PARTE1:

Podemos abstraernos de la forma de las fichas, para trabajar las mismas de forma genérica, recurriendo a representarlas como números. Así podemos usar como argumento la “forma de una ficha”, pasando por ej. 2, cuando queremos mencionar al “círculo”, o 5 cuando queremos mencionar “estrella”. El número elegido se corresponde con la cantidad de bolitas azules que usamos para representar cada forma de ficha. Veremos más adelante en la materia que esta no es la mejor solución, pero servirá por ahora.

1)

Con esto en cuenta, escribir el procedimiento `Intercambiar_Por_Al_` que dados dos números que corresponden a formas de fichas, donde el primero es la forma de la ficha en la celda actual y el segundo la forma de la ficha en la celda lindante en la dirección dada, y una dirección, intercambia la ficha actual por la ficha en la dirección dada. Puede asumir que la primera de las formas de fichas se corresponde con la ficha en la celda actual y que efectivamente hay una ficha lindante en dicha dirección que se corresponde a la segunda forma de ficha dada.

2)

Escribir el procedimiento `Reponer_FichasAl_` que dados un número y una dirección repone tantas celdas consecutivas hacia la dirección dada como el número pedido, comenzando desde la celda actual. Ej. sí el número es 3 y la dirección es Este, repone la actual, y las siguientes 2 al Este. Reponer fichas implica colocar fichas nuevas de forma “aleatoria” donde explotaron fichas y quedaron espacios vacíos.

Para solucionar este problema puede hacer uso del siguiente procedimiento primitivo.

```
procedure PonerFichaAleatoria() {
/*
  PROPÓSITO: Pone una ficha aleatoria en la celda actual.
  PARÁMETROS: No tiene.
  PRECONDICIONES:
    * La celda actual debe estar vacía.
*/
```

PARTE 2:

Se pide desarrolle la función `direcciónDeLaSucesión` que describe la dirección hacia la cual se encuentra una sucesión de fichas que comienza en la celda actual. Se asume que en la celda actual hay una ficha y que es el inicio de una sucesión de 3 o más fichas en alguna de las direcciones (pero no en más de una).

Para solucionar el problema cuenta con la siguiente función que se presenta como primitiva:

```
function haySucesiónAl_

/*

  PROPÓSITO: Indica sí, comenzando en la celda actual, y hacia
la dirección dada, hay una sucesión de 3 o más fichas de la
misma forma.

  PARÁMETROS:

    * Dirección - La dirección hacia la cual evaluar sí hay una
sucesión.

  RESULTADO: Booleano

  PRECONDICIONES:

    * Debe haber una ficha en la celda actual.

*/
```

Escribir el procedimiento ExplotarSucesiones que quita del tablero todas las sucesiones que hubieran en el mismo. Tener en cuenta que no necesariamente hay una ficha en cada celda del tablero (puede haber obstáculos o incluso no haber nada). Se asume que no hay nunca dos o más sucesiones que comienzan en la misma celda.

Se cuenta con las siguientes funciones primitivas para resolver el problema:

```
function haySucesión

/*

  PROPÓSITO: Indica sí, comenzando en la celda actual, y hacia
cualquiera de las direcciones, hay una sucesión.   PARÁMETROS:
No tiene   RESULTADO: Booleano.   PRECONDICIONES:     * Debe
haber una ficha en la celda actual.

*/

function hayFicha

/*

  PROPÓSITO: Indica sí hay una ficha en la celda actual.
PARÁMETROS: No tiene   RESULTADO: Booleano.   PRECONDICIONES:
* Ninguna

*/

function fichaActual

/*
```

```
  PROPÓSITO: Describe la forma de la ficha en la celda actual.
PARÁMETROS: No tiene   RESULTADO: Número.   PRECONDICIONES:
* Debe haber una ficha en la celda actual.

*/

procedure SacarFicha

/*

  PROPÓSITO: Saca la ficha en la celda actual, dejando la celda
vacía.   PARÁMETROS: No tiene   PRECONDICIONES:     * Debe haber
una ficha en la celda actual.

*/
```