data / /
S T Q Q S S D

lista 05

1- A sequência de dupla de Fibonacci é uma sequência matemática definida da seguinte maneira: o primeiro elemento é 1, o segundo elemento 1 e apartir daí os próximos elementos são construídos como a soma dos dois anteriores multiplicado por 2.

1, 1, 4, 10, 28, 76, ...

Proponha uma recorrência para caracterizar a sequência dupla de Fibonacci.

$F(n) = (F(n-1) + F(n-2)) * 2 \quad n > 2$

$F(1) = 1$

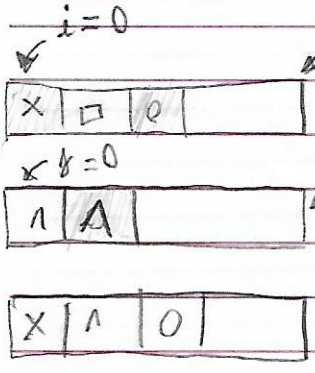
$F(2) = 1$

2- Crie um algoritmo que calcula o número na posição n, dada de entrada, da sequencia dupla de Fibonacci.

```
int F (int n)
se (n>2)
   retorna (F(n-1) + F(n-2) * 2)
senão
   retorna 1;
```

3- Escreva um algoritmo recursivo que recebe um vetor a de tamanho n e retorna um vetor ordenado vec de tamanho n com os elementos de a.

→ Ordenar as partes do vetor < dividir o vetor de alguma maneira
→ Combinar partes < ordenar
< combinar.

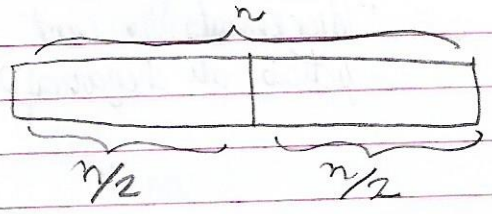


data / /

S T Q Q S S D

Função Merge

```
int [n+m] MERGE (int a[n], int b[m], int n, int m)
var[int] i := 0;
var[int] j := 0;
var[int] k := 0;
Var[int] c[n+m];
    Para  K := 0  até  n+m
        se (a[i] < b[j]) então
            c[K] := a[i];
        fim  i := i + 1;
        senão
            c[K] := b[j];
            j := j + 1;
Retorna c;
```

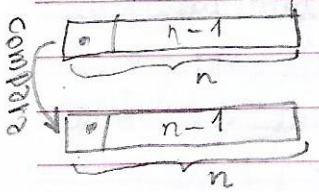


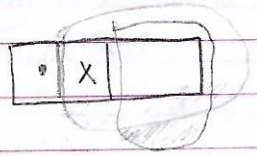

```
int [n] ORDENA (int a[n], int n)
var[int] b[n/2]
Var[int] c[n/2]
Var[int] i
    se (n>1) então
        Para i := 0 até n/2
            b[i] := a[i];
            c[j] := a[i + n/2];
var[int] b.ord := ORDENA (b, n/2)
Var[int] c.ord := ORDENA (c, n/2)
Retorna MERGE (b.ord, c.ord, n/2, n/2);
fim
senão
    Retorna a;
```

data / /

S T Q Q S S D

4. Escreva um algoritmo recursivo que (bool) testa se dois vetores a e b de mesmo tamanho, n, são iguais, ou seja, possuem os mesmos elementos nas mesmas posições.





bool COMP (int a[n], int b[n], int n)

se n > 0 (Compara 1ª posição) & (compara o resto do vetor.)

Retorna (a[n-1] = b[n-1]) & COMP(a,b,n-1);

fim

senão

Retorna a[0] = b[0])

fim

Sempre que a função comp for chamada ela "decrementa" o contador até chegar a 0.

data / /

S T Q Q S S D

6. Determine uma recorrência para calcular o número de sequências de 0's e 1's de tamanho n sem 0's consecutivos.

$$B(n) = \begin{cases} B(n-1) + B(n-2) & \text{se } n > 2 \\ B(2) = 3, \\ B(1) = 2 \end{cases}$$

7. Escreva um algoritmo que calcula o número de sequencia de 0's e 1's de tamanho n sem 0's consecutivos.

```
bool B(int n) {
  se n>2 então
    retorna B(n-1) + B(n-2);
  fim
  senão
    se n=2 então
      retorna 3;
    fim
    senão
      retorna 2
    fim
  fim
```

8. Escreva um algoritmo recursivo que recebe um vetor a de tamanho n e determina se seu vetor é um palíndromo. Um vetor a de tamanho n é dito ser palíndromo se a[i] = a[n-i] para todo $i \leq \frac{n}{2}$.

```
Entrada: int a[n], int n;
Var [int] i := 0;
  Para i=0 até i ≤ n/2
    Se a[i] = a[n-1]
      retorna Palíndromo
    Senão
      retorna Não é Palíndromo
    fim s
  fim
```

data / /
S T Q Q S S D

Lista 04.

1. Escreva uma função que recebe um vetor de inteiros a[n] de tamanho n no intervalo de 0 até 99 e retorna seu vetor característico.

```
Bool [n] CACT (int a[n], int n) {
    Var [bool] vec [100];
    Var [int] i;
        Para i:=0 até n-1
            vec [a [i]]:= V;
        fim
    Retorna vec;
```

2. Crie uma função bool dois_primo (int a, int b). Esta função recebe dois números inteiros e retorna verdadeiro se ambos os números são números primos.

```
bool dois_primo ( int a, int b ) {
    Var [bool] pa:= primo (a);
    Var [bool] pb:= primo (b);
    Retorna pa & pb;
```

3. Crie uma função que recebe duas variáveis inteiras e inverte seus conteúdos.

```
bool SWAP ( Var int x, Var int y) {
    Var [int] aux;
        aux:= x;
        x:= y;
        y:= aux;
    Retorna V;
}
```

data / /
S T Q Q S S D

4. Escreva uma função que recebe um vetor de variáveis inteiras a[n] de tamanho n e o ordena em ordem não decrescente.

```
bool ORDENA (Var int a[n], int n) {

  Var[int] i:=0;      // marca a posição do vetor que ainda será ordenada
  Var[int] j:=0;      // percorre o vetor
  Var[int] min:=0;    // valor mínimo encontrado
  Var[int] pos:=0;    // posição do menor valor

  Para i:=0 até n-1
      min := a[i];
      pos:= i;

      para j:= i+1 até n-1
        se (a[j] < min)
            min := a[j];
            pos := j;

      fim
      SWAP(a[i], a[pos]); // função recebe 2 variáveis
      fim retorna V;
```

5. Escreva uma função que realiza a união entre dois conjuntos de inteiros, no intervalo de 0 até 99, representados por dois vetores de variáveis inteiras a e b, dados como entrada.

```
Entrada: int a[i], int b[j], int n, int m.
Saída: a ∪ b
Var[int] i:=0;
Var[int] j:=0;
Var[int] k:=0;
    para i:=0 até 99
    vec_A[i]:= a[i].

    fim
    para j:=0 até 99
    vec_B[j] = a[j].

    para k:=1 até 199
    vec_C[K]:= VEC_A[K] | VE_B[K];

    fim
    retorna Vet_C.
```

data / /
S T Q Q S S D

7. (int [n+m] merge() (int a[n], int b[m], int n, int m)
Escreva uma função que recebe dois vetores ordenados a e b, cria um terceiro a e b, cria um terceiro vetor ordenado vec que é construído com os elementos de a e de b.

```
int [n+m] MERGE (int a [n], int b [m], int n, int m ).
Var [int] i := 0;
Var [int] j := 0;
var [int] K := 0;
Var [int] C [n+m];
    Para K := 0 até n+m
        se ( a[i] < b[j] ) entao
            c[K] := a[i];
            i := i+1;
    fim
    Senão
        c [K] := b[j]
            j := j+1;
    Retorna c;

int [n] ORDENA (int a[n], int n)
Var [int] b [n/2]
Var [int] c [n/2]
Var [int] i := 0
    Se (n>1) entao
        Para i := 0 até n/2
        b [i] := a [i];
        c [i] := a [i+n/2]
Var [int] b.ord := ORDENA (b, n/2)
Var [int] c.ord := ORDENA (c, n/2)
Retorna MERGE (b.ord, c.ord, n/2, n/2).
fim
Senão
    retorna a;
```