Práctica 5
Recorridos y variables

Introducción a la Programación
1er Semestre de 2014

Nota: los ejercicios 1–13 pueden resolverse sin usar variables. Se sugiere fuertemente hacer el esfuerzo de evitar su uso.

1. Funciones y repetición condicional (repaso)

Ejercicio 1

Escribir el procedimiento `MoverNRotativo(n,d)` que mueva el cabezal `n` posiciones en dirección `d` con la siguiente salvedad: en el momento en el que el cabezal no se pueda mover en dirección `d`, el mismo tiene que desplazarse hacia el extremo `opuesto(d)`. Estructure su solución utilizando un `repeat` que utilice el procedimiento `IrAlBorde`.

Ejercicio 2

Escribir el procedimiento `PonerSalteado(n, c, d)` que, dado un número `n`, un color `c` y una dirección `d`, ponga una bolita de color `c` en: la celda actual, la celda a distancia `n` en dirección `d`, la celda a distancia `2n` en dirección `d`, y así siguiendo mientras pueda. **Ayuda**: recuerde la función `puedeMoverN` de la práctica anterior. ¿Se le ocurre alguna forma de resolver el ejercicio actual sin utilizar `puedeMoverN`? ¿Qué problemas tendría su solución si los efectos de `puedeMoverN` no desaparecieran? Discutir las ventajas de que las funciones no produzcan efectos.

2. Recorridos estándar

Ejercicio 3

Escribir las siguientes funciones y procedimientos que sirven para recorrer el tablero. El recorrido se hace avanzando internamente en dirección `d1` y externamente en dirección `d2`. Por ejemplo, la Figura 1 muestra el recorrido cuando `d1` denota `Sur` y `d2` denota `Oeste`; el recorrido empieza en el extremo Noreste, y se va avanzando por columnas hacia el oeste, recorriendo cada columna de norte a sur.

1. `IrAlInicioT(d1, d2)`, que recibe dos direcciones `d1` y `d2` y mueve el cabezal al inicio del recorrido. En otras palabras, el cabezal se mueve al extremo `opuesto(d1) + opuesto(d2)`.

2. `puedeMoverT(d1, d2)` que denota `True` cuando el cabezal se puede avanzar a la siguiente celda. En otras palabras, denota `True` cuando el cabezal no se encuentra en el extremo `d1+d2`.
3. `MoverT(d1, d2)` que mueve el cabezal a la siguiente celda del recorrido. En otras palabras, el cabezal se ubica en la celda lindante en dirección `d1` de ser posible. Si no es posible, entonces el cabezal se mueve en dirección `d2` y hacia el extremo `opuesto(d1)`. ¿Cuál es la precondición de este procedimiento?

13	9	5	1
14	10	6	2
15	11	7	3
16	12	8	4

Figura 1: Ejemplo del recorrido del tablero con direccion `d1` = Sur y `d2` = Oeste.

Ejercicio 4

Identifique el esquema de recorrido de las siguientes funciones y procedimientos, indicando qué elementos se recorren (e.g., las celdas del tablero, las celdas de una fila, las celdas de una columna, las celdas que tienen tales bolitas, etc.), los procedimientos que se usan para ir a la primer celda, avanzar a la siguiente celda y procesar cada celda, y la función que se utiliza para determinar si hay más celdas por procesar dentro del recorrido. Escriba, asimismo, las precondiciones de las funciones y procedimientos.

```
procedure LimpiarTablero() {
    IrAlInicioT(Norte, Este)
    while(puedeMoverT(Norte, Este)) {
        LimpiarCelda()
        MoverT(Norte, Este)
    }
    LimpiarCelda()
}
```

```
procedure LimpiarFila() {
    IrAlExtremo(Oeste)
    while(puedeMover(Este)) {
        LimpiarCelda()
        Mover(Este)
    }
    LimpiarCelda()
}
```

```
procedure hayVaciaHacia(d) {
    while(puedeMover(d) && not vacia()) {
        Mover(d)
    }
    return(vacia())
}
```

```
procedure IrASiguienteColor(c) {
    while(not hayBolitas(c)) {
        MoverT(Norte, Este)
    }
}
```

```
procedure haySiguienteColor(c) {
    while(puedeMoverT(Norte, Este) &&
          not hayBolitas(c)) {
        MoverT(Norte, Este)
    }
    return(hayBolitas(c))
}
```

```
procedure IrAUltimaConColor(c) {
    while(haySiguienteColor(c)) {
        IrASiguienteColor(c)
    }
}
```

Ejercicio 5

Escribir un procedimiento `PonerEnVacias(c)` que ponga una bolita de color `c` en todas las celdas vacías del tablero. El procedimiento debe utilizar un esquema de recorrido que recorra todas las celdas del tablero.

Ejercicio 6

Escribir un procedimiento `DuplicarBolitas()` que duplique la cantidad de bolitas de cada celda del tablero. Es decir, si una celda contiene 8 bolitas azules, 4 rojas y una verde, entonces deberá contener 16 bolitas azules, 8 rojas y 2 verdes al finalizar el procedimiento. Se sugiere escribir un procedimiento que duplique la cantidad de bolitas de una celda.

Ejercicio 7

Escribir la función `hayCeldaCromatica()` que denote `True` si alguna celda del tablero contiene una bolita de cada color. Para ello, escribir una función `esCromatica` que indique si la celda actual tiene una bolita de cada color.

Ejercicio 8

Escribir un procedimiento `DegradarTablero(c,deg)` que, dado un color `c` y un número `deg`, haga lo siguiente para cada celda del tablero. Si hay más de `deg` bolitas de color `c` en una celda, entonces el procedimiento saca `deg` bolitas de la celda; caso contrario, saca todas las bolitas de color `c`. **Ayuda:** este ejercicio es similar al procedimiento `DegradarCuadrado` de la Práctica 3 (parte: repetición indexada); trate de reutilizar procedimientos del mismo.

Ejercicio 9

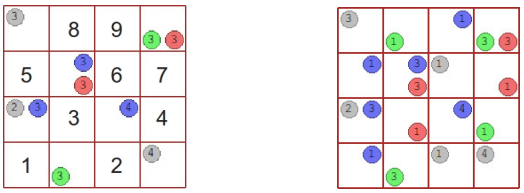
Escribir el procedimiento `BuscarSiguienteVacia(d1,d2)` que posicione el cabezal en la siguiente celda vacía del tablero en un recorrido en direcciones `d1 + d2` (ver Ejercicio 3). Puede suponer que dicha celda siempre existe.

Ejercicio 10

Escribir el procedimiento `DistribuirVacias` que ponga una bolita en cada celda vacía del tablero, de forma tal que los colores de las bolitas nuevas alternen entre `Azul`, `Negro`, `Rojo` y `Verde` (en ese orden) en un recorrido del tablero en direccion `Este + Norte` (ver Figura 2). Recordar que no está permitida la anidación de repeticiones. **Ayuda:** escribir un procedimiento `PonerUnaDeCadaEnCeldasVacias` que recorra los colores y, por cada color `c`, busque una celda vacía y ponga una bolita de color `c`. Luego, repita este procedimiento mientras queden celdas vacías.

Ejercicio 11

Escribir un procedimiento `RecorrerComoReloj(c)` que realice la siguiente acción: comienza moviéndose hacia el `Norte` tantas celdas como bolitas de color `c` haya en la celda actual, una vez ahí repite la acción pero moviéndose el `Este` (tomando la cantidad de bolitas de color `c` de la nueva celda actual). Continuar moviéndose cambiando la dirección en el sentido de las agujas del reloj, hasta caer en una celda que no tenga bolitas de color `c`. `Sugerencia`: escribir



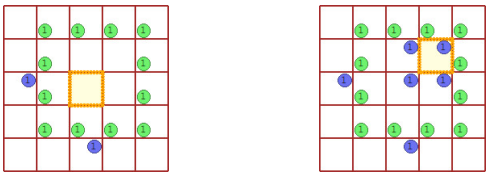
(a) (b)

Figura 2: Ejemplo de la aplicación de `DistribuirVacias`. El Tablero (a) tiene 9 celdas vacías, que se recorrerían en el orden indicado. El Tablero (b) muestra el resultado esperado en este caso.

un procedimiento que recorra una vez cada dirección, y luego ponga este procedimiento dentro de un recorrido. ¿Qué precondición tiene el procedimiento `RecorrerComoReloj(c)`?.

Ejercicio 12

Escribir un procedimiento `Rellenar(c, r)` que, suponiendo que la celda actual se encuentra dentro de un rectángulo vacío (de dimensión desconocida) demarcado por bolitas de color `c`, rellene su interior con bolitas de color `r` (ver Figura 3). Notar que no hay bolitas de color `c` dentro del rectángulo. `Sugerencia`: escriba los procedimientos `IrAInicioRect`, `MoverRect`, y la función `puedeMoverRect` que implementen un recorrido por el rectángulo.



(a) (b)

Figura 3: El Tablero (b) muestra el resultado de aplicar `Rellenar(Verde, Azul)` al Tablero (a).

Ejercicio 13

Suponga la siguiente codificación de movimientos usando colores:

1 azul	moverse en dirección Norte
2 azules	moverse en dirección Este
3 azules	moverse en dirección Sur
4 azules	moverse en dirección Oeste
verdes	distancia
0 azules	finalizar

1. Escribir un procedimiento `MoverPorAzul` que lea la codificación de la celda actual y pase a la celda siguiente según la misma. Por ejemplo, si la celda tiene una bolita azul y dos verdes, debe moverse dos celdas al Norte. Si no tiene bolitas azules, el cabezal no debe moverse. Puede suponer que ninguna celda tiene más de 4 bolitas azules. ¿Cuál es la precondición de este procedimiento?

2. Escribir un procedimiento `RecorridoPorAzul`, que recorra las celdas del tablero según la codificación, colocando una bolita negra en cada celda visitada. El recorrido termina cuando se llega a una celda con bolitas rojas. ¿Cuál es la precondición del procedimiento?
3. Escriba la función `recorridoAzulValido` que indique si el recorrido es posible y termina. Para ello, considere (a) escribir la función `puedeMoverAzul` que denote si el movimiento siguiente es valido, y (b) marcar cada celda recorrida agregando 5 bolitas azules para saber si el camino pasa dos veces por el mismo lugar (en ese caso, el camino no termina).

3. Variables

Ejercicio 14

Corresponder las siguientes oraciones que describen propósitos con sus respectivas funciones.

- Denota el mayor entre `n1` y `n2`.
- Denota de cuál de los colores `c1` y `c2` hay mas bolitas en la celda actual.
- Denota el color del cual hay más bolitas en la celda actual.
- Denota la cantidad de bolitas (de cualquier color) de la celda actual.
- Denota la cantidad de bolitas de entre `c1` y `c2` que más bolitas tiene.
- Denota la cantidad de bolitas de color `c` que tiene el tablero.
- Denota cual es la mayor cantidad de bolitas de color `c` que tiene alguna celda del tablero.

```
function a(c1, c2) {
  mayor := c1
  if(nroBolitas(c2) > nroBolitas(c1)) {
    mayor := c2
  }
  return(mayor)
}
```

```
function b(n1, n2) {
  mayor := n1
  if(n1 > n2) {
    mayor := n2
  }
  return(mayor)
}
```

```
function c() {
  mayor := minColor()
  repeatWith c in minColor()..maxColor() {
    mayor := a(mayor, c)
  }
  return(mayor)
}
```

```
function d() {
  cant := 0
  repeatWith c in minColor()..maxColor() {
    cant := cant + nroBolitas(c)
  }
  return(cant)
}
```

```
function e(c1, c2) {
  return(B(nroBolitas(c1), nroBolitas(c2)))
}
```

```
function f(c1, c2) {
  return(nroBolitas(A(c1, c2)))
}
```

```
function g(c) {
  IrAlInicioT(Norte, Este)
  cant := 0
  while(puedeMoverT(Norte, Este)) {
    cant := cant + nroBolitas(c)
    MoverT(Norte, Este)
  }
  return(cant + nroBolitas(c))
}
```

```
function h(c) {
  IrAlInicioT(Norte, Este)
  mayor := nroBolitas(c)
  while(puedeMoverT(Norte, Este)) {
    mayor := b(mayor, nroBolitas(c))
    MoverT(Norte, Este)
  }
  return(b(mayor, nroBolitas(c)))
}
```

Ejercicio 15

Encuentre los errores en los siguientes procedimientos. **Justifique**.

```
procedure P(p) {
    p := p + 1;
    PonerN(p, Azul)
}
```

```
procedure Q(p) {
    if(p /= 3) {
        v := 3
    }
    PonerN(v, Azul)
}
```

```
procedure R() {
    PonerN(v, Azul)
    MoverN(v, Este)
}
```

```
procedure S(p) {
    v := Este {
    MoverN(5, v)
    v := 5
    MoverN(v, Este)
}
```

Ejercicio 16

Asocie las siguientes oraciones que describen propósitos de variables con sus respectivas variables. En base a su respuesta, ¿cual debería ser el nombre de `var` en cada función? Nota: `max` es una función que denota el mayor de dos números.

- Denota la cantidad de bolitas de color `c` que ya se recorrieron.
- Denota la mayor cantidad de bolitas de color `c` que tenía alguna de las celdas recorridas.

```
function nroBolitasTotal(c) {
//nro bolitas c en el tablero
  IrAlInicioT(Sur, Oeste)
  var := 0
  while(puedeMoverT(Norte, Este)) {
    var := var + nroBolitas(c)
    MoverT(Norte, Este)
  }
  return(var + nroBolitas(c))
}
```

```
function h(c) {
//nro maximo bolitas c en una celda
  IrAlInicioT(Sur, Oeste)
  var := nroBolitas(c)
  while(puedeMoverT(Norte, Este)) {
    var := max(var, nroBolitas(c))
    MoverT(Norte, Este)
  }
  return(max(var, nroBolitas(c)))
}
```

Ejercicio 17

Escribir una función `nroVacias()` que denote la cantidad de celdas vacías del tablero. Orga-

nice su solución con un recorrido del tablero que utilice una variable `vacias` cuyo propósito sea denotar la cantidad de celdas vacías que se van recorriendo.

Ejercicio 18

Escribir un procedimiento `IrANesimaVacia(n)` que posicione el cabezal en la celda vacía número `n` que se encuentra en un recorrido del tablero (puede recorrer el tablero de la forma que desee). Por ejemplo, `IrANesimaVacia(1)` posiciona el cabezal en la primer celda vacía, `IrANesimaVacia(2)` posiciona el cabezal en la segunda celda vacía, etc. Organice su solución con un recorrido que utilice una variable `vacias_vistas` cuyo propósito sea denotar la cantidad de celdas vacías que ya se recorrieron.

Ejercicio 19

Escribir una función `minColor` que denote el color más chico (en el orden Azul, Negro, Rojo, Verde) del cual hay bolitas en la celda actual. ¿Qué ocurre si en la celda actual no hay bolitas de ningún color? ¿Su solución funciona si se agregan más colores a GOBSTONES? En caso negativo, modifique su solución para que funcione con cualquier cantidad de colores (ayuda: considere un recorrido de colores que use una variable `color_actual` cuyo propósito sea denotar el color que se está recorriendo).

Ejercicio 20

Escribir las funciones `nroFilas` y `nroColumnas` que retornen la cantidad de filas y columnas del tablero. ¿Cómo puede calcular la cantidad celdas?

Ejercicio 21

Escriba las funciones `coordenadaX` y `coordenadaY` que retornen la coordenada columna y la coordenada fila de la celda actual, respectivamente. Asuma que 0 es la coordenada de la primer fila y columna.

Ejercicio 22

Escribir un procedimiento `IrACoordenada` que reciba dos parámetros `x` e `y` y mueva el cabezal a la columna `x` y fila `y`. Asuma que 0 es la coordenada de la primer fila y columna. ¿Cuál es la precondición del procedimiento?

4. Interpretación de enunciados complejos

Ejercicio 23

Resolver en papel el parcial "Batalla Naval" del segundo semestre del 2009.[1]

Ejercicio 24

Resolver en papel el parcial "Juego de la Vida" del segundo semestre del 2010.[2]

[1] https://sites.google.com/site/inpr2009c2/repositorio-de-archivos/Parcial1.pdf
[2] https://sites.google.com/site/inpr2010c2/repositorio/PrimerParcialC1C2.pdf

Ejercicio 25

Resolver en papel el parcial "Buscaminas II" del primer semestre del 2011.[3]

Ejercicio 26

Resolver en papel el parcial "SameGame" del segundo semestre del 2011.[4]

[3] https://sites.google.com/site/inpr2011s1/repositorio/PrimerParcialB.pdf
[4] https://sites.google.com/site/inpr2011s2/repositorio/samegame.pdf