1. Crie a função recursiva `soma_digitos(n)`, que devolve a soma dos dígitos que compõem o número natural `n`. Por exemplo, a chamada `soma_digitos(7859)` deve devolver a resposta `29` (= `7+8+5+9`).

2. Usando a função `soma_digitos()`, criada no exercício anterior, crie a função recursiva `super_digito(n)`, que devolve o *super dígito* de um número natural `n`, computado como exemplificado a seguir:
   `super_digito(9875)` = `super_digito(29)` = `super_digito(11)` = `super_digito(2)` = `2`.

3. Considere as funções a seguir:
   ```
   int suc(int n) { return n+1; }
   int pred(int n) { return n-1; }
   ```
   Usando essas funções (e *nenhum* operador aritmético), crie funções recursivas para determinar:
   (a) A *soma* de dois números naturais.
   (b) A *subtração* de dois números naturais (o resultado pode ser negativo).
   (c) O *dobro* de um número natural.
   (d) A *metade* inteira de um número natural.
   (e) O *mínimo* entre dois números naturais.
   (f) O *máximo* entre dois números naturais.
   (g) Se um número natural é *par*.
   (h) Se um número natural é *ímpar*.

4. Usando as funções do exercício anterior (e *nenhum* operador aritmético), crie funções recursivas para calcular:
   (a) O *produto* de dois números naturais.
   (b) O *quociente* da divisão inteira de dois números naturais.
   (c) O *resto* da divisão inteira de dois números naturais.
   (d) A *potência* de um número natural elevado a outro número natural.
   (e) O *quadrado* de um número natural n, considerando que $n^2 = 1 + 3 + 5 + ... + (2n-1)$.

5. Crie uma função recursiva para determinar quantos dígitos são necessários para representar um número natural n em binário.

6. Crie uma função recursiva para determinar o número mínimo de movimentos necessários para resolver o problema das *Torres de Hanói* com $n>0$ discos.

7. Crie a função recursiva `inverte(c,p,u)`, que inverte uma cadeia de caracteres `c`, cujo primeiro caractere está na posição `p` e cujo último caractere está na posição `u`. Por exemplo, se a cadeia `c` for `"amor"`, após a chamada `inverte(c,0,3)`, a cadeia `c` deverá ser `"roma"`.

8. Crie a função recursiva `palindroma(c,p,u)`, que determina se uma cadeia de caracteres `c`, cujo primeiro caractere está na posição `p` e cujo último caractere está na posição `u`, é *palíndroma*. Por exemplo, a chamada `palindroma("subi no onibus",0,13)` deve devolver `1` como resposta.

9. Crie a função recursiva `palavras(c)`, que determina quantas palavras existem na cadeia de caracteres `c` (considere que `c` contém apenas letras e espaços; que entre duas palavras consecutivas na cadeia `c` existe pelo menos um espaço; e que pode haver um ou mais espaços tanto no começo quanto no final da cadeia `c`). Por exemplo, a chamada `palavras(" Quantas  palavras existem  nesta cadeia ")` deve devolver a resposta `5`.

10. Crie uma função recursiva para verificar se dois vetores de inteiros $v[n]$ e $w[n]$ são iguais, para $n>0$.

11. Crie uma função recursiva para verificar se um vetor de inteiros $v[n]$ está em ordem crescente.

12. Crie uma função recursiva para informar o item máximo de um vetor de inteiros $v[n]$.

13. Crie uma função recursiva para calcular a média de um vetor de reais $v[n]$.

Importante: *Resolva também os exercícios do Capítulo 6 do livro usado em aula.*