PRÁCTICA 7

# // EJ 1

### A)

**function cantidadTotalDeBolitasAqui() {**

**/\***

**Prop.: Describe la cantidad total de bolitas en la celda actual**

**Prec.: Ninguna**

**Tipo: Número**

**\*/**

**return (nroBolitas(Azul) + nroBolitas(Negro) + nroBolitas(Rojo) + nroBolitas(Verde))**

**}**

**program {**

**return (cantidadTotalDeBolitasAqui())**

**}**

### B)

**function hayMas5BolitasAcá() {**

**/\***

**Prop.: Indica si hay mas de 5 bolitas en la celda actual**

**Prec.: Ninguna**

**Tipo: Booleano**

**\*/**

**return (**

**cantidadTotalDeBolitasAqui() > 5**

**)**

**}**

**program {**

**return (hayMas5BolitasAcá())**

**}**

### C)

### D)

### E)

**function hayDeTodosLosColores() {**

**/\***

**Prop.: Indica si hay en la celda actual**

**bolitas de todos los colores**

**Prec.: Ninguna**

**Tipo: Booleano**

**\*/**

**return (hayBolitas(Azul) && hayBolitas(Negro) && hayBolitas(Rojo) && hayBolitas(Verde))**

**}**

**program {**

**return (hayDeTodosLosColores())**

**}**

### F)

### G)

**function faltanBolitasDeAlgunColor() {**

**/\***

**Prop.: Indica si en la celda actual faltan**

**bolitas de algun color**

**Prec.: Ninguna**

**Tipo: Booleano**

**\*/**

**return (**

**not(**

**hayBolitas(Azul) &&**

**hayBolitas(Negro) &&**

**hayBolitas(Rojo) &&**

**hayBolitas(Verde)**

**)**

**)**

**}**

**program {**

**return (faltanBolitasDeAlgunColor())**

**}**

# **// EJ 2** (HECHO)

## BIBLIOTECA

### A)

**function esCeldaVacía() {**

**/\***

**Propósito: Indica si la celda actual está vacía.**

**Precondiciones: Ninguna.**

**Tipo : Booleano.**

**\*/**

**return (not (hayBolitas(Azul)) && not (hayBolitas(Negro)) && not(hayBolitas(Rojo))**

**&& not(hayBolitas(Verde)))**

**}**

### B)

**function tieneUnaDeCada() {**

**/\***

**Propósito: Indica si la celda actual tiene al menos una bolita de cada color.**

**Precondiciones: Ninguna.**

**Tipo : Booleano.**

**\*/**

**return (hayBolitas(Azul) && hayBolitas(Negro) && hayBolitas(Rojo) && hayBolitas(Verde))**

**}**

### C)

**function esCeldaConBolitas() {**

**/\***

**Propósito: Indica si la celda actual tiene al menos una bolita, de cualquier color.**

**Precondiciones: Ninguna.**

**Tipo : Booleano.**

**\*/**

**return (hayBolitas(Azul) || hayBolitas(Negro) || hayBolitas(Rojo) || hayBolitas(Verde))**

**}**

# **// EJ 3**

### A)

**function cantidadDeSoldadosDe\_(colorDelEjército) {**

**/\***

**Prop.: Describe la cantidad de soldados**

**representado por \*colorDelEjército\***

**que hay en la ubicacion actual**

**Prec.: \*colorDelEjército\* == Rojo || Negro**

**Param.: \*colorDelEjército\*. Color. El Color**

**del bando de lucha**

**Tipo: Número**

**\*/**

**return (nroBolitas(colorDelEjército))**

**}**

**program {**

**return (cantidadDeSoldadosDe\_(Rojo))**

**}**

### B)

### C)

### 

### 

### D)

### 

### 

### E)

### 

### 

### F)

# **// EJ 4**

## 

# **// EJ 5 (HECHO)**

**function árbol(){**

**/\***

**PROPOSITO: Describre un árbol.**

**PRECONDICION: Ninguna.**

**TIPO: Color.**

**\*/**

**return(Verde)**

**}**

**function semilla(){**

**/\***

**PROPOSITO: Describe una semilla.**

**PRECONDICION: Ninguna.**

**TIPO: Color.**

**\*/**

**return(Rojo)**

**}**

**function bomba(){**

**/\***

**PROPOSITO: Describe una bomba.**

**PRECONDICION: Ninguna.**

**TIPO: Color.**

**\*/**

**return(Negro)**

**}**

**function nutriente(){**

**/\***

**PROPOSITO: Describe un nutriente.**

**PRECONDICION: Ninguna.**

**TIPO: Color.**

**\*/**

**return(Azul)**

**}**

**procedure GerminarSemilla(){**

**/\***

**PROPOSITO: Transforma una semilla en un arbol en la ubicación actual, consumiento 3 nutrientes.**

**PRECONDICION: Ninguna.**

**\*/**

**if(haySemilla() && cantidadNutrientes() >= 3){**

**Sacar\_Nutrientes(3)**

**SacarUnaSemilla()**

**PonerUnÁrbol()**

**}**

**}**

**procedure AlimentarÁrboles(){**

**/\***

**PROPOSITO: Alimenta los árboles de la ubicación actual consumiendo 1 nutriente cada uno.**

**Si no hay 1 nutriente por cada árbol, se consumen todos los que haya.**

**PRECONDICION: Ninguna.**

**\*/**

**repeat(cantidadNutrientes()){**

**SacarUnNutriente()**

**}**

**}**

**function cantidadNutrientes(){**

**/\***

**PROPOSITO: Indica la cantidad de nutrientes en la ubicación actual.**

**PRECONDICION: Ninguna.**

**TIPO: Numero.**

**\*/**

**return(nroBolitas(nutriente()))**

**}**

**function haySemilla(){**

**/\***

**PROPOSITO: Indica si hay al menos una semilla en la ubicación actual.**

**PRECONDICION: Ninguna.**

**TIPO: Booleano.**

**\*/**

**return(hayBolitas(semilla()))**

**}**

**function cantidadÁrboles(){**

**/\***

**PROPOSITO: Indica la cantidad de árboles en la ubicación actual.**

**PRECONDICION: Ninguna.**

**TIPO: Numero.**

**\*/**

**return(nroBolitas(árbol()))**

**}**

**procedure ExplotarBomba(){**

**/\***

**PROPOSITO: Explota una bomba en la celda actual, derribando 5 árboles en la ubicación actual**

**y 3 en la celda lindante al norte. Si no puede mover al norte, solo derriba arboles en la ubicación actual.**

**PRECONDICION: Ninguna.**

**\*/**

**if(hayBombaAcá()){**

**SacarUnaBomba()**

**Derribar\_ÁrbolesAcá(5)**

**Derribarl\_Al\_(3, Norte)**

**}**

**}**

**function hayBombaAcá(){**

**/\***

**PROPOSITO: Indica si hay una bomba en la ubicación actual.**

**PRECONDICION: Ninguna.**

**TIPO: Booleano.**

**\*/**

**return(hayBolitas(Negro))**

**}**

**procedure Derribar\_ÁrbolesAcá(cantidadDeÁrbolesASacar){**

**/\***

**PROPOSITO: Saca \*cantidadDeÁrbolesASacar\* árboles de la celda actual si hubiera más de esa cantidad, sino, saca todos los árboles que haya.**

**PRECONDICION: Ninguna.**

**PARAMETROS: \*cantidadDeÁrbolesASacar\* - Número. Indica cuántos árboles se sacarán.**

**\*/**

**if(cantidadÁrboles() >= cantidadDeÁrbolesASacar){**

**repeat(cantidadDeÁrbolesASacar){**

**SacarUnÁrbol()**

**}**

**}**

**elseif(cantidadÁrboles() < cantidadDeÁrbolesASacar){**

**repeat (cantidadÁrboles()){**

**SacarUnÁrbol()**

**}**

**}**

**}**

**procedure Derribarl\_Al\_(cantidadDeÁrbolesASacar, dirección){**

**/\***

**PROPOSITO: Derriba \*cantidadDeÁrbolesASacar\* árboles de la ubicación lindante hacia el \*dirección\*.**

**PRECONDICION: Ninguna.**

**PARAMETROS:**

**\*cantidadDeÁrbolesASacar\*- Número. Cantidad de árboles a derribar.**

**\*direccion\*- Dirección. Hacia donde se derribarán árboles.**

**\*/**

**if(puedeMover(dirección)){**

**Mover(dirección)**

**}**

**Derribar\_ÁrbolesAcá(3)**

**Mover(opuesto(dirección))**

**}**

**procedure Polinizar(){**

**/\***

**PROPOSITO: Poliniza la celda lindante al este, generando 3 semillas menos por**

**cada árbol que haya en la ubicación actual.**

**PRECONDICION: Debe haber al menos una celda lindante al este.**

**\*/**

**Poner\_SemillasAl\_(cantidadÁrboles() - 3, Este)**

**}**

# **// EJ 6**