (* n nao eh 2 *)
        else eh_primo:= true
    else begin (* n nao eh par, testar todos os impares *)
        i:= 3;
        while (i <= trunc(sqrt(n))) and eh_primo do
        begin
            if n mod i = 0 then
                eh_primo:= false; (* achamos um divisor impar *)
            i:= i + 2;
        end;
    end;
    if eh_primo then
        writeln (n,' eh primo')
end.
```

COMPARANDO...

Melhorias obtidas nas versões:

x	$\frac{x}{2}$	$\sqrt{x}$
1000000	500000	1000
1000000000	500000000	31622
1000000000000	500000000000	1000000

E a história continua....

(1980) Algoritmo de Miller – Rabin: muito eficiente, permite probabilidade de erro
(2002) Algoritmo de AKS : 1° algoritmo polinomial para PRIMOS

NÚMEROS DE FIBONACCI

A sequência de números de Fibonacci é:

$$1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \ldots$$

E é definido pela seguinte **relação de recorrência:**

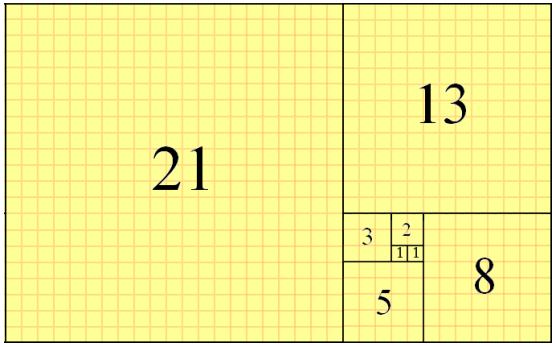
$$F_n = F_{n-1} + F_{n-2},$$
$$F_1 = 1,\ F_2 = 1$$

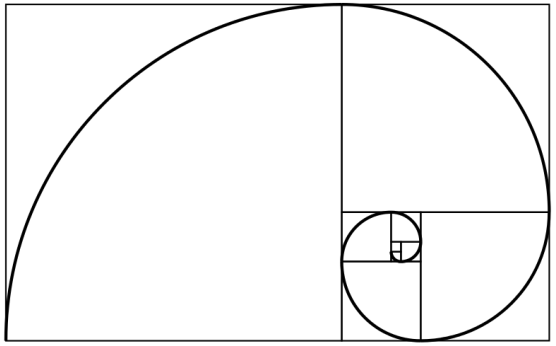
Alunos de BCC vão ouvir falar muito disso...

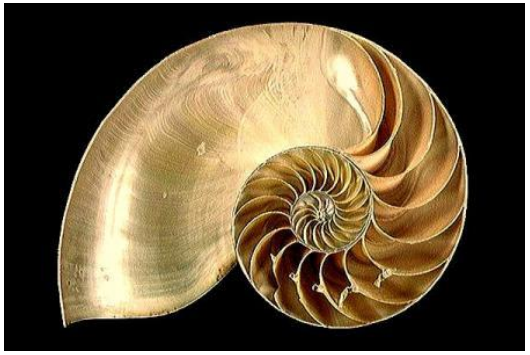
FIBONACCI

```
program Fibonacci;
const max=93;  (* A partir do 94o estoura a capacidade do tipo qword *)
var ultimo, penultimo, soma, cont: integer;
begin
      ultimo:= 1;              (* inicializacao das variaveis principais *)
      penultimo:= 1;
      writeln ('1 ',penultimo);   (* imprime os dois primeiros valores *)
      writeln ('2 ',ultimo);
      cont:= 3 ;                        (* calcula do terceiro em diante *)
      while cont <= max do
      begin
            soma:= penultimo + ultimo;
            writeln (cont, ' ',soma);
            penultimo:= ultimo;         (* a ordem destes dois comandos *)
            ultimo:= soma;              (* eh relevante no codigo       *)
            cont:= cont + 1;
      end;
end.
```

RAZÃO ÁUREA







$$\varphi = \frac{1+\sqrt{5}}{2} = 1.6180339887\ldots$$

Calculando por divisões de números de Fibonacci sucessivos:

$$\frac{1}{1} = 1, \frac{2}{1} = 2, \frac{3}{2} = 1.5, \frac{5}{3} = 1.66, \frac{8}{5} = 1.60, \frac{13}{8} = 1.625, \frac{21}{13} = 1.615, \ldots$$

RAZÃO ÁUREA

```
program numero_aureo;
const PRECISAO=0.00000000000001;
var ultimo, penultimo, soma: integer;
    naureo, naureo_anterior: real;
begin
      ultimo:= 1;        (* inicializacao das variaveis principais *)
      penultimo:= 1;
      naureo_anterior:= -1;  (* para funcionar o primeiro teste  *)
      naureo:= 1;
      writeln (naureo:15:14);
                                  (* calcula do terceiro em diante *)
      while abs(naureo - naureo_anterior) >= PRECISAO do
      begin
            soma:= penultimo + ultimo;
            naureo_anterior:= naureo;
            naureo:= soma/ultimo;
            writeln (naureo:15:14);
            penultimo:= ultimo;
            ultimo:= soma;
      end;
end.
```

NÚMERO E

Problema: Calcular o valor no número $e = 2.718281$ ... pela série abaixo, considerando apenas os vinte primeiros termos:

$$e = \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \frac{1}{5!} + \frac{1}{6!} + \frac{1}{7!} + \ldots$$

```
program neperiano;
var e, novotermo: real;
    fat, i: longint;
const itmax=20;
begin
      e:= 0;                          (* inicializacao das variaveis *)
      fat:= 1;
      i:= 1;                                    (* calculo da serie *)
      while i <= itmax do
      begin
            novotermo:= 1/fat;
            e:= e + novotermo;
            fat:= i*fat;
            i:= i + 1;
      end;
      writeln ('e= ',e);
end.
```