Segundo Parcial

- **NO OLVIDAR PONER EL NOMBRE Y APELLIDO EN CADA HOJA**
- **NUMERAR LAS HOJAS CON LA FORMA <número de hoja>/<cantidad total de hojas>**
- Leer el examen completo antes de empezar a resolver los ejercicios, ya que ayuda a comprender mejor el dominio. Se puede consultar cualquier material (en papel) que haya sido escrito antes de comenzar el examen y usar sin definir todas las funciones y procedimientos vistos durante la cursada.
- Pensar bien la estrategia a seguir, consultar lo que no se entienda y recordar que el parcial es una instancia más de aprendizaje donde algún docente va a dar una devolución personalizada de todo lo que se escriba, es necesario para aprovechar esta instancia como algo más que solamente un momento para sacar una nota.

Juegos Olimpicos

El comité olímpico internacional requiere antes del comienzo de París 2024 agregar funcionalidad a su sistema informático que al momento cuenta con los siguientes tipos

```
type JuegoOlímpico is record {
  /*
  PROPÓSITO:  Modelar un juego olimpico
  INV. REP. :
    * **disciplinas** no tiene dos disciplinas con el mismo nombre.
    * **delegaciones** no tiene dos delegaciones del mismo pais.
    * No hay dos atletas con el mismo número de atleta entre todos los atletas
    de todas las delegaciones.
  */
  field sede          // String
  field año           // Numero
  field delegaciones  // [Delegacion]
  field disciplina    // [Disciplina]
}
```

```
type Disciplina is record {
  /*
  PROPÓSITO: Modelar una disciplina deportiva.
  INV. REP.:
  * **nombre** no es un string vacío.
  */  */
  field nombre          // String
  field categoria       // Categoria
  field yaFinalizo      // Booleano
}
```

```
type Delegacion is record {
  /*
  PROPOSITO: Modelar la delegación de deportistas de un pais.
  INV. REP.:
    *  **pais** no es un string vacío
    *  **atletas** no está vacía
    *  **atletas** no tiene dos atletas con el mismo número
  */
  field pais          //String
  field atletas       // [Atleta]
}
```

```
type Medalla is variant {
  /*
  PROPOSITO: Modelar los distintos tipos de medella
  entregados.
  */
  case Oro     {}
  case Plata   {}
  case Bronce  {}
}
```

```
type Categoria is variant {
  /*
  PROPOSITO: Modelar las distintas categorías de disciplina.
  */
  case Atletismo   {}
  case Gimnasia    {}
  case Equipo      {}
  case Acuatico    {}
  case Otros       {}
}
```

```
type Atleta is record {
  /*
  PROPOSITO: Modelar atleta olímpico.
  INV. REP. :
    * **númeroDeAtleta** es mayor a cero
    * **nombre** no es un string vacío
    * **disciplinas** no está vacía
  */
  field nombre          // String
  field categoria       // Categoria
  field yaFinalizo      // Booleano
}
```

```
function ganadorDeMedallaDe_En_(medalla, disciplina) {
  /*
  PROPÓSITO:  Describe el número del atleta que ganó la
medalla de **medalla** en la disciplina **disciplina**
    PARÁMETROS:   * medalla:    Medalla
                  * disciplina: Disciplina
    TIPO: Atleta
    PRECONDICIONES:
      * La competencia de esa disciplina ya ha finalizado
  */
}
```

Usando este modelo como base considere las siguientes situaciones

Punto 1)

Se pide implementar paísLiderDelMedalleroEn_(juegoOlímpico) que describa el nombre del país de la delegación que lleva ganado más Oros en el juego olímpico, en caso de empate definen las medallas de plata y en caso de nuevo empate las de bronce. Si aún así hubiera empate, decribe cualquiera de las delegaciones.

Punto 2)

Implemetar la función juegoOlímpico_LuegoDeAgregarDisciplina_AAtletaNro_(juegoOlímpico, disciplina, nroDeAtleta) que dado un juegoOlímpico, una disciplina y el número de un atleta que seguro participa de los juegos dados, describa un Juego Olímpico donde el atleta con ese número compite también en la disciplina dada. Si el atleta ya practicaba esa disciplina se describe el juego olímpico sin cambios.

Para resolver este problema puede hacer uso de la primitiva reemplazarEnAtletas_AtletaConNúmero_Por_ , que dada una lista de atletas, un número de atleta y un atleta, reemplaza en la lista dada el atleta cuyo número coincide con el número dado por el nuevo atleta dado.

Punto 3)

Dada la siguiente función cuyo contrato es correcto

```
function participa_EnLaDisciplina_(delegación, disciplina) {
  /*
  PROPOSITO: Indica si la delegación **delegación** posee al menos un atleta que participa en la
        disciplina **disciplina**.
  PRECONDICIONES: Ninguna.
  PARÁMETROS:
    * delegación:   Delegación
    *disciplina:    Disciplina
  TIPO: Booleano
  OBSERVACIONES: Recorrido de búsqueda sobre una delegación dada por parámetro.
  */
  seEncuentraDisciplina:= False
  foreach atleta in atletas(delegación) {
    seEncuentraDisciplina:=  compite_En_(atleta, disciplina)
  }
  return (
    seEncuentraDisciplina
  )
}

function compite_En_(atleta, disciplina) {
  /*
    PROPÓSITO:  Indica si el atleta **atleta** participa en la disciplina **disciplina**
    PARÁMETROS:
      * atleta:      Atleta
      * disciplina:   Disciplina
    TIPO: Booleano
    PRECONDICIONES:
      * Ninguna
  */
  return (
    contiene_A_(disciplinas(atleta), disciplina)
  )
}
```

Se pide que determine si:
(a) La solución es correcta (soluciona el problema planteado en el propósito o no), y
(b) Si es adecuada (sigue los buenos criterios de programación vistos en la materia o no)
JUSTIFIQUE sus respuestas (la justificación no puede tener más de 5 renglones).