#### Fundamentos de la programación

### Práctica 3: Algunas pistas





#### Práctica 3: tipos

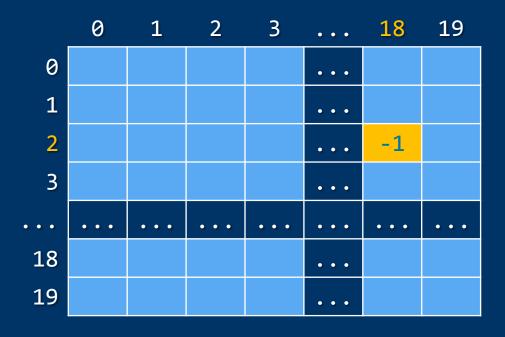
```
// Dimensión del tablero: DIM x DIM
const int DIM = 20;
// Codificación de las casillas del tablero
const int LIBRE = -1;
const int PARED = -2;
const int AGUJERO = -3;
const int EXITO = -4;
// Las casillas >= 0 son bolas
// Número máximo de bolas
const int NB = 10;
// Codificación de las inclinaciones
typedef enum {
  derecha, arriba, izquierda, abajo, nulo
} tDireccion;
```



#### Práctica 3: tipos

```
// Array bidimensional para un tablero
typedef int tTablero[DIM][DIM];
tTablero tablero;

tablero[2][18] = -1;
```



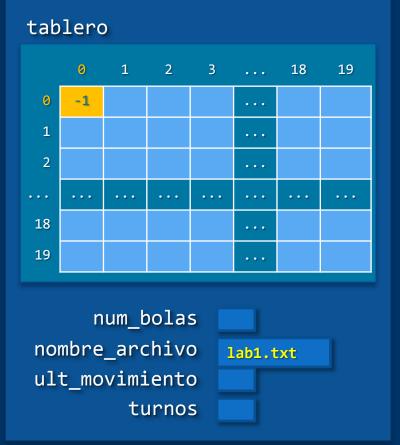


## Carlos Cervigón Rückauer/Luis Hernández Yáñez

#### Práctica 3: tipos

```
// Array del tablero
typedef int tTablero[DIM][DIM];
// Tipo cadena al estilo C
typedef char tCadena[20];
// Estructura para el juego
typedef struct{
   tTablero tablero;
   int num bolas;
   string nombre archivo;
   tDireccion ult movimiento;
   int turnos;
} tJuego;
// Tipo auxiliar
typedef bool tBolasMarcadas[NB];
tJuego juego;
juego.tablero[0][0] = LIBRE;
```

juego





#### Práctica 3: programa principal

Inicializar el tablero de juego

Cargar los datos del archivo en el juego

Mientras no se resuelva y no se quiera salir

Mostrar el tablero

Pedir opción del menú

Procesar la opción elegida

Si se ha resuelto

Mostrar el tablero final

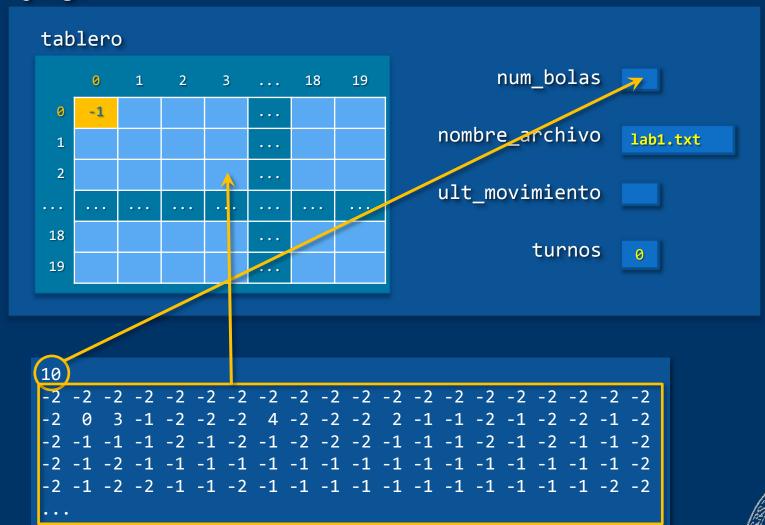




## Carlos Cervigón Rückauer/Luis Hernández Yáñez

#### Práctica 3: Carga del archivo







#### Práctica 3: Carga del archivo

```
Leer el número de bolas
Si el número de bolas es correcto // <= NB
   fila = 0
   Mientras fila < DIM y no se detecte error
      columna = 0
      Mientras columna < DIM y no se detecte error
         Leer entero
         Si es un valor válido y no se repite bola
            Asignar a tablero[fila][col]
            columna++
         Si no
            Error detectado
      Si no se ha detectado error
         fila++
```



#### Práctica 3: Carga del archivo

Después de la lectura hay que validar las fronteras

Verificar que en los bordes hay paredes:

Para todos los valores de k...

```
tablero[0][k] == PARED // Borde superior
tablero[DIM-1][k] == PARED // Borde inferior
```

Para todos los valores de k menos 0 y DIM-1...

```
tablero[k][0] == PARED // Borde izquierdo
tablero[k][DIM-1] == PARED // Borde derecho
```



#### Práctica 3: programa principal

Inicializar el tablero de juego Cargar los datos del archivo en el juego Mientras no se resuelva y no se quiera salir

Mostrar el tablero

Pedir opción del menú Procesar la opción elegida Si se ha resuelto

Mostrar el tablero final

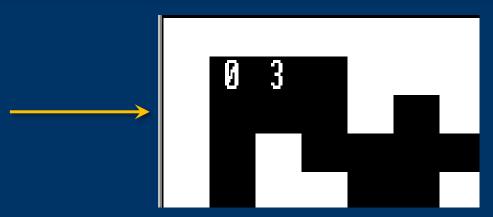




# Carlos Cervigón Rückauer/Luis Hernández Yáñez

#### Práctica 3: mostrar

-2	-2	-2	-2	-2	-2	-2
-2	О	3	-1	-2	-2	-2
-2	-1	-1	-1	-2	-1	-2
-2	-1	-2	-1	-1	-1	-1
-2	-1	-2	-2	-1	-1	-2



Para cada fila

Para cada columna

Si es pared

Mostrar char(219)  $\times$  3

Si no Si es libre

Mostrar tres espacios

Si no Si es agujero

Mostrar char(176)  $\times$  3

Si no

Mostrar el número de bola con espacios





#### Práctica 3: programa principal

Inicializar el tablero de juego

Cargar los datos del archivo en el juego

Mientras no se resuelva y no se quiera salir

Mostrar el tablero

Pedir opción del menú

Procesar la opción elegida

Si se ha resuelto

Mostrar el tablero final





#### Práctica 3: inclinaTablero()

```
inclinaTablero(juego, direccion, terminado)
// Desplaza todas las bolas según la dirección
  switch (direccion)
    case arriba:
      En orden creciente de filas
        En orden creciente de columnas
          Si hay una bola
            desplazaBola(juego, fila, columna, direccion, terminado)
    case abajo:
      En orden decreciente de filas
        En orden creciente de columnas
          Si hay una bola
            desplazaBola(juego, fila, columna, direccion, terminado)
```



#### Práctica 3: inclinaTablero()

```
case izquierda:
    En orden creciente de columnas
    En orden creciente de filas
        Si hay una bola
        desplazaBola(juego, fila, columna, direccion, terminado)
case derecha:
    En orden decreciente de columnas
    En orden creciente de filas
        Si hay una bola
        desplazaBola(juego, fila, columna, direccion, terminado)
```



#### Práctica 3: desplazaBola()

```
desplazaBola(juego, fila, columna, direccion, terminado)
// direccion marca el desplazamiento de los índices
switch (direccion)
  case arriba:
      incrFila = -1; incrColumna = 0;
  case abajo:
      incrFila = 1; incrColumna = 0;
  case izquierda:
      incrFila = 0; incrColumna = -1;
  case derecha:
     incrFila = 0; incrColumna = 1;
```

Mientras no encuentre otra bola o pared Incrementar fila en incrFila Incrementar columna en incrColumna



