Prolog

```
/* Grafo */

              Maria -- Divino
                |         \
  Vicente -- Geralda     Ana, Lilian
          |
Gabriel / Adan / Theo


* Fatos */

pai(Divino, Geralda).
pai(Divino, Ana).
pai(Divino, Lilian).
mae(Maria, Geralda).
mae(Geralda, Gabriel).
mae(Geralda, Adan).
mae(Geralda, Theo).
marido(Divino, Maria).
marido(Vicente, Geralda).

/* Regras */

filhos(X, Y) :- pai(Y, X).
                mae(Y, X).

irmaos(X, Y) :- pais(A, X),
                pais(A, Y),
                X \== Y.
pais(X, Y) :- pai(A, Y), pai(A, X),
              mae(A, Y), mae(A, X),
              X \== Y.

tios(X, Y) :- irmaos(X, A),
              pais(A, Y).

avos(X, Y) :- pais(X, A),
              pais(A, Y).

netos(X, Y) :- filhos(Y, A),
               filhos(A, X).
```

Java

```
// Implementar um sistema de registros de animais,
// de maneira modular o bastante que permita o reuso
// eficiente do código de acordo com o animal graças
// à programação orientada a objetos; neste exemplo,
// o registro de um gato.

public class RegistroAnimal()
{
    String dono;
    Sreing nome;
    String especie;
    String raca;
    int idade;
    float tamanho;
    Boolean castrado;

    // Construtor
    public RegistroAnimal(String dono, String nome, String especie

                          String raca, int idade, int tamanho)
    {
        this.dono = dono;
        this.nome = nome;
        this.especie = especie;
        this.raca = raca;
        this.idade = idade;
        this.tamanho = tamanho;
```

```
    }

    // Declarações de getters e setters

}

// A classe RegistroGato herda atributos e métodos
// da classe mãe RegistroAnimal, economizando na
// implementação dos mesmos.

public class RegistroGato extends registroAnimal()
{
    private int tamanhoDasUnhas;
    private float intensidadeMiado;

    public RegistroGato()
    {
        // Código do construtor
    }

    // Getters e setters
}
```

Common Lisp

```
;; Implementar um jogo de adivinhação de números por meio
;; de uma busca binária

(defparameter *max* 100)
(defparameter *min* 1)

(defun adivinha ()
    ;; A função ash executa um shift binário de acordo
    ;; com o parâmetro passado, dobrando ou dividindo
    ;; o valor pela metade.
    (ash (+ *max* *min*) -1))

(defun maior ()
    (setf *min* (1- (adivinha)))
    (adivinha))

(defun menor ()
    (setf *max* (1- (adivinha)))
    (adivinha))

(defun recomecar ()
    (defparameter *max* 100)
    (defparameter *min* 1)
    (adivinha))
```