

# Roteiro de Laboratório - PICAT

Grau de dificuldade: médio

1. Construa a árvore geneológica de sua família, tendo as relações (predicados): filho e pai;
2. A partir do programa anterior, construa as relações: irmão, avô e ancestral. Um ancestral generaliza os conceitos de pai, avô, bisavô, etc;
3. Quais dos predicados abaixo casam, e se a unificação ocorrer, quais são os resultados. Escreva os seus significados, tomando por base os conceitos de termos ou átomos, “matching” (“casamento”), e unificação:

- `Picat> 2 > 3.`
- `Picat> >(2, 3).`
- `Picat> 2 == 2.`
- `Picat> a(1, 2) = a(X, X).`
- `Picat> a(X, 3) = a(4, Y).`
- `Picat> a(a(3, X)) = a(Y).` /\* deixar para depois \*/
- `Picat> 1+2 = 3.`
- `Picat> X = 1+2.`
- `Picat> a(X, Y) = a(1, X).`
- `Picat> a(X, 2) = a(1, X).`
- `Picat> 1+2 = 3`
- `Picat> X + 2 = 3 * Y.`
- `Picat> X+Y = 1+2.`
- `Picat> 1+Y = X + 3.`
- `Picat> lectures(X, ai) = lectures(alison, Y).`
- `Picat> X+Y = 1+5, Z = X.`
- `Picat> X+Y = 1+5, X=Y.`

4. Seja o programa abaixo:

```
b(1).  
b(z).
```

```

d(3).
d(11).
c(3).
c(z).
a(X) => b(X), c(X), false.
a(X) => c(X), d(X).

```

Encontre os valores para “ $a(Y)$ ” e explique como foi o esquema de busca. Onde e porquê ocorreu “*backtracking*”?

5. Repasse TODOS exercícios apresentados em LPO, na sala, e reescreva-os em PICAT. **Não avance aos próximos exercícios sem fazer todos exercícios acima.**

6. Resolva por recursão os seguintes problemas:

- Cálculo do fatorial (Obs:  $\text{fat}(0)$  é 1,  $\text{fat}(N)$  é  $\text{fat}(N-1) * N$ );
- Soma intervalar a partir de um valor “ $n1$ ” até “ $n2$ ” tal que “ $n2 > n1$ ”.
- Seja F a versão recursiva da função de Fibonacci. A função de Fibonacci é definida assim:  $F(0) = 0$ ,  $F(1) = 1$  e  $F(n) = F(n-1) + F(n-2)$  para  $n > 1$ . O cálculo do valor da expressão  $F(3)$  provocará a seguinte sequência de invocações da função:

```

F(3)
  F(2)
    F(1)
      F(0)
    F(1)

```

Qual a sequência de invocações da função durante o cálculo de  $F(5)$ ? Implemente-a em PICAT.

- Implemente em PICAT um programa que imprima um retângulo de (X,Y) de asteriscos, onde X é o número de linhas e Y é o número de colunas.
- Implemente em PICAT um programa que imprima um triângulo de X de asteriscos, na primeira linha, X-1 na linha seguinte, e assim sucessivamente, até 1 asterisco na última linha.
- Reescreva o problema acima, mas com o triângulo invertido.

- Implemente em PICAT um programa que imprima um triângulo de X de asteriscos, na primeira linha, contudo, este deverá ter um formato de um triângulo equilátero.
- Reescreva o problema acima, mas com o triângulo equilátero invertido.
- Write a program in Picat to print first 50 natural numbers using recursion.
- Write a program in Picat to calculate the sum of numbers from 1 to n using recursion.
- Write a program in Picat to Print Fibonacci Series using recursion.
- Write a program in Picat to print even or odd numbers in given range using recursion. Example:

Test Data :

Input the range to print starting from 1 : 10

Expected Output :

All even numbers from 1 to 10 are : 2 4 6 8 10

All odd numbers from 1 to 10 are : 1 3 5 7 9

- The digital root of a non-negative integer  $n$  is computed as follows. Begin by summing the digits of  $n$ . The digits of the resulting number are then summed, and this process is continued until a single-digit number is obtained. For example, the digital root of 2019 is 3 because  $2+0+1+9=12$  and  $1+2=3$ . Write a recursive function `digitalRoot(n)` which returns the digital root of  $n$ . Assume that a working definition of `digitalSum` will be provided for your program.
- The hailstone sequence starting at a positive integer  $n$  is generated by following two simple rules. If  $n$  is even, the next number in the sequence is  $n/2$ . If  $n$  is odd, the next number in the sequence is  $3 * n + 1$ . Repeating this process, we generate the hailstone sequence. Write a recursive function `hailstone(n)` which prints the hailstone sequence beginning at  $n$ . Stop when the sequence reaches the number 1 (since otherwise, we would loop forever 1, 4, 2, 1, 4, 2, ...) For example, when  $n=5$ , your program should output the following sequence:

5

16  
8  
4  
2  
1