Proyecto de Simulación y Programación Declarativa.

Tema: Agentes Inteligentes.

Autora: Thalia Blanco Figueras.

Grupo: C-412.

Problema:

El ambiente en el cual intervienen los agentes es discreto y tiene la forma de un rectángulo de $N \times M$. El ambiente es de información completa, por tanto todos los agentes conocen toda la información sobre el agente. El ambiente puede variar aleatoriamente cada t unidades de tiempo. El valor de t es conocido.

Las acciones que realizan los agentes ocurren por turnos. En un turno, los agentes realizan sus acciones, una sola por cada agente, y modifican el medio sin que este varíe a no ser que cambie por una acción de los agentes. En el siguiente, el ambiente puede variar. Si es el momento de cambio del ambiente, ocurre primero el cambio natural del ambiente y luego la variación aleatoria. En una unidad de tiempo ocurren el turno del agente y el turno de cambio del ambiente.

Los elementos que pueden existir en el ambiente son obstáculos, suciedad, niños, el corral y los agentes que son llamados Robots de Casa. A continuación se precisan las características de los elementos del ambiente:

Obstáculos: estos ocupan una única casilla en el ambiente. Ellos pueden ser movidos, empujándolos, por los niños, una única casilla. El Robot de Casa sin embargo no puede moverlo. No pueden ser movidos ninguna de las casillas ocupadas por cualquier otro elemento del ambiente.

Suciedad: La suciedad es por cada casilla del ambiente. Solo puede aparecer en casillas que previamente estuvieron vacías. Esta, o aparece en el estado inicial o es creada por los niños.

Corral: El corral ocupa casillas adyacentes en número igual al del total de niños presentes en el ambiente. El corral no puede moverse. En una casilla del corral solo puede coexistir un niño. En una casilla del corral, que esté vacía, puede entrar un robot. En una misma casilla del corral pueden coexistir un niño y un robot solo si el robot lo carga, o si acaba de dejar al niño.

Niño: Los niños ocupan solo una casilla. Ellos en el turno del ambiente se mueven, si es posible (si la casilla no está ocupada: no tiene suciedad, no está el corral, no hay un Robot de Casa), y aleatoriamente (puede que no ocurra movimiento), a una de las casilla adyacentes. Si esa casilla está ocupada por un obstáculo este es empujado por el niño, si en la dirección hay más de un obstáculo, entonces se desplazan todos. Si el obstáculo está en una posición donde no puede ser empujado y el niño lo intenta, entonces el obstáculo no se mueve y el niño ocupa la misma posición.

Los niños son los responsables de que aparezca suciedad. Si en una cuadrícula de **3 por 3** hay un solo niño, entonces, luego de que él se mueva aleatoriamente, una de las casillas de la cuadrícula anterior que esté vacía puede haber sido ensuciada. Si hay dos niños se pueden ensuciar hasta 3. Si hay tres niños o más pueden resultar sucias hasta 6.

Los niños cuando están en una casilla del corral, ni se mueven ni ensucian.

Si un niño es capturado por un Robot de Casa tampoco se mueve ni ensucia.

Robot de Casa: El Robot de Casa se encarga de limpiar y de controlar a los niños. El Robot se mueve a una de las casillas adyacentes, las que decida. Solo se mueve una casilla sino carga un niño. Si carga un niño pude moverse hasta dos casillas consecutivas.

También puede realizar las acciones de limpiar y cargar niños. Si se mueve a una casilla con suciedad, en el próximo turno puede decidir limpiar o moverse. Si se mueve a una casilla donde está un niño, inmediatamente lo carga. En ese momento, coexisten en la casilla Robot y niño.

Si se mueve a una casilla del corral que está vacía, y carga un niño, puede decidir si lo deja esta casilla o se sigue moviendo. El Robot puede dejar al niño que carga en cualquier casilla. En ese momento cesa el movimiento del Robot en el turno, y coexisten hasta el próximo turno, en la misma casilla, Robot y niño.

Objetivos: El objetivo del Robot de Casa es mantener la casa limpia. Se considera la casa limpia si el 60 % de las casillas vacías no están sucias.

Solución:

Consideraciones acerca de los elementos del ambiente:

- **Obstáculos:**
    - Existen inicialmente y no se crean.
    - Ocupan una única casilla (en una casilla donde exista un obstáculo no existe ningún otro elemento del ambiente).
    - Solo pueden ser desplazados por niños y en los turnos de cambio de ambiente.
    - Solo pueden ser desplazados una casilla por turno de cambio de ambiente.
    - Pueden ser desplazados en una determinada dirección siempre y cuando, la posición resultante del desplazarlo en esa dirección, sea una casilla vacía o una casilla ocupada por otro obstáculo que a su vez puede ser desplazado también en dicha dirección.
- **Suciedad:**
    - Puede existir inicialmente y puede ser creada por los niños en los turnos de cambio de ambiente.
    - Solo puede ser creada en casillas previamente vacías.
    - Pueden compartir la casilla con:
        - Un robot.
        - Un robot y un niño siempre que el robot lo tenga cargado.
- **Corral:**
    - La cantidad de casillas que ocupa el corral es igual a la cantidad de niños.
    - Las casillas que ocupa el corral tienen que ser adyacentes.
    - No puede moverse (las casillas en que fue ubicado el corral inicialmente se mantienen hasta que culmine la simulación).
    - En una casilla del corral puede haber:

            - Un niño.
            - Un robot.
            - Un robot y un niño si el niño es cargado por el robot.
            - Un robot y un niño si el niño es dejado por el robot.
- **Niño:**
    - Si el niño es cargado por robot ó si está dentro del corral:
        - No se mueve.
        - No ensucia.
    - Si el niño está fuera del corral y no está siendo cargado por un robot:
        - Se mueve (o no) en los turnos de cambio de ambiente de forma aleatoria. En caso de que decida moverse es posible a casillas adyacentes donde:
            - No haya suciedad.
            - No esté el corral (se considera que un niño no puede entrar en el corral por sí solo).
            - No haya robot.
            - Si hay obstáculo y puede ser empujado (teniendo en cuenta el criterio anteriormente mencionado de desplazamiento de obstáculos).
        - Si en una cuadrícula de 3 por 3 hay un solo niño que se mueve, entonces, luego de que él se mueva aleatoriamente, una de las casillas de la cuadrícula anterior que esté vacía puede haber sido ensuciada (o no). Si hay dos niños que se mueven en esa cuadrilla se pueden ensuciar hasta 3 y si hay tres niños o más pueden resultar sucias hasta 6.
    - Por tanto un niño que se encuentra en una determinada casilla solo puede compartir dicha casilla con:
        - El corral.
        - El corral y un robot si este lo está cargando.
        - El corral y un robot si este lo acaba de dejar en esa casilla.
        - Un robot si acaba de ser dejado por este.
        - Un robot si está siendo cargado por este.
        - Una posición sucia y un robot por el que está siendo cargado.
- **Robot de Casa (Agentes):**
    - Se encarga de limpiar y de controlar a los niños.
    - Si carga un niño:
        - No puede moverse a una posición con niño.
        - Puede moverse hasta dos casillas consecutivas.
        - Puede dejar al niño que carga en cualquier casilla que no esté sucia.
        - Si se mueve a una casilla del corral que está vacía, puede decidir si lo deja en esta casilla o se sigue moviendo.
        - Si deja al niño cesa el movimiento del Robot en el turno, y coexisten hasta el próximo turno, en la misma casilla, Robot y niño.
        - Si se mueve a una casilla sucia, puede decidir si la limpia cargando al niño, lo cual le tomaría ese turno, o si decide seguir moviéndose.
    - Si no carga un niño:
        - Solo puede moverse a casillas adyacentes que no estén ocupadas por obstáculos ni otros robots.

        - Si se mueve a una casilla con suciedad en el próximo turno puede decidir limpiarla o moverse.
        - Si se mueve a una casilla donde está un niño, inmediatamente lo carga. En ese momento, coexisten en la casilla Robot y niño.
    - Un robot no puede compartir casilla con (y por tanto no puede moverse hacia casillas con dichas condiciones):
        - Un obstáculo.
        - Otro robot.
        - Un niño si está cargando a otro niño.
        - Un niño que no esté cargando o que no acaba de dejar.

Variables que se tienen en cuenta para la simulación:

- Tiempo durante el cual se estará analizando la simulación: **t**.
- Tiempo global de la simulación: **t_g** (cuando **t_g=t** culmina la simulación).
- Cada cuanto tiempo el ambiente puede cambiar: **t_c**.
- Tiempo del próximo cambio de ambiente: **t_cambio**.

Variables que se necesitan para generar el ambiente:

- Cantidad de filas del ambiente: **n**.
- Cantidad de columnas del ambiente: **m**.
- Cantidad de Obstáculos con que se crea el ambiente: **o_0**.
- Cantidad de Casillas Sucias inicialmente con que se crea el ambiente: **s_0**.
- Cantidad de niños con que se crea el ambiente: **n_0**.
- Cantidad de Robots con que se crea el ambiente: **r_0**.
- Cantidad de Robots con que se crea el ambiente: **r_0**.

Dirección de los movimientos que se pueden efectuar dada una posición (x,y):

- Norte: (x-1 , y).
- Sur : (x+1 , y).
- Este : (x , y+1).
- Oeste: (x , y-1).

Dirección de las posiciones en las que puede generar basuras los niños que se mueven (posiciones asociadas a su cuadrícula de 3x3), dada su posición (x,y):

- Norte: (x-1 , y).
- Sur : (x+1 , y).
- Este : (x , y+1).
- Oeste: (x , y-1).
- Noreste: (x-1,y+1)
- Sureste: (x+1,y+1)
- Noroeste:(x-1,y-1)
- Suroeste:(x+1,y-1)

Ideas seguidas para solucionar el problema:

Para solucionar el problema:

- Primeramente se generará un ambiente inicial con los valores de entrada para niños, obstáculos, corrales, suciedades, robots así como del tamaño del rectángulo. La cantidad de casillas del rectángulo debe ser mayor que los valores de entradas asociados a los elementos del ambiente para garantizar que sean posibles los movimientos tanto de niños como de robots.
- Durante todo el tiempo que dure la simulación (mientras que **t\=t_g**):
    - Se estarán generando cambios de ambiente por los agentes (robots). En caso que no haya suciedad ni niños fueras del corral no tiene sentido que se pongan a caminar aleatoriamente, por lo que el porciento de limpieza a partir de ese momento, y hasta que culmine la simulación, será del 100%.
    - Cada t_{cambio} se estarán produciendo cambios en el ambiente, por los niños que decidan moverse.
    - Como se conoce la cantidad de casillas sucias y la cantidad de casillas total del ambiente (**N x M**), se calcula que porciento de la casa se encuentra limpio: $\frac{NxM-SUCIAS}{NxM} * 100$, ya que el objetivo es que los robots mantengan la casa con al menos un 60% de limpieza.
    - Como la simulación puede durar mucho tiempo (en dependencia del valor de entradas), se puede dificultar ver los porcientos de casa limpia, por lo que para tener una idea general de la eficiencia de los robots, se promedian dichos porcientos, al finalizar la simulación.

Ideas seguidas para implementar el problema:

- Para la inicialización del ambiente (función `crearA`):
    - Se crea primeramente el corral:
        - Se crea una posición aleatoria dentro de los límites del rectángulo de **N x M**.
        - A partir de esta posición se crean $n_0 - 1$ posiciones adyacentes.
    - Se crean las posiciones ocupadas por los niños:
        - Como los niños pueden ocupar posiciones dentro y fuera del corral se generan n_0 posiciones aleatorias dentro de los límites del rectángulo de **N x M**.
        - Se quitan las posiciones que coinciden con las generadas del corral, ya que los niños que se encuentran en corral como no se mueven y no ensucian, se asume que durante toda la simulación deben permanecer ahí (es estrategia para evitar que generen basura y alcanzar el objetivo).
        - Se generan también las posiciones del corral que se encuentran sin niños, o sea, se tiene una lista que resulta de eliminar, de las posiciones del corral, las que coincidían con las posiciones aleatorias para niños, inicialmente generadas (es análogo al punto anterior, pero es una lista nueva que resultará de interés para los robots que cargan niños ya que en solo estas posiciones podrán dejarlos).
    - Se crean las posiciones ocupadas por los robots:

        - Se generan r_0 posiciones aleatorias dentro de los límites del rectángulo de **N x M** que no coincidan con las posiciones del **corral** ni de los **niños**.
    - Se crean las posiciones ocupadas por suciedades:
        - Se generan s_0 posiciones aleatorias dentro de los límites del rectángulo de **N x M** que no coincidan con las posiciones del **corral** ni de los **niños** ni de los **robots**.
    - Se crean las posiciones ocupadas por obstáculos:
        - Se generan o_0 posiciones aleatorias dentro de los límites del rectángulo de **N x M** que no coincidan con las posiciones del **corral** ni de los **niños** ni de los **robots** ni de las **suciedades**.
- Para los cambios de ambiente (función `cambioN`) que equivalen a los momentos en que los niños se mueven, por cada niño del ambiente:
    - Calcular sus posiciones adyacentes que se encuentran dentro de los límites del rectángulo que se genera de **N x M**.
    - Calcular cuales de las posiciones anteriores pueden ser ocupadas, dígase las que no están siendo ocupadas ni por otros **niños**, ni por **robots**, ni por **suciedad**, ni por **corral** (se asume que un niño no entra por sí solo en el corral), ni por **obstáculos** que no puedan ser desplazados (si en alguna de su direcciones adyacentes se encuentra algún obstáculo, se calcula si este puede ser desplazado en la dirección en la que se encuentra este con respecto al niño).
    - Luego de manera aleatoria se decide si el niño en este cambio de ambiente desea o no moverse.

      En caso de que no decida moverse este no está contemplado para generar basuras ni mover obstáculos, y su posición se mantiene.

      En caso de que decida moverse, solo es posible si en su lista de posiciones adyacentes no ocupadas hay al menos un elemento, ya que de lo contrario aunque quiera moverse no tiene hacia donde y por tanto no le queda otra alternativa que permanecer en esa posición; y no podría ni generar basuras, ni mover obstáculos.
    - En caso que haya decidido moverse, y le sea posible, se selecciona una posición aleatoria dentro de las cuales puede moverse, y que no haya sido seleccionada anteriormente por ninguno de los niños (o sea si un niño decidió anteriormente que se movería hacia alguna posición, ningún otro niño puede moverse hacia esa ya que coincidirían los dos en la misma posición, en el próximo turno, y en el problema no se contempla esta posibilidad).
    - Se tienen entonces una serie de niños que se mueven y hacia que posiciones lo harán, pero pueden existir conflictos a la hora en que la posición que desean ocupar esté ocupada por un obstáculo porque aunque se consideró que este podía ser movido, no se tiene en cuenta los cambios que se pueden provocar el ambiente por el movimiento de los demás niños.

      Por tanto por cada posición a mover, que esté ocupada por obstáculo, se verifica si este puede realmente ser movido, teniendo en cuenta el ambiente que resulta del movimiento de los demás obstáculos (como los obstáculos se analizan uno por uno en caso de que el movimiento de alguno de estos impida el movimiento de otro posterior, equivale que el niño que ocuparía la posición de ese obstáculo, ya no la podrá ocupar). Los obstáculos que no pueden ser movidos, no pueden entonces ser ocupados por niños, y por tanto los niños no

se pueden mover hacia ellos; en estos casos la simulación va a considerar que esos niños no se moverán en el presente cambio de ambiente (función `moverN`).

- Luego se tiene una serie de niños que se van a mover hacia una posición de sus adyacentes, por cada uno de ellos, y teniendo en cuenta la cantidad de niños de su cuadrilla de **3x3**, que se van a mover, se generan (o no) nuevas suciedades.
- Al culminar este proceso se pueden producir cambios en el ambiente en cuanto a:
    - Las posiciones de los niños ya que algunos de estos se pueden mover.
    - Las posiciones de los obstáculos ya que algunos de estos pueden ser movidos.
    - Las posiciones de las suciedades ya que se pueden generar nuevas suciedades.

- Para los cambios provocados por los agentes (robots) se van a tener en cuenta dos modelos diferentes:
    - **Modelo 1** en el que los robots tendrán ciertas características de los agentes proactivos ya que estará orientado a objetivos: su objetivo fundamental será llevar a los niños a los corrales. Tendrá también ciertos comportamientos asociados a los cambios del ambiente (va a tener ciertas características de los agentes reactivos). Las conductas que asumirán los robots con este modelo serán:
        - Si se encuentra cargando un niño: se moverá en busca de una posición del corral donde pueda dejarlo (una posición del corral que no sea ocupada por otro niño). No le importará las posiciones sucias incluso cuando se encuentre ubicado en ellas, solo estará encaminado a dejar al niño en el corral. Siempre que pueda moverse dos posiciones, lo hará (función `movConNinno1`).
        - Cuando no se encuentre cargando un niño (función `movSinNinno`), va a:
            - Si se encuentra en una posición sucia la limpiará.
            - Sino, buscará si en alguna de sus posiciones adyacentes, de las que se puede mover, se encuentra un niño (fuera del corral). En caso de que exista lo cargará y asumirá el comportamiento anteriormente mencionado de cuando se encuentra cargando un niño.
            - Sino, buscará si en alguna de sus posiciones adyacentes, de las que se puede mover, hay suciedad, y en caso de que así sea, se moverá hacia esa posición para limpiarla.
            - Sino, buscará cuál de sus posiciones adyacentes, de las que se puede mover, se encuentra más próxima de un niño. Se moverá siguiendo esta estrategia en busca del niño más próximo.
            - Si todos los niños ya se encuentran en el corral, el robot pasará a buscar las posiciones sucias, siguiendo la misma estrategia planteada en el punto anterior pero asociada a las posiciones sucias en vez de con niños.

Este modelo aunque su prioridad es ubicar a los niños en el corral, le concede también cierta importancia a la limpieza. Si solo se encaminara a recoger niños, la cantidad de casillas sucias puede ser sumamente grande, pues solo se realizaría la limpieza de las casillas cuando todos los niños estuvieran en el corral, lo cual puede tardar bastante, y en algunos casos, en dependencia de la disposición de los elementos del ambiente, puede llegar a no ocurrir.

- **Modelo 2** en el que los robots se comportarán como agentes reactivos ya que sus decisiones serán en consecuencias con el estado, en cada momento, del ambiente. Las conductas que tomaran los robots con este modelo serán:
    - Cuando no se encuentra cargando a un niño, asumirá el mismo comportamiento planteado anteriormente en el **modelo 1** (función `movSinNinno`). Se emplea el mismo ya que en el anterior modelo, el carácter reactivo se mostraba precisamente en el momento en que no cargaba niño, porque en cuanto lo cargaba, pasaba a ser proactivo. Y este modelo como cambia en dependencia de las condiciones del ambiente, trata en todo momento de ejecutar alguna de sus funciones (dígase cargar niño o limpiar), y en caso que no pueda realizar ninguna, le dará prioridad a la búsqueda de los niños ya que al ser estos los que generan la basura es más sensato tenerlos en los corrales donde no pueden ensuciar.
    - Si se encuentra cargando un niño se moverá siempre que pueda dos posiciones en busca de una posición del corral sin niño, pero si en su trayectoria se encuentra una posición sucia (alguna adyacente sucia), se moverá hacia ella para posteriormente (próximo turno) limpiarla.

      Con este modelo se considera que el robot es más eficiente en comparación al anterior, ya que no deja tan de lado la función de limpiar, por lo que debe ser mayor el porciento de casillas limpias.

Aunque se considere que con el **modelo 2** los robots van a alcanzar mayores porcientos de limpieza (en comparación con el **modelo 1**), puede que esto no siempre se logre; ya que como la generación de suciedades asume un comportamiento aleatorio, y lo mismo con la distribución del ambiente, los resultados no son predecibles, de ahí que se lleve a cabo dichas simulaciones.

Entonces para correr la simulación en `haskell`:

1. Ubicarse en la ruta de la carpeta con los códigos.
2. Correr el comando `ghci`.
3. Correr el comando `:load Ambiente.hs`.

Ya aquí se ofrecen dos métodos para efectuar la simulación:

- `simulacion` que devuelve los porcientos de limpieza, así como estos valores promediados. Los resultados que se mostrarán posteriormente resultarán de llamar a este método.
- `simulacionConAmbiente` que devuelve, además de los valores de `simulacion` (porcientos y promedios), todos los ambientes que se fueron generando a lo largo de la simulación. Como son tantos es sumamente complejo entenderlos ya que son demasiados ambientes, pero este método es sumamente útil pues evidencia todos los cambios que se fueron llevados a cabo durante la simulación.

Ambas funciones reciben los mismos argumentos:

`Int-> Int->Int ->(Int,Int)->Int->Int->Int->Int->String`

`tipo t_g t_cambio (n,m) n_0 r_0 s_0 o_0`

Donde **tipo** es 1 cuando se quiere llevar a cabo la simulación con el Modelo de Agentes 1 y **tipo** es 2 cuando se quiere llevar a cabo la simulación con el Modelo de Agentes 2.

Los restantes valores son las variables del ambiente y de la simulación inicialmente mencionadas.

Para que se impriman correctamente los saltos de líneas en los resultados es conveniente llamar a la simulación con la función `putStrLn`:

- Ejemplo 1 (tipo 1):

```
*Ambiente> putStrLn (simulacion 1 100 5 (10,12) 5 2 5 2)
```

Se obtiene:

```
Cantidad de Iteraciones::100
Tiempo de Cambio de Ambiente::5
Porcientos de limpieza obtenidos::
[95.833336,95.833336,95.833336,96.666664,96.666664,95.833336,95.833336,95.833336
,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.0,95.0
,95.0,95.0,95.0,95.0,95.0,95.0,95.833336,95.833336,95.833336,95.833336,95.833336
,95.833336,95.833336,95.0,95.0,95.0,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336]
Promedio de Porcientos::95.75834
```

- Ejemplo 1 (tipo 2):

```
*Ambiente> putStrLn (simulacion 2 100 5 (10,12) 5 2 5 2)
```

Se obtiene:

```
Cantidad de Iteraciones::100
Tiempo de Cambio de Ambiente::5
Porcientos de limpieza obtenidos::
[95.833336,95.833336,96.666664,96.666664,96.666664,94.166664,94.166664,94.166664
,94.166664,94.166664,93.333336,93.333336,93.333336,93.333336,94.166664,94.166664
,94.166664,95.0,95.0,95.0,94.166664,94.166664,95.0,95.0,95.0,95.0,95.0,95.0,95.0
,95.0,94.166664,95.0,95.0,95.833336,95.833336,95.833336,95.833336,96.666664,96.6
66664,96.666664,96.666664,96.666664,96.666664,96.666664,96.666664,96.666664,96.6
66664,96.666664,96.666664,96.666664,96.666664,97.5,97.5,97.5,97.5,98.333336,98.3
33336,98.333336,99.166664,99.166664,99.166664,99.166664,99.166664,99.166664,99.1
66664,99.166664,99.166664,99.166664,99.166664,99.166664,100.0,100.0,100.0,100.0,
100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,10
0.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0,100.0]
Promedio de Porcientos::97.37498
```

- Ejemplo 2 (tipo 1):

```
*Ambiente> putStrLn (simulacion 1 100 2 (10,12) 5 2 7 2)
```

Se obtiene:

```
Cantidad de Iteraciones::100
Tiempo de Cambio de Ambiente::2
Porcientos de limpieza obtenidos::
[94.166664,94.166664,90.833336,90.833336,90.0,90.0,90.0,90.0,89.166664,89.166664
,88.333336,88.333336,88.333336,88.333336,88.333336,89.166664,88.333336,89.166664
,89.166664,89.166664,88.333336,89.166664,87.5,88.333336,88.333336,89.166664,89.1
66664,89.166664,89.166664,89.166664,89.166664,90.0,89.166664,90.0,90.0,90.0,89.1
66664,89.166664,89.166664,89.166664,88.333336,88.333336,88.333336,88.333336,88.3
33336,88.333336,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5
,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,86.666664,86.6
66664,86.666664,86.666664,86.666664,86.666664,86.666664,86.666664,85.833336,85.8
33336,85.833336,85.833336,85.833336,85.833336,85.833336,85.833336,85.833336,85.8
33336,85.833336,85.833336,85.833336,85.833336,85.833336,85.833336,85.833336,85.8
33336,85.833336,85.833336,85.833336]
Promedio de Porcientos::87.92499
```

- Ejemplo 2 (tipo 2):

```
*Ambiente> putStrLn (simulacion 2 100 2 (10,12) 5 2 7 2)
```

Se obtiene:

```
Cantidad de Iteraciones::100
Tiempo de Cambio de Ambiente::2
Porcientos de limpieza obtenidos::
[94.166664,94.166664,93.333336,93.333336,92.5,92.5,90.833336,90.833336,90.0,90.0
,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,90.0,91.666664
,91.666664,92.5,92.5,93.333336,93.333336,93.333336,93.333336,94.166664,94.166664
,94.166664,95.0,95.0,95.0,95.0,95.0,95.0,95.0,95.0,95.0,95.0,95.0,95.0,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.833336,95.8
33336,95.833336,95.833336,95.833336]
Promedio de Porcientos::94.24169
```

- Ejemplo 3 (tipo 1):

```
*Ambiente> putStrLn (simulacion 1 100 2 (10,12) 10 2 7 2)
```

Se obtiene:

```
Cantidad de Iteraciones::100
Tiempo de Cambio de Ambiente::2
Porcientos de limpieza obtenidos::
[94.166664,94.166664,93.333336,93.333336,92.5,92.5,91.666664,91.666664,90.0,90.0
,88.333336,89.166664,89.166664,90.0,89.166664,89.166664,89.166664,89.166664,89.1
66664,89.166664,87.5,87.5,87.5,87.5,87.5,87.5,87.5,87.5,86.666664,86.666664,85.8
33336,86.666664,85.833336,85.833336,85.833336,85.833336,85.0,85.0,84.166664,84.1
66664,83.333336,83.333336,83.333336,83.333336,83.333336,83.333336,81.666664,81.6
66664,80.833336,80.833336,80.833336,80.833336,80.833336,80.833336,79.166664,79.1
66664,78.333336,78.333336,77.5,77.5,76.666664,76.666664,75.833336,75.833336,75.8
33336,75.833336,75.833336,75.833336,75.833336,75.833336,75.833336,75.833336,75.0
,75.0,75.0,75.0,75.0,75.0,75.0,74.166664,74.166664,74.166664,74.166664,74.1
66664,74.166664,74.166664,74.166664,74.166664,74.166664,74.166664,74.166664,74.1
66664,74.166664,74.166664,74.166664,73.333336,73.333336,73.333336,73.333336]
Promedio de Porcientos::81.524994
```

- Ejemplo 3 (tipo 2):

```
*Ambiente> putStrLn (simulacion 2 100 2 (10,12) 10 2 7 2)
```

Se obtiene:

```
Cantidad de Iteraciones::100
Tiempo de Cambio de Ambiente::2
Porcientos de limpieza obtenidos::
[94.166664,94.166664,92.5,92.5,91.666664,91.666664,89.166664,90.0,89.166664,89.1
66664,87.5,87.5,87.5,88.333336,87.5,87.5,87.5,87.5,85.0,85.0,85.833336,85.833336
,84.166664,84.166664,83.333336,84.166664,84.166664,84.166664,85.0,85.0,85.0,85.0
,84.166664,84.166664,84.166664,84.166664,84.166664,84.166664,84.166664,84.166664
,85.0,85.0,85.833336,85.833336,85.833336,85.833336,85.0,85.0,85.0,85.0,84.166664
,84.166664,83.333336,83.333336,83.333336,83.333336,82.5,82.5,81.666664,81.666664
,81.666664,82.5,80.833336,81.666664,81.666664,81.666664,80.0,80.0,78.333336,78.3
33336,78.333336,78.333336,77.5,77.5,76.666664,76.666664,76.666664,76.666664,75.0
,75.0,74.166664,74.166664,73.333336,73.333336,71.666664,71.666664,70.833336,70.8
33336,70.0,70.0,70.0,70.0,70.0,70.0,70.0,70.0,69.166664,69.166664,69.166664,69.1
66664]
Promedio de Porcientos::81.49167
```

Los resultados obtenidos en estos ejemplos, muestran mejores resultados en el objetivo de los robots haciendo uso del **modelo 2** con respecto al **1**, pero estos resultados pueden variar mucho en dependencia de los datos que se introduzcan y los ambientes y movimientos que resulten.