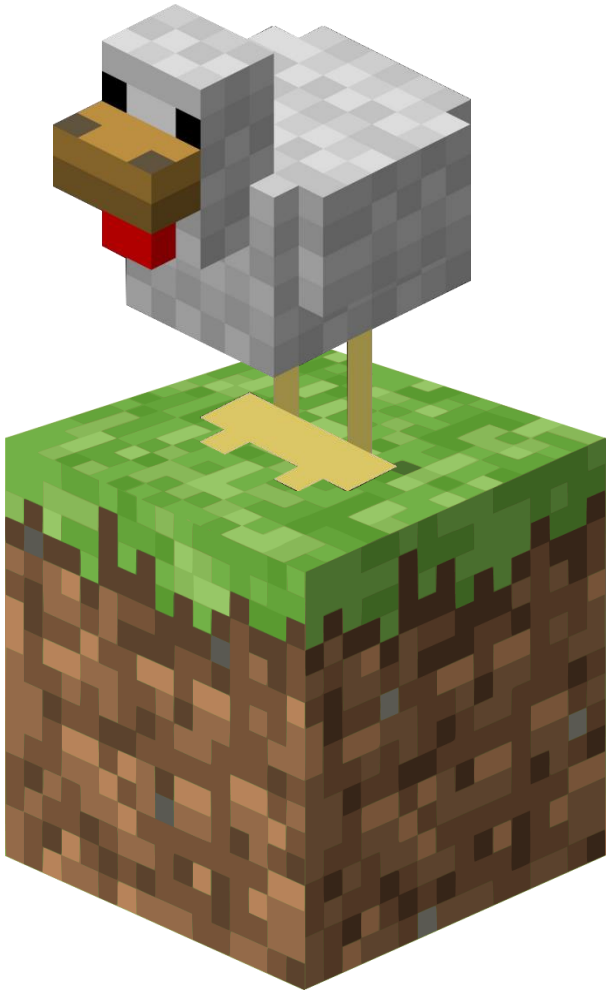
MINEROY



Antonio Hermosin Garcia

Índice

1.- Explicación de las clases

El proyecto consiste en 12 clases, que se dividen en:

-1 clase abstracta Bloque

-8 clases Bloque___ (Incluyendo la clase BloqueJugador, que se usa cuando se muestra el mapa)

-1 clase Jugador

-1 clase JuegoException

-1 clase Juego (main)

Además tiene 3 interfaces, que indican de que tipo son los bloques.

La clase abstracta Bloque tiene 6 constantes(1 por cada tipo de bloque posible, sin BloqueVacio), 3 atributos que son las coordenadas (x, y, z), los métodos get, set y toString, un metodo para hacer desaparecer el bloque(cambiar las coordenadas a -1) y dos métodos abstractos que deben incluir las clases hijas

Cada una de las clases Bloque___ hereda de la clase abstracta Bloque e implementa una de las interfaces. El método destruir mira si la herramienta que usas para destruirlo es la correcta y suma la materia al jugador si lo es; e independientemente de la herramienta, cambia las coordenadas del bloque. El método toString de cada clase hija está sobrescrito para que devuelva una String que identifica el bloque (“Alb” en caso del albero, “Hie” en caso del hierro…). Estas clases hijas

también tienen un método getTipo() que devuelve el tipo de bloque que es (Los tipos son las constantes de la clase Bloque.

2.- Clase Jugador

La clase Jugador tiene una constante que guarda el uso de las herramientas por defecto (en este caso, 5)

Además, consta de varios atributos:

-String nombre, que guarda el nombre del personaje.

-int x, y, z. Son las coordenadas del jugador, su posición en el mapa

-Array de int. Guarda la cantidad de cada una de las materias que tiene el jugador, algo así como el inventario

-int usosHacha, usosPico, usosPala. Controla los usos de cada herramienta, y se pone por defecto cuando se construye el personaje.

Sus métodos son:

-toString. Es el estado del personaje, muestra las coordenadas y la cantidad de materias que tiene de cada tipo

-sumaMateria. Suma una materia del tipo dado al array.

-restaMateria. Resta una materia del tipo dado al array.

-Getters y Setters de los atributos.

3.- Clase Juego

Esta clase tiene dos constantes, una que indica el numero de materias que se necesitan para ganar y otra que indica el tamaño de un lado del mundo (es un cubo, tiene la misma medida en las tres dimensiones)

3.1.- Método main

El main empieza definiendo variables que se usarán mas adelante.

Después, crea un array tridimensional con los datos del tamaño del mundo. Posteriormente lo recorre con un triple for anidado y llama al método generarBloqueAleatorio para llenarlo de bloques.

Lo siguiente es crear el jugador en el primer BloqueVacio, para ello se recorre otra vez el array, esta vez añadiéndole una condición booleana para que pare. Cuando encuentra un BloqueVacio, crea el Jugador con esas coordenadas.

Luego se encuentra el bucle principal, que se repite ahsta que la variable haGanado sea true. Este bucle llama al método menú y al método comprobarSiGana.

Por último imprime el mensaje de victoria del jugador

3.2.- Método generarBloqueAleatorio

Este método se divide en dos partes:

-La primera, cuando la coordenada Z es 0 o 1, no se generan BloqueVacio, solo materias sólidas.

-La segunda, cuando la altura vale más de 1, el random se multiplica por 3, para que los casos en los que se generan BloqueVacio se aumenten y se cumpla que al menos la mitad de los bloques sean BloqueVacio.

3.3.- Método menú

Este método imprime el menú y pide una opción por teclado, con la cual llama al método tratarMenu

3.4.- Método tratarMenu

Consiste en un switch que trata las opciones del menú:

-Opción 1: Mover. Pregunta en que dirección se quiere mover el personaje, guarda la opción y llama al método moverJugador.

-Opción 2: Crear herramienta. Pregunta por la herramienta que se quiere crear y llama al método crearHerramienta.

-Opción 3: Estado. Imprime el método toString del jugador.

-Opción 4: Poner Bloques(Opción inventada). Pregunta la dirección y llama al método ponerBloque con esa dirección.

-Opción 5: Mostrar mapa (Opción inventada).

Llama al método MostrarMapaCompleto.

3.5.- Método mover

Consta de un switch que mira la opción elegida en el método tratarMenu.

Cada una de las opciones es diferente, pero esencialmente iguales, solo cambia que, en vez de sumar 1 a la x, se le resta uno, o se suma y resta otra coordenada. Por ello, voy a describir una parte únicamente.

Hay un if que mira si el bloque la coordenada a la que te quieres mover se sale del mundo, en ese caso, la coordenada a la que se movería seria el otro lado, saliendo del array también (-1 o TAMANNO_MUNDO, según corresponda)

Después mueve al jugador, con el set correspondiente.

El siguiente if comprueba que el bloque que se rompe no es un bloqueVacio o null, en caso de entrar en el if, se llama al método destruir y se cambia ese bloque a BloqueVacio. EL método destruir llama a su vez al método elegirHerramienta.

Independientemente del anterior if, se comprueba si la posición del jugador se sale del array, en cuyo caso se pone la coordenada a 0 o TAMANNO_MUNDO, según corresponda.

3.6.- Método elegirHerramienta

Pregunta que herramienta se elige, la devuelve para que al pueda usar el método destruir, y resta un uso a esa herramienta.

3.7.- Método crearHerramienta

Dependiendo de la opción que se elija, resta una cantidad de materias diferente, dependiendo del coste de crear esa herramienta, y pone el uso de la herramienta elegida a 5.

3.8.- Método mostrarMapa (Opción secreta)

Recorre el array con los tres for anidados e imprime los toString del bloque correspondiente con un formato agradable a la vista.

3.9.- Método comprobarSiGana

Comprueba si gana, mirando si en cada materia del jugador hay 7 o más materias, y devuelve si gana o no, en forma de booleano.

3.10.- Método ponerBloque (Opción Inventada)

Este método se parece al método mover, con la diferencia de que no pone el bloque por el otro lado si se sale del array.

Se crea un bloque aleatorio, apyandose en el método generarBloqueAleatorio y en cada una de las direcciones comprueba si no es un bloqueVacio. En caso de no serlo, se intenta destruir.

Independientemente de si es BloqueVacio o no, el bloque en esas coordenadas se cambia por el bloque aleatorio.

Finalmente, se le resta una materia de ese tipo al jugador