**Práctica integradora de Registros**

Comidas Gobianas

Hay gente que se ha obsesionado tanto con Gobstones que comenzó a basar su incluso su alimentación en este lenguaje. Estas personas se denominan Gobianos, y solo se alimentan a base de una serie de comidas que ellos consideran que son aptos para el desarrollo de un programador exitoso.

No vamos a juzgar sus decisiones alimenticias, pero si vamos a querer modelar las comidas de estas personas. Para ello, contaremos con los siguientes tipos:

|  |
| --- |
| **type ComidaGobiana is record {**  /\* PROPÓSITO: Modela una comida que come un gobiano.  INVARIANTES DE REPRESENTACIÓN:  \* calorías es un número mayor a cero. \*/  **field nombre** // tipo String  **field origen** // tipo Continente  **field calorias** // tipo Número  **}** |

|  |
| --- |
| **type Continente is variant {**  /\* PROPÓSITO: Modela un continente de donde puede  proceder una comida gobiana. \*/  **case África {}**  **case América {}**  **case Asia {}**  **case Europa {}**  **case Oceanía {}**  **}** |

1. Como primer ejercicio queremos modelar dos de las comidas Gobianas más típicas, las Bolitas de Fraile, y las Bolitas de Queso. Las primeras son una comida originaria de América, y tiene 500 calorías. Las segundas, son de Europa, y tienen tan solo 150 calorias. Se pide implemente entonces:
   1. **bolitasDeFraile()**, una función que describe la comida gobiana Bolitas de Fraile.
   2. **bolitasDeQueso()**, una función que describe la comida gobiana Bolitas de Queso.
2. Deseamos determinar si, dadas dos comidas gobianas, la primera tiene más calorías que la segunda. Para ello, realizar la función **tieneMásCalorias\_Que\_(comida1, comida2)** que describe Verdadero cuando la primer comida gobiana dada tiene más calorías que la segunda, y Falso en cualquier otro caso.
3. Además, queremos poder elegir de forma automática la comida más saludable, dadas dos comidas gobianas. Para ello, queremos crear la función **comidaMásSaludableEntre\_Y\_(comida1, comida2)** que describe la comida gobiana más saludable de las dos comidas dadas. La comida más saludable es aquella que tiene menos calorías.
4. Algunos fanáticos de las comidas gobianas solo comen comidas si provienen de determinadas regiones. Por ello, deseamos poder crear la función **proviene\_De\_(comida, unContinente)** que describe Verdadero si una comida proviene del continente dado, y Falso en caso contrario.
5. Hay gente que tiene la idea de que, si se le unta manteca a una comida, esta se vuelve mucho mejor. Por este motivo, se quiere hacer la función **enmantecar(comida)** que describe una nueva comida, igual que la dada, pero donde las calorías aumentan en 200. Por ejemplo, enmantecar una bolita de fraile, produce una bolita de fraile de 700 calorias.
6. En el mundo Gobiano hay también Chefs, que se han vuelto muy renombrados por sus recetas. Queremos poder modelar a un chef y las tres recetas más populares del mismo.

|  |
| --- |
| **type ChefGobiano is record {**  PROPÓSITO: Modela un chef de comida gobiana.  **field nombre** // tipo String  **field receta1** // tipo ComidaGobiana  **field receta2** // tipo ComidaGobiana  **field receta3** // tipo ComidaGobiana  **}** |

Nos interesará saber diversas cosas de un Chef, pero por el momento solo queremos modelar al Chef “Simón Bolitas”, el gran cocinero que ha dado origen a todo este movimiento. Cree la función **simonBolitas()** que describe al chef cuyas comidas características incluyen las bolitas de fraile y las bolitas de queso, y una tercer receta a su elección.

1. Queremos saber si un chef tiene o no comidas de un determinado continente. Para ello, realice la función **tienePlatosDe\_(unContinente, unChef)** que describa verdadero si el chef tiene como plato popular uno que venga del continente dado.
2. Se desea saber si chef es o no exótico. Un chef es exótico cuando algunos de sus platos proviene de África, Asia, u Oceanía. Realice la función **esChefExótico(unChef)** que describa Verdadero si el chef dado es exótico y Falso en caso contrario.
3. Queremos saber también cuántas comidas pulenta sabe preparar un chef. Para ello, elabore la función **cantidadDeComidasPulentaDe(unChef)** que describa la cantidad total de comidas pulenta (de más de 500 calorias) que el chef sabe preparar. Note que el resultado puede estar entre cero y tres.
4. Para ir cerrando esta sección queremos saber si un chef es demasiado light. Un chef es demasiado light si ninguna de las comidas que sabe preparar es una comida pulenta. Elabore la función **elChefEsMuyLight(unChef)** que describa Verdadero si ninguna de sus comidas es pulenta, y Falso en caso de que prepare al menos una comida pulenta.