Fundamentos de programación - Clase 5

Avalancha

Verano 2021

Avalancha

Objetivo de hoy

Queremos hacer un programa que *simule* la dinámica de un proceso de **avalancha**.

Objetivo de hoy

Queremos hacer un programa que *simule* la dinámica de un proceso de **avalancha**.

Un proceso de avalancha tiene las siguientes características:

- Se acumula un elemento *chiquito* en un terreno (en nuestro caso serán copos de nieve).

Objetivo de hoy

Queremos hacer un programa que *simule* la dinámica de un proceso de **avalancha**.

Un proceso de avalancha tiene las siguientes características:

- Se acumula un elemento *chiquito* en un terreno (en nuestro caso serán copos de nieve).
- A medida que cada posición se va llenando, se empieza a **derramar** hacia los costados.

Objetivo de hoy

Queremos hacer un programa que *simule* la dinámica de un proceso de **avalancha**.

Un proceso de avalancha tiene las siguientes características:

- Se acumula un elemento *chiquito* en un terreno (en nuestro caso serán copos de nieve).
- A medida que cada posición se va llenando, se empieza a **derramar** hacia los costados.
- Los vecinos también se van llenando y éstos, a su vez, derraman para sus vecinos.

Objetivo de hoy

Queremos hacer un programa que *simule* la dinámica de un proceso de **avalancha**.

Un proceso de avalancha tiene las siguientes características:

- Se acumula un elemento *chiquito* en un terreno (en nuestro caso serán copos de nieve).
- A medida que cada posición se va llenando, se empieza a **derramar** hacia los costados.
- Los vecinos también se van llenando y éstos, a su vez, derraman para sus vecinos.
- Llega un momento que todo está tan lleno que un último copo de nieve genera que haya un *derrame* que involucra a una gran parte de las posiciones.

El Modelo

Para capturar esta dinámica, vamos a programar lo siguiente:

- El terreno donde cae la nieve lo vamos a modelar con una matriz (como un `array` de `numpy`, pero de dos dimensiones).

El Modelo

Para capturar esta dinámica, vamos a programar lo siguiente:

- El terreno donde cae la nieve lo vamos a modelar con una matriz (como un `array` de `numpy`, pero de dos dimensiones).
- Cada posición de esta matriz va a representar un pequeño sector del terreno.

El Modelo

Para capturar esta dinámica, vamos a programar lo siguiente:

- El terreno donde cae la nieve lo vamos a modelar con una matriz (como un `array` de `numpy`, pero de dos dimensiones).
- Cada posición de esta matriz va a representar un pequeño sector del terreno.
- Cada posición del terreno va a poder soportar una cantidad **fija** máxima de nieve.

El Modelo

Para capturar esta dinámica, vamos a programar lo siguiente:

- El terreno donde cae la nieve lo vamos a modelar con una matriz (como un `array` de `numpy`, pero de dos dimensiones).
- Cada posición de esta matriz va a representar un pequeño sector del terreno.
- Cada posición del terreno va a poder soportar una cantidad **fija** máxima de nieve.
- Cuando una posición tuviera más nieve que la que puede soportar, va a **derramar** hacia todos sus vecinos.

El Modelo

Para capturar esta dinámica, vamos a programar lo siguiente:

- El terreno donde cae la nieve lo vamos a modelar con una matriz (como un `array` de `numpy`, pero de dos dimensiones).
- Cada posición de esta matriz va a representar un pequeño sector del terreno.
- Cada posición del terreno va a poder soportar una cantidad **fija** máxima de nieve.
- Cuando una posición tuviera más nieve que la que puede soportar, va a **derramar** hacia todos sus vecinos.
- Para simplificar, vamos a hacer que caiga nieve en una **única** parte del terreno.

Detalles antes de empezar

- Vamos a trabajar en **dos** dimensiones. Para definir una matriz en numpy, hacemos un array y lo reorganizamos:

```
import numpy as np
n=12 #dimension del tablero donde vamos a trabajar
t = np.repeat(0, n*n) #Creamos un array de la cantidad
t = t.reshape(n, n) #Pasamos de un array a una matriz
```

Detalles antes de empezar

- Vamos a trabajar en **dos** dimensiones. Para definir una matriz en numpy, hacemos un array y lo reorganizamos:

```
import numpy as np
n=12 #dimension del tablero donde vamos a trabajar
t = np.repeat(0, n*n) #Creamos un array de la cantidad
t = t.reshape(n, n) #Pasamos de un array a una matriz
```

Detalles antes de empezar

- Vamos a trabajar en **dos** dimensiones. Para definir una matriz en numpy, hacemos un array y lo reorganizamos:

```
import numpy as np
n=12 #dimension del tablero donde vamos a trabajar
t = np.repeat(0, n*n) #Creamos un array de la cantidad
t = t.reshape(n, n) #Pasamos de un array a una matriz
```

Si imprimimos t, veríamos algo así:

```
[[0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 0 0]]
```

Más detalles

- Para acceder a cada elemento de una matriz, ahora necesitamos su fila y su columna.

Más detalles

- Para acceder a cada elemento de una matriz, ahora necesitamos su fila y su columna.
- En numpy hay una forma de indicar esto. Por ejemplo, para acceder a la tercer fila y cuarta columna, hacemos:

```
t[(2,3)] # Recordar que se numera a partir del 0
```

Más detalles

- Para acceder a cada elemento de una matriz, ahora necesitamos su fila y su columna.
- En numpy hay una forma de indicar esto. Por ejemplo, para acceder a la tercer fila y cuarta columna, hacemos:

```
t[(2,3)] # Recordar que se numera a partir del 0
```

- Lo que va entre paréntesis, `(2,3)`, se llama **tupla** y es *como una lista*, pero **inmutable**.

Más detalles

- Para acceder a cada elemento de una matriz, ahora necesitamos su fila y su columna.
- En numpy hay una forma de indicar esto. Por ejemplo, para acceder a la tercer fila y cuarta columna, hacemos:

```
t[(2,3)] # Recordar que se numera a partir del 0
```

- Lo que va entre paréntesis, `(2,3)`, se llama **tupla** y es *como una lista*, pero **inmutable**.

Más detalles

- Para acceder a cada elemento de una matriz, ahora necesitamos su fila y su columna.
- En numpy hay una forma de indicar esto. Por ejemplo, para acceder a la tercer fila y cuarta columna, hacemos:

```
t[(2,3)] # Recordar que se numera a partir del 0
```

- Lo que va entre paréntesis, `(2,3)`, se llama **tupla** y es *como una lista*, pero **inmutable**.
- Para representar posiciones vamos a usar una tupla: (mi_fila, mi_col)

Más detalles

- Para acceder a cada elemento de una matriz, ahora necesitamos su fila y su columna.
- En numpy hay una forma de indicar esto. Por ejemplo, para acceder a la tercer fila y cuarta columna, hacemos:

```
t[(2,3)] # Recordar que se numera a partir del 0
```

- Lo que va entre paréntesis, `(2,3)`, se llama **tupla** y es *como una lista*, pero **inmutable**.
- Para representar posiciones vamos a usar una tupla: (mi_fila, mi_col)
- Para acceder al tablero vamos a hacer: tablero[(mi_fila, mi_col)]

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.
- Vamos a hacer que la nieve caiga en una única posición del tablero y que caiga "una" por turno.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.
- Vamos a hacer que la nieve caiga en una única posición del tablero y que caiga "una" por turno.
- Si se supera la capacidad de una posición del terreno (es decir, si tiene cuatro copos o más), se derrama uno hacia cada uno de sus vecinos.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.
- Vamos a hacer que la nieve caiga en una única posición del tablero y que caiga "una" por turno.
- Si se supera la capacidad de una posición del terreno (es decir, si tiene cuatro copos o más), se derrama uno hacia cada uno de sus vecinos.
- Vamos a decir que las posiciones internas del tablero son las **posiciones utilizables**.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.
- Vamos a hacer que la nieve caiga en una única posición del tablero y que caiga "una" por turno.
- Si se supera la capacidad de una posición del terreno (es decir, si tiene cuatro copos o más), se derrama uno hacia cada uno de sus vecinos.
- Vamos a decir que las posiciones internas del tablero son las **posiciones utilizables**.
- Si al derramar hacia los costados, hubiera un vecino que es un borde, la nieve que cae sobre ese borde se pierde.

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.
- Vamos a hacer que la nieve caiga en una única posición del tablero y que caiga "una" por turno.
- Si se supera la capacidad de una posición del terreno (es decir, si tiene cuatro copos o más), se derrama uno hacia cada uno de sus vecinos.
- Vamos a decir que las posiciones internas del tablero son las **posiciones utilizables**.
- Si al derramar hacia los costados, hubiera un vecino que es un borde, la nieve que cae sobre ese borde se pierde.
- Es posible que un derrame en una posición produzca derrames en otros vecinos (el fenómeno de **avalancha**).

Mecánica de la avalancha

- Vamos a usar una matriz para representar el terreno donde cae nieve.
- Vamos a marcar los bordes con el valor -1. En los bordes no se acumula nieve ni se desborda.
- Las posiciones del terreno tienen una **capacidad** para soportar nieve. En nuestro modelo, es cuatro.
- Vamos a hacer que la nieve caiga en una única posición del tablero y que caiga "una" por turno.
- Si se supera la capacidad de una posición del terreno (es decir, si tiene cuatro copos o más), se derrama uno hacia cada uno de sus vecinos.
- Vamos a decir que las posiciones internas del tablero son las **posiciones utilizables**.
- Si al derramar hacia los costados, hubiera un vecino que es un borde, la nieve que cae sobre ese borde se pierde.
- Es posible que un derrame en una posición produzca derrames en otros vecinos (el fenómeno de **avalancha**).
- Procesar todos los derrames hasta que se estabilice (no haya ocurrido ningún otro derrame) y recién ahí habremos terminado de procesar ese turno.