PROCEDIMENTS EXAMEN

- FACTORIAL D'UN NÚMERO

```
/* Calcula el factorial el número que recibe por parámetro.
 *
 * @param n (Valor: enter) Número de entrada (n >= 0).
 * @return (enter) Devuelve el factorial del número de entrada
(n).
*/

función factorial (/* Parámetro de entrada*/ n: entero) devuelve
entero és
```

- PERMUTACIONS AMB REPETICIONS

```
/* Calcula las permutaciones con repeticiones.
*
* @param m (Valor: enter) Número de elementos.
* @param a (Valor: enter) Repetición del primer elemento (a >
0).
* @param b (Valor: enter) Repetición del segundo elemento (b >
0).
* @param c (Valor: enter) Repetición del tercer elemento (c >
0).
* @return (enter) Devuelve las permutaciones con repetición.
*/

función permutaciones_repeticion ( /* Parámetro de entrada*/
m: enter, a: enter, b: enter, c: enter) devuelve entero és
```

- VARIACIONS SENSE REPETICIÓ

```
/* Calcula las variaciones de m elementos cogidos de n en n (sin
repeticiones).
*
* @param m (Valor: enter) Número de elementos.
* @param n (Valor: enter) Cuantos elementos cogemos.
* @return (enter) Devuelve las variaciones sin repetición.
*/

funció variaciones (m: enter, n: enter) retorna enter és
```

- LLEGIR NOMBRE ENTRE MAX I MIN

```
/* Lee un valor entero acotado entre un máximo y un mínimo.
*
* @param min (Valor: enter) Valor mínimo.
* @param max (Valor: enter) Valor máximo.
* @return (enter) Devuelve un valor entero acotado entre dos
valores [min y max]
*/

función lee_entero_acotado ( /* Parámetro de entrada*/   min:
enter, max:enter) devuelve entero és
```

- MOSTRAR MENÚ PER PANTALLA I TORNAR L'OPCIÓ TRIADA

```
/* Muestra un menú y devuelve la opción elegida.
 * La opción siempre será válida.
 *
 * @return (enter) Devuelve la opción elegida.
*/

función menu () devuelve entero és
```

- MOSTRAR MENÚ PER PANTALLA I EXECUTAR L'OPCIÓ TRIADA

```
/* Muestra un menú y ejecuta la opción elegida.
*
*/
acció lab09() és
```

- CALCULAR INCREMENTS

```
/* Determina los incrementos de fila y columna para recorrer una
matriz
*  siguiendo la orientación indicada.
*
* @param orientacion (Valor: enter) Identificación de la
orientación.
* @param incf (Ref: enter) Incremento aplicable a las filas.
* @param incc (Ref: enter) Incremento aplicable a las columnas.
*/

acció tb_calcula_incrementos(/*Parámetros de entrada*/
orientacion: enter,/*Parámetros de salida*/ var incf: enter, var
incc: enter) és
```

- LLEGIR CARÀCTER

```
/* Lee un carácter de teclado. El carácter ha de pertenecer a
una cadena.
 *
 * @param cadena (Ref: taula[] de caràcter) Cadena que contiene
los caracteres válidos.
 * @param texto (Ref: taula[] de caràcter) Texto informativo que
se muestra al usuario.
 * @return (caràcter) Devuelve un carácter que se encuentra
entre los de la cadena.
 */

función lee_caracter_cadena(/*Parámetro d'entrada*/ cadena:
taula[] de caràcter, texto: taula[] de carácter) devuelve
caràcter és
```

- INICIALITZAR UNA MATRIU

```
/* Inicializa una matriz con un carácter.
*
* @param matriz (Ref: taula[][] de caràcter) Matriz a
inicializar.
* @param nfilas (Valor: enter) Número de filas de la matriz
(nfilas > 0).
* @param mcols (Valor: enter) Número de columans de la matriz
(mcols > 0).
* @param caracter (Valor: caràcter) Caracter con el que se
inicializa.
*/

acció tb_inicializa_matriz(/* Parámetros de salida */   var
matriz: taula[][] de caràcter, /* Parámetros de entrada */
     nfilas: enter, mcols: enter, caracter: caràcter) és
```

- BUSCA A UNA MATRIU

```
/* Busca en una matriz un determinado carácter e
*  indica la fila y la columna donde se encuentra.
*
* @param matriz (Ref: taula[][] de caràcter) Matriz donde
buscamos.
* @param nfilas (Valor: enter) Número de filas de la matriz
(nfilas > 0).
* @param mcols (Valor: enter) Número de columans de la matriz
(mcols > 0).
* @param f (Ref: enter) Fila donde se ha encontrado el caracter
buscado.
* @param c (Ref: enter) Columna donde se ha encontrado el
caracter buscado.
* @param caracter (Valor: caràcter) Caracter que se está
buscando.
```

```
* @return (booleà) Retorna cierto si lo ha encontrado y falso en
caso contrario.
*/

funció tb_busca_matriz( /* Parámetros de entrada */
matriz: taula[][] de caràcter, nfilas: enter, mcols: enter,
/* Parámetros de salida */ var f: enter, var c: enter,
/* Parámetros de entrada */     caracter: caràcter) retorna
booleà és;
```

- ENTER ALEATORI ACOTAT

```
/* Determina un número aleatorio acotado [min..max].
 * Los valores mínimo y máximo pueden no estar ordenados.
 *
 * @param min (Valor: entero) Valor mínimo que puede tomar el
número aleatorio.
 * @param max (Valor: entero) Valor máximo que puede tomar el
aleatorio.
 * @return (entero) Devuelve un aleatorio perteneciente a
[min..max].
 */

funció entero_aleatorio_acotado ( /* Parámetros de entrada */
min: enter, max: enter) retorna enter és
```

- CARÀCTER ALEATORI ACOTAT

```
/* Determina un carácter aleatorio acotado [inf..sup]. Los
valores inferiores y
 * superiores se determinan según el código ascii
correspondiente.
 * Pueden no estar ordenados.
 *
 * @param inf (Valor: carácter) Carácter inferior que limita los
carácteres válidos.
 * @param sup (Valor: carácter) Carácter superior que limita los
carácteres válidos.
 * @return (carácter) Devuelve un aleatorio perteneciente a
[inf..sup].
 */

funció caracter_aleatorio_acotado ( /* Parámetros de entrada */
inf: caràcter, sup: caràcter) retorna caràcter és
```

- SEGÜENT JUGADOR

```
/* Determina el jugador que tendrá el turno siguiente.
 *
 * @param jugador (Valor: entero) Identificador del jugador que
tiene el turno actualmente.
 * @param num_jugadores (Valor: entero) Número total de
jugadores del juego. Si el valor es inferior o igual a uno, no
cambia el turno.
 * @return (entero) Devuelve el identificador del siguiente
jugador.
 */

funció siguiente_jugador ( /* Parámetros de entrada */
      jugador: enter, num_jugadores: enter) retorna enter és
```

- CANVIA DE COLOR

```
/* Cambia el color de la ficha: si la ficha era de color era
negro,
 * nos devuelve el color blanco (FICHA_BLANCA) y si era blanco,
nos devuelve negro (FICHA_NEGRA).
 * Si el color no es correcto, nos devolverá la FICHA_VACIA.
 *
 * @param color_ficha (Valor: caràcter) Color actual de la
ficha.
 * @return (caràcter) Devuelve el color contrario al recibido
como parámetro o FICHA_VACIA en caso de error.
*/

/* @enun fichas_t.
* Determina los símbolos para representar las fichas en el
tablero.
*/

constants
      FICHA_BLANCA <- 'O';
      FICHA_NEGRA <- '@';
      FICHA_VACIA <- ' ';
fconstants

funció ot_cambia_color ( /* Parámetros de entrada */
color_ficha: caràcter) retorna caràcter és
```

- DINS DELS LÍMITS

```
/* Determinan si una fila y columna dada están dentro de los
límites de una matriz [nfilas][mcols]
*
* @param f (Valor: entero) Número de la fila.
* @param c (Valor: entero) Número de la columna.
* @param nfilas (Valor: entero) Número máximo de filas de la
matriz.
```

```
* @param mcols (Valor: entero) Número máximo de columnas de la
matriz.
* @return (booleà) Devuelve cierto si la fila pertenece a
[0..nfilas) y
*               columna pertenece a [0..mcols) y falso en caso
contrario.
*/

funció tb_dentro_limites ( /* Parámetros de entrada */ f: enter,
c: enter, nfilas: enter, mcols: enter) retorna booleà és
```

- BUSCA A LA MATRIU

```
/* @def MCOLS_MAX
Numero máximo de columnas de las matrices.
*/
constants MCOLS_MAX <- 10;  fconst

/* Busca un carácter en una matriz de nfilas x mcols, a partir
de una determinada posicion (fila,col).
 * La busqueda se realiza en una determinada orientacion.
 * No se evalua la casilla inicial (fila, col)
 * El procedimiento devuelve si existe o no. En caso de que
existe,
 * la casilla donde se encuentra el elemento se devuelve a
partir de los mismos parametros de entrada fila y col.
 *
 * @param matriz (Ref: tabla[][MCOLS_MAX] de carácter) Matriz de
caracteres donde buscamos.
 * @param nfilas (Valor: entero) Número real de filas de la
matriz (<= NFILAS_MAX i > 0).
 * @param mcols (Valor: entero) Número real de columnas de la
matriz (<= MCOLS_MAX i > 0).
 * @param fila (Ref: entero) Fila inicial y, fila en la que se
encuentra.
 * @param col (Ref: entero) Columna inicial y, columna en la que
se encuentra.
 * @param car (Valor: char) Carácter que se desea buscar.
 * @param orientacion (Valor: enter) Orientación en la que se va
a buscar.
 * @return (booleà) Devuelve cierto si se ha encontrado el
carácter y
 *                          falso en caso contrario.
*/

funció tb_busca_orientacion( /* Parámetros de entrada */ matriz:
taula [][MCOLS_MAX] de caràcter, nfilas: enter, mcols: enter, /*
Parámetros de entrada/salida */ var fila: enter, var col: enter,
/* Parámetros de entrada */     car: caràcter, orientacion:
enter) retorna booleà és
```

- BUSCA POSSIBLES ORIENTACIONS

```
/* A partir de una posición de la matriz (fila y columna
determinadas),
 * crea una tabla con todas las orientaciones en las que la
casilla adyacente
 * contiene el carácter que se pasa como parámetro.
 * Devuelve el número de veces que se ha encontrado el carácter.
La casilla del centro no se evalúa.
 *
 * @param matriz (Ref: tabla[][MCOLS_MAX] de carácter) Matriz de
caracteres donde buscamos.
 * @param nfilas (Valor: entero) Número real de filas de la
matriz (<= NFILAS_MAX i > 0).
 * @param mcols (Valor: entero) Número real de columnas de la
matriz (<= MCOLS_MAX i > 0).
 * @param fila (Valor: entero) Fila central a partir de la cual
buscamos.
 * @param col (Valor: entero) Columna central a partir de la
cual buscamos.
 * @param car (Valor: char) Carácter que se desea buscar.
 * @param orientaciones (Ref: taula[] de enter) Orientación en
la que se va a buscar.
 * @return (enter) Devuelve el número de veces que se ha
encontrado el carácter
 *              en la casillas vecinas.
 */
/* @def MCOLS_MAX
Numero máximo de columnas de las matrices.
*/

constants MCOLS_MAX <- 10;  fconst

funció tb_orientaciones ( /* Parámetros de entrada */ matriz:
taula [][MCOLS_MAX] de caràcter, nfilas: enter, mcols: enter,
fila: enter, col: enter, car: caràcter, /* Parámetros de salida
*/ orientaciones: taula[] d'enter) retorna enter és
```

- DATES IGUALS

```
/**
* fecha_t
* Información de una fecha
*/
tipus
registre fecha_t és
dia, mes , año: enter;
fregistre
ftipus
/*
Compara dos fechas, f1 y f2, y devuelve cierto si son iguales.
@param f1 (Ref: fecha_t) Primera fecha a evaluar.
@param f2 (Ref: fecha_t) Segunda fecha a evaluar.
@return (booleà) Devuelve cierto si las fechas son iguales y
falso en caso contrario.
```

```
*/

funció fechas_iguales ( /* Parámetros de entrada */ f1: fecha_t,
f2: fecha_t) retorna booleà és
```

- COMPARAR DATES

```
Compara dos fechas, f1 y f2, y devuelve un valor entero según la
relación existente entre ellas:

1 - f1 > f2 La primera fecha es más reciente que la segunda.
0 - f1 = f2 Ambas fechas son iguales.
-1 - f1 < f2 La segunda fecha es más antigua que la segunda.
@param f1 (Ref: fecha_t) Primera fecha a evaluar.
@param f2 (Ref: fecha_t) Segunda fecha a evaluar.
@return (enter) Devuelve un valor según la relación existente
entre las fechas:
- 1 si f1 es más reciente que f2.
- 0 si ambas son iguales.
- -1 si f1 es más antigua que f2.
*/

funció compara_fechas ( /* Parámetros de entrada */ f1: fecha_t,
f2: fecha_t) retorna enter és
```

- ESCRIURE RECORD

```
/**
* record_t
* Información de una fecha
*/

tipus
      registre record_t és
            nom, ficha_color: caracter;
            puntuacion: entero;
            fecha: fecha_t;
      fregistre
      registre fecha_t és
            dia,año:enter;
            mes:caracter;
      fregistre
ftipus

/**
Imprime por pantalla la información de un récord siguiendo el
siguiente formato:
dd-mmm-aa Color Puntuación Nombre
Separados por tabulación y con salto de ´línea al final.
@param record (Ref: record_t) Registro con toda la información
del récord.
*/

acció ot_escribe_record ( record: record_t) és
```

- ESCRIURE RECORDS

```
/**
Imprime un número determinado de los récords guardados en la
tabla.
@param records (Ref: tabla[] de record_t) Tabla con los records,
ordenados o no.
@param num_records (Valor: entero) Número máximo de registros a
imprimir.
*/

acció ot_escribe_records /* Parámetros de entrada */ records:
taula[] de record_t, num_records: enter) és
```

- ESCRIURE DATA

```
/**
Muestra por pantalla la fecha formateada: dd-mmm-aaaa
@param fecha (Ref: fecha_t) Fecha a escribir por pantalla.
*/

acció escribe_fecha /* Parámetros de entrada */ fecha: fecha_t)
és
```

- TAULA D'OTHELLO

```
/**
Inicializa el tablero de juego inicial, colocando las primeras
fichas en el lugar correcto.
@param tablero (Ref: tabla de carácter) Tablero con la situación
del juego.
@param dim (Valor: entero) Dimensión real con la que se jugará.
*/

acció ot_tablero_inicial (var tablero: taula [][MCOLS_MAX] de
caràcter, /*Parámetros de entrada*/ dim: enter) és
```

- LLEGIR JUGADOR

```
/**
* jugador_t
* Información de una fecha
*/

tipus
     registre jugador_t és
          nom: taula[10] de caracter;
          color_ficha: caracter;
          persona: boolea;
          puntuacion: enter;
          coordenada: cordenada_t;
     fregistre
```

```
      registre cordenada_t és
            filas,col: enter;
      fregistre
ftipus

/**
Permite dar valor inicial a los datos de un jugador:
1. si es o no una máquina,
2. nombre: si es humano pedirlo, o maquina_color si es máquina
3. color asignado,
4. última tirada = (-1, '-') Coordenada no válida.
5. puntuación inicial = 2
@param jugador (Ref: jugador_t) Información del jugador.
@param color (Valor: carácter) Color asigando al jugador.
*/

acció ot_lee_jugador ( var jugador: jugador_t, color: caràcter)
és
```

- OBTENIR DADES DEL JUGADOR

```
/**
Recupera la información de un jugador: nombre, puntuación...
@param jugador (Ref: jugador_t) Datos del jugador.
@param nombre (Ref: taula[] de caràcter) Nombre del jugador.
@param color (Ref: enter) Color del jugador.
@param maquina (Ref: booleà) Cierto si es tipo máquina y falso
si es humano.
@param fila (Ref: enter) Fila de la última tirada [1..dim]
@param col (Ref: caracter) Columna de la última tirada
['A'..dim]
@param puntuacion (Ref: enter) Puntuación del jugador.
*/

tipus
      registre jugador_t és
            nom: taula[10] de caracter;
            color_ficha: caracter;
            persona: boolea;
            puntuacion: enter;
            coordenada: cordenada_t;
      fregistre
      registre cordenada_t és
            filas,col: enter;
      fregistre
ftipus

acció ot_obtener_datos_jugador (/* Parámetros de entrada */
jugador: jugador_t, /* Parámetros de salida */ var nombre:
taula[] de caràcter, caracter, var maquina: boolea, var fila:
enter, var col: enter, var puntuacio: enter;) és
```