Modelo

|  |  |  |
| --- | --- | --- |
| 1 | Dibujo de un perro  Descripción generada automáticamente con confianza media | procedure Linea(a,b){  Poner(b)  repeat(4) {  Mover(a)  Poner(b)  }  }  procedure HacerH(a){  Mover(Oeste)  Mover(Norte)  Poner(a)  Mover(Norte)  Mover(Norte)  Poner(a)  }  program {  Linea(Norte,Negro)  Mover(Este)  Mover(Este)  Linea(Sur,Negro)  HacerH(Negro)  Mover(Oeste)  repeat(3){  Mover(Sur)  }  } |
| 2 | Definí el procedimiento DibujarMesa para que dibuje una mesa como la que te mostramos del color que reciba como argumento. El procedimiento debe dibujar desde el extremo Sur Oeste pero no te preocupes por dónde termina el cabezal. | procedure Linea(a,b,c){  repeat(a){  Poner(b)  Mover(c)  }  }  procedure Volver(a,b){  repeat(a){  Mover(b)  }  }  procedure DibujarMesa(a){  Linea(2,a,Norte)  Linea(3,a,Este)  Linea(2,a,Sur)  Poner(a)  Volver(3,Oeste)  } |
| 3 | Definí la función concatenacionEsIgual.    Ahora vamos a definir una función concatenacionEsIgual que recibe tres strings y nos dice si la concatenación de los dos primeros es igual al tercero | function concatenacionEsIgual(a,b,c){  return a+b===c;  } |
| 4 | Podríamos tener una función que dado un nombre y un horario retorne el saludo correcto. Si es antes de las 12 debería ser "Buenos días", si es después "Buenas tardes". Por ejemplo:  saludar("Luis", 11)  "Buenos días Luis"  saludar("Carolina", 12)  "Buenas tardes Carolina"  saludar("Rocío", 13)  "Buenas tardes Rocío" | function saludar(a,b){  if(b>=12){  return "Buenas tardes "+a;  }  else{  return "Buenos días "+a;  }  } |
| 5 | Queremos obtener la suma de los elementos mayores a 7 de una lista. Por ejemplo:  sumaDeLosMayoresASiete([2, 5, 9, 8, 11])  28 //Porque es la suma de 9, 8 y 11. | function sumaDeLosMayoresASiete(numeros){  let suma=0;  for(let i of numeros){  if(i>7){  suma+=i;  }  }  return suma;  } |
| 6 | Queremos definir una función que retorne un resumen de manera simple.  resumenCancion(elGenioDeLaNada)  "El genio de la nada de la banda Eruca Sativa tiene una duración de 240 segundos"  resumenCancion(lotusFlower)  "Lotus flower de la banda Radiohead tiene una duración de 300 segundos" | function resumenCancion (a){  return a.nombre+" de la banda "+a.banda+" tiene una duración de "+a.duracion\*60+" segundos"  } |
| 7 | * calentarla una temperatura (si la calentamos 10 grados, su temperatura incrementa en 10); * ver si tiene la temperatura correcta, es decir, si está a exactamente 80 grados.   Definí en Ruby, el objeto AguaParaMate que tenga un atributo @temperatura que inicialmente está en 0 con su getter. AguaParaMate entiende los mensajes calentar\_agua! (que recibe la cantidad de grados a sumar por parámetro) y temperatura\_exacta?. | module AguaParaMate  @temperatura=0    def self.temperatura  @temperatura  end    def self.calentar\_agua!(grados)  @temperatura+=grados  end    def self.temperatura\_exacta?  @temperatura==80  end  end |
| 8 | Teniendo en cuenta que los libros saben responder al mensaje nombre...  Definí en Ruby el método nombres\_de\_libros que responda el nombre de los libros de la Biblioteca. | module Biblioteca  @libros = [Fundacion, Contacto, LaInsoportableLevedadDelSer, Socorro, ComoAguaParaChocolate]  def self.nombres\_de\_libros  @libros.map {|libro| libro.nombre}  end  end |
| 9 | Cada vez que un Camion carga un mueble, el mueble sufre una modificación:   * cada Colchon pierde 4 resortes; * cada Sillon sube su nivel\_de\_polvo en 20; * a los Electrodomesticos no les pasa nada.   Definí los metodos cargar\_muebles! en la clase Camion. También definí el método ser\_cargado! en los muebles junto con su respectivo getter. | class Camion  def initialize(muebles)  @muebles = muebles  end  def cargar\_muebles!  @muebles.each {|mueble| mueble.ser\_cargado!}  end  end  class Colchon  def initialize(resortes)  @cantidad\_de\_resortes = resortes  end  def cantidad\_de\_resortes  @cantidad\_de\_resortes  end  def ser\_cargado!  @cantidad\_de\_resortes-=4  end  end  class Sillon  def initialize(polvo)  @nivel\_de\_polvo = polvo  end  def nivel\_de\_polvo  @nivel\_de\_polvo  end  def ser\_cargado!  @nivel\_de\_polvo+=20  end  end  class Electrodomestico  def ser\_cargado!  end  end |