Recorrência

? Exercício 1: Potências de 2

```
using System;
namespace RC1
{class PotenciasDe2
 {static long PotenciaDe2 (int E)
  {if (E==0)
   return 1;
   else
  return 2*PotenciaDe2(E-1);}
static void Main(string[] args)
 {Console.Write("Expoente da potência de 2 (entre 0 e 62) ");
  int Expo=Convert.ToInt16(Console.ReadLine());
  while (Expo<0 || Expo>62)
   {Console.Write("Expoente da potência de 2 (entre 0 e 62) ");
    Expo=Convert.ToInt16(Console.ReadLine());}
  long Pot=PotenciaDe2(Expo);
  Console.WriteLine("2 elevado a {0} = {1}", Expo, Pot);
 }}}
```

? Exercício 2: Contagem decrescente

```
using System;
namespace RC2
 {class ContagemDecrescente
  {static void Contagem(int N)
   {if (N == 0)
    Console.WriteLine(N);
    else
     {Console.WriteLine(N);
      Contagem(N - 1);}}
 static void Main(string[] args)
  {int N = 5;
   Contagem(N);
 }}}
```

? Exercício 3: Adição de N inteiros

```
using System;
namespace RC3
```

```
{class AdicaoDeN
 {static long AdicaoN (int N)
  {if (N==0)
   return 0;
   else
   return N+AdicaoN(N-1);}
 static void Main(string[] args)
  {Console.Write("Maior inteiro a somar ");
   int N=Convert.ToInt16(Console.ReadLine());
   long Total = AdicaoN(N);
   Console.WriteLine("Total da adição dos {0} primeiros "+
                     "inteiros={1}", N, Total);
}}}
```

? Exercício 4: Factorial

```
using System;
namespace RC4
{class FactorialDeN
 {static long Factorial (int X)
  {if (X==0)
   return 1;
   else
   return X*Factorial(X-1);}
 static void Main(string[] args)
  {Console.Write("Digite um numero inteiro positivo ");
   int N=Convert.ToInt16(Console.ReadLine());
   Console.WriteLine("Factorial de {0}!={1}", N, Factorial(N));
}}}
```

? Exercício 5: Números de Fibonacci

```
using System;
namespace RC5
{class FibonnaciDeN
 {static long Fibonacci(int X)
  {if (X <= 2)
    return 1;
    else
     return Fibonacci(X - 1) + Fibonacci(X - 2);}
 static void Main(string[] args)
  { Console.Write("Digite a ordem do número de Fibonnaci que quer
  calcular ");
  int N=Convert.ToInt16(Console.ReadLine());
```

```
  Console.WriteLine("F({0})={1}", N, Fibonacci(N));
 }}}
```

? EXERCÍCIO 6: MÁXIMO DIVISOR COMUM (MDC)

```
using System;
namespace RC6
{class MaxDivComum
 {static int MDC(int X, int Y)
  {if (X == Y)
   return X;
   else
   if (X > Y)
    return MDC(X - Y, Y);
    else
     return MDC(X, Y - X);}
 static void Main(string[] args)
  {Console.Write("Digite um inteiro ");
   int X= Convert.ToInt16(Console.ReadLine());
   Console.Write("Digite outro inteiro ");
   int Y= Convert.ToInt16(Console.ReadLine());
   Console.WriteLine("MDC({0}, {1})={2}", X, Y, MDC(X,Y));
 }}}
```

? EXERCÍCIO 7: INVERSÃO DO NÚMERO

```
using System;
namespace RC7
{class inversaoNumero
 {static void Inversao(int N)
  {if (N > 0)
   {Console.Write(N % 10);
    Inversao(N / 10);}}
 static void Main(string[] args)
  {int N = 654321;
   Inversao(N);
   Console.WriteLine("\n");
}}}
```

? EXERCÍCIO 8: CAPITALIZAÇÃO COMPOSTA

```
using System;
namespace RC8
```


© FCA – Editora de Informática

```
{class CapitalizacaoComposta
 {static double CapAcum(int Inicial, double Taxa,int N)
  {if (N == 0)
   return Inicial;
   else
    return CapAcum(Inicial, Taxa, N-1)* (1+Taxa);}
  public static void Main(string[] args)
   {Console.Write("Capital inicial ");
    int Inicial= Convert.ToInt16(Console.ReadLine());
    Console.Write("Taxa anual de juro em % ");
    double Taxa = Convert.ToDouble(Console.ReadLine())/100;
    Console.Write("Duração da capitalização em anos ");
    int N= Convert.ToInt16(Console.ReadLine());
    Console.WriteLine("{0} euros capitalizados à taxa anual"+
                  " de {1} durante {2} anos={3,5:f2} euros",
    Inicial, Taxa, N, CapAcum(Inicial, Taxa, N));
}}}
```

? EXERCÍCIO 9: TABELAMENTO DE FUNÇÃO

```
using System;
namespace RC9
{class TabelamentoFx
 {static int TabelaF(int X)
  {if (X <= 3)
   return 7;
   else
    return TabelaF(X - 1)+12;}
 static void Main(string[] args)
  {string Tabela="x".PadLeft(3)+"F(x)".PadLeft(6)+"\n";
  for (int X = 0; X <= 10; X++)
    Tabela+=Convert.ToString(X).PadLeft(3)+
                   Convert.ToString(TabelaF(X)).PadLeft(5)+"\n";
  Console.WriteLine(Tabela);
}}}
```

? EXERCÍCIO 10: SOMATÓRIO DE TERMOS DE SÉRIE

```
using System;
namespace RC10
{class SomatorioSerie
 {static double SomaSerie(int N, double Numerador, double
                          Denominador, int Ordem)
  {if (Ordem > N)
```

```
    {return 0;}
    else
    {if (Ordem % 2 == 0)
     Numerador +=2;
     else
     Denominador +=2;
     return Numerador / Denominador + SomaSerie(N, Numerador,
            Denominador, Ordem+1);
   }}
   static void Main(string[] args)
   {int N = 21;
   double Numerador = 1;
   double Denominador = 0;
   int Ordem = 1;
   Console.WriteLine("Somatorio dos {0} termos={1,5:F3}",
   N,SomaSerie(N, Numerador, Denominador,Ordem));
}}}
```

? EXERCÍCIO 11: IMPRESSÃO DE UM QUADRADO

```
using System;
namespace RC11
{class Program
 {static String Quadrado(int N, int I, int J,char C1)
  {if (I < 0)
    { return "\n"; }
      else
      {if (J <= N-1)
        return C1 + Quadrado(N, I, J + 1, C1);
        else
        return "\n" + Quadrado(N, I - 1, 0, C1);}}
 static void Main(string[] args)
 {char C1 = '*';
  int N = 6;
  Console.WriteLine(Quadrado(N, N - 1, 0, C1));
}}}
```

? EXERCÍCIO 12: NÚMERO PRIMO

```
using System;
namespace RC12
{class NumeroPrimo
 {static Boolean Primo(int N, int Div)
  {if (N==1)
   return false;
```


© FCA – Editora de Informática

```
    else
     if (N == 2)
      return true;
      else
          if (Div >(int)Math.Sqrt (N))
           return true;
           else if (N % Div == 0)
                 return false;
           else return Primo(N, Div + 1);}
 static void Main(string[] args)
  {Console.Write("Digite um inteiro ");
  int N=Convert.ToInt16(Console.ReadLine());
  int Div = 2;
  Console.WriteLine("{0} {1}", N, Primo(N, Div)? "Primo"
                   : "Não primo");}
}}
```

? EXERCÍCIO 13: IMPRESSÃO DE UM VECTOR

```
using System;
namespace Vectores
{class Program
 {static void Impressao(int[] A, int I)
  {Console.Write("{0, 5}",A[I]);
  if (I ==A.Length-1)
   Console.WriteLine();
   else
    {Console.Write(",");
     Impressao(A, I+1);}}
 public static void Main(string[] args)
  {int[] A = { 35, 45, 23, 78, 90, 65, 78, 90, 76, 34, 25, 100 };
  Impressao(A,0);
  }}}
```

? EXERCÍCIO 14: IMPRESSÃO INVERSA DE UM VECTOR

```
using System;
namespace RC14
{class ImpressaoInversaDeVector
 {static void ImpressaoInversa(int[] A, int I)
  {if (I < 0)
   Console.WriteLine();
   else
   {Console.Write("{0, 5}",A[I]);
   ImpressaoInversa(A, I-1);}}
```

```
public static void Main(string[] args)
 { int[] A = { 35, 45, 23, 78, 90, 65, 78, 90, 76, 34, 25, 100 };
  int N = A.Length;
  ImpressaoInversa(A,N-1);
}}}
```

? EXERCÍCIO 15: CONTAGEM DOS ELEMENTOS DE UM VECTOR

```
using System;
namespace RC15
{class ContagemInfaX
 {static int Contagem(int[] A, int X, int I)
  {int Conta = 0;
   if (I >=0)
    {if (A[I] < X)
     Conta++;
     Conta+= Contagem(A, X, I - 1);}
   return Conta;}
 static void Main(string[] args)
 {int N=12;
  int[] A = { 35, 45, 23, 78, 90, 65, 78, 90, 76, 34, 25, 100 };
  Console.Write("Digite o limite superior da contagem ");
  int X=Convert.ToInt16(Console.ReadLine());
  Console.WriteLine("Número de elementos inferiores a {0}={1}",
  X, Contagem(A,X,N-1));
}}}
```

? EXERCÍCIO 16: SOMATÓRIO DOS ELEMENTOS DE UM VECTOR

```
using System;
namespace Vectores
{class Program
 {static int Somatorio(int[] A, int I)
  {int Total = 0;
   if (I == 0)
    return Total+= A[I];
    else
       return Total+=A[I]+Somatorio(A, I-1);}
 public static void Main(string[] args)
 {int[] A = { 35, 45, 23, 78, 90, 65, 78, 90, 76, 34, 25, 100 };
  Console.WriteLine("Somatório dos elementos do vector={0}",
  Somatorio(A, A.Length-1));
}}}
```


© FCA – Editora de Informática

? EXERCÍCIO 17: MAIOR NÚMERO DE UM VECTOR

```
using System;
namespace RC17
{class MaiorNumero
 {static int Maior (int[] A, int Inferior, int Superior) {
  int Meio, Esquerda, Direita;
  if (Inferior == Superior)
  return A[Inferior];
   else
   {Meio = (Inferior + Superior) / 2;
    Esquerda = Maior(A, Inferior, Meio);
    Direita = Maior(A, Meio + 1, Superior);
    if (Esquerda > Direita)
     return Esquerda;
    else
     return Direita;}}
 static void Main(string[] args)
  {int[] A={1203, 67, 89, 124, 12, 156};
  Console.WriteLine("Maior elemento={0}",
                    Maior(A, 0,A.Length-1));
}}}
```

? EXERCÍCIO 18: PESQUISA LINEAR

```
using System;
namespace RC18
{class PesquisaLinear
 {static String ExisteLinear(int X, int[] A, int I)
  {if (I == A.Length)
   return X + " Não existe no vector";
   else
    if (A[I] == X)
     return X + " existe na posição " + I;
     else
      return ExisteLinear(X, A, I + 1);}
 static void Main(string[] args)
  {int[] A = { 3, 8, 2, 19, 56, 20, 90 };
   Console.Write("Número a procurar ");
   int X=Convert.ToInt16(Console.ReadLine());
   Console.WriteLine(ExisteLinear(X, A, 0));}
}}
```

? EXERCÍCIO 19: PESQUISA BINÁRIA

```
using System;
namespace RC19
{class PesquisaBinaria
 {static String ExisteBinaria(int X, int[] A, int Inferior, int
                              Superior)
  {int Meio;
   if (Inferior > Superior)
    return X + " não existe";
    else
     {Meio = (Inferior + Superior) / 2;
     if (A[Meio] == X)
      return X + " existe na posição " + Meio;
      else
       if (A[Meio] > X)
       return ExisteBinaria(X, A, Inferior, Meio - 1);
       else
        return ExisteBinaria(X, A, Meio + 1, Superior);}}
 static void Main(string[] args)
 {int[] A = { 2, 3, 8, 19, 20, 56, 90 };
  Console.Write("Número a procurar ");
  int X=Convert.ToInt16(Console.ReadLine());
  int Inferior=0;
  int Superior=A.Length-1;
  Console.WriteLine(ExisteBinaria(X, A,Inferior, Superior));}
}}
```

? EXERCÍCIO 20: FUNÇÃO DE ACKERMANN

```
using System;
namespace RC20
{class Ackermann
 {static int Acker(int m, int n)
  {if (m==0)
   return 1;
   else
    if (m==1 && n==0)
     return 2;
     else
      if (m>1 && n==0)
       return m+2;
       else
        return Acker(Acker(m-1,n),n-1);}
 static void Main(string[] args)
```


© FCA – Editora de Informática

```
  {int m=1, n=0;
   Console.WriteLine("Ackermann({0},{1})={2}", m, n, Acker(m,n));
   m=3; n=0;
   Console.WriteLine("Ackermann({0},{1})={2}", m, n, Acker(m,
   n));
   m = 4; n = 3;
   Console.WriteLine("Ackermann({0},{1})={2}", m, n, Acker(m,
   n));
}}}
```

? EXERCÍCIO 21: FRASE DO FIM PARA O INÍCIO

```
using System;
namespace RC21
{class Inversaofrase
 {static string InversaoFrase(string X, int N)
  {if (N==0)
   return X;
   else
    return X.Substring(N,1)+InversaoFrase(X.Substring(0,N),N-1);}
 static void Main(string[] args)
 {string X = "adreuqse a arap atierid ad";
  int N=X.Length-1;
  Console.WriteLine(InversaoFrase(X, N));
}}}
```

? EXERCÍCIO 22: PALÍNDROMO

```
using System;
namespace RC22
{class Palindromo
 {static string Tirarespacos(string Frase, int Indice)
  {if (Indice==0)
   return Frase.Substring(Indice, 1);
   else
    if (Frase.Substring(Indice, 1) == " ")
     return Tirarespacos(Frase, Indice - 1);
     else
      return Frase.Substring(Indice,1)+
             Tirarespacos(Frase,Indice - 1);}
 static Boolean Palind(string Palavra, int Indice)
  {string Ce, Cd;
   if (Indice > (Palavra.Length - 1)/2)
    return true;
    else
```

```
      {Ce = Palavra.Substring(Indice, 1);
       Cd = Palavra.Substring(Palavra.Length - 1 - Indice, 1);
       if (Ce != Cd)
        return false;
        else
         return Palind(Palavra, Indice + 1); }}
 static void Main(string[] args)
 {string Frase = "A base do teto desaba";
  string Palavra=Tirarespacos(Frase.ToLower(), Frase.Length-1);
  Console.Write("'"+Frase+"'");
  Console.WriteLine(Palind(Palavra, 0)?" é um palíndromo.":" não
  é um palíndromo.");
}}}
```

? EXERCÍCIO 23: TORRES DE HANOI

```
using System;
namespace RC23
 {class Torreshanoi
  {static void Hanoi(int Altura, char De, char Para, char Usando)
   {if (Altura==1)
    Console.WriteLine("Mudar o disco de {0} para {1}", De, Para);
    else
     {Hanoi(Altura-1, De, Usando, Para);
      Console.WriteLine("Mudar o disco de {0} para {1}", De,
                        Para);
      Hanoi(Altura-1, Usando, Para, De);}}
  static void Main(string[] args)
  {int Altura = 4;
  Hanoi(Altura, 'A', 'C', 'B');
}}}
```

? EXERCÍCIO 24: ORDENAÇÃO RÁPIDA

```
using System;
namespace QuickSort
{class Program
 {static void OrdenaRapida(int[] V, int Inicio, int Fim)
 {int Temp ;
  int Esq=Inicio;
  int Direita=Fim;
  int Inicial=V[(Inicio+Fim)/2];
  while (Esq<Direita)
   {while (V[Esq]<Inicial)
     Esq++;
```


© FCA – Editora de Informática

```
    while (Inicial<V[Direita])
      Direita--;
    if (Esq<=Direita)
     {Temp=V[Esq];
     V[Esq]=V[Direita];
     V[Direita]=Temp;
     Esq++;
     Direita--;}}
  if (Inicio<Direita)
   OrdenaRapida(V, Inicio,Direita);
  if (Esq<Fim)
   OrdenaRapida(V, Esq, Fim);}
 static void Main(string[] args)
 {int[] V = { 10, 90, 67, 45, 78, 32, 17, 89, 100, 108, 21};
  OrdenaRapida(V, 0, V.Length-1);
  foreach (int I in V)
   Console.Write("{0,5:D2}", I);
  Console.WriteLine();
}}}
```

? EXERCÍCIO 25: LABIRINTO

```
using System;
namespace RC25
{class Labirinto
{static void Saidas(char[,] L, int Linha, int Coluna)
 {const char Saida = 'S';
  const char Sim = ' ';
  L[Linha, Coluna] = Saida;
  Boolean Fim = Bordo(Linha, Coluna, L.GetLength (0));
  if (Fim == true)
   ImprimirLabirinto(L);
   else
    {if (L[Linha - 1, Coluna] ==Sim)
     Saidas(L, Linha - 1, Coluna);
    if (L[Linha, Coluna + 1] == Sim)
     Saidas(L, Linha, Coluna + 1);
    if (L[Linha + 1, Coluna] == Sim)
     Saidas(L, Linha + 1, Coluna);
    if (L[Linha, Coluna - 1] ==Sim )
     Saidas(L, Linha, Coluna - 1);}}
 static Boolean Bordo(int Linha, int Coluna, int N)
  {return (Linha == N-1) || (Coluna == N-1) || (Linha==0) ||
            (Coluna==0);}
 static void ImprimirLabirinto(char[,] L)
  {String Ts = "";
   for (int I = 0; I <= L.GetLength(0) - 1; I++)
```

```
    {for (int J = 0; J <= L.GetLength(0) - 1; J++)
       Ts += L[I, J];
     Ts += "\n";}
   Console.WriteLine("Saída do labirinto\n\n" + Ts);}
 static void Main(string[] args)
  {const char Nao='*';
   const char Sim = ' ';
   int N = 10;
   char[,] L = new char[N, N];
   for (int I = 0; I <=N-1; I++)
    for (int J = 0; J <= N-1; J++)
     L[I, J] = Nao;
   L[0, 3] = Sim; L[1, 3] = Sim; L[1, 4] = Sim; L[1, 5] = Sim;
   L[1, 6] = Sim;
   L[2, 3] = Sim; L[2, 6] = Sim; L[2, 7] = Sim; L[3, 3] = Sim;
   L[3, 4] = Sim; L[3, 5] = Sim;
   L[3, 7] = Sim; L[3, 8] = Sim; L[4,4 ] = 'P'; L[4,5 ] = Sim;
   L[4,8 ] = Sim;
   L[5,5 ] = Sim; L[5,8 ] = Sim; L[6,5 ] = Sim; L[7,5 ] = Sim;
   L[7,6 ] = Sim;
   L[7,7 ] = Sim; L[8,4 ] = Sim; L[8,5 ] = Sim; L[8,7 ] = Sim;
   L[9, 4] = Sim; L[9, 7] = Sim;
   Console.Clear();
   Saidas(L, 4, 4);
}}}
```


© FCA – Editora de Informática