**Project Template**

**Project Report: Group Number**

Friday, October 20, 2023

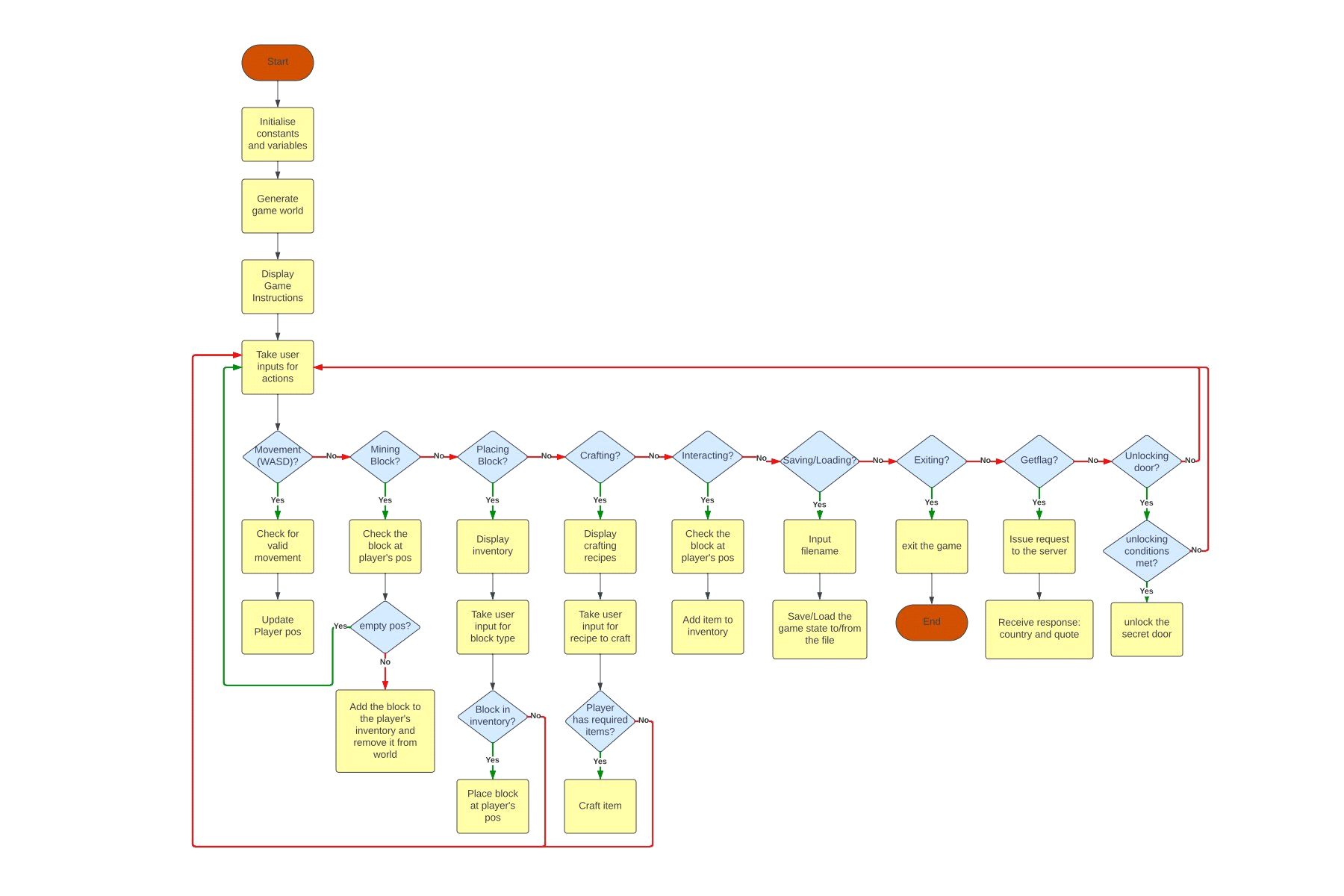
|  |  |
| --- | --- |
| Attribute | Details |
| Group Name | Group33 |
| Group Number | 33 |
| TA | Sam Goldie |

|  |  |
| --- | --- |
| Student Name | Student ID |
| Abdul Hadi Ayoub | 6341466 |
| Mohamad Yahia Lababidi | 6366266 |
| Magdy | 6362437 |
| [Name] | [Student ID] |

**Introduction**

This report is for JavaCraft project. Describing what we did, how we did it and what we used for it.

**JavaCraft’s Workflow**

* Flowchart For Game: 
* Pseudocode For Game:

//! This is a text-based game inspired by Minecraft.

// This is the main() method where the game starts

initialize the game world with the width and height.

generate the game world and place blocks randomly.

display game instructions for the players.

IF the user wants to play:

clear the screen and display the legend, world and inventory.

take the user command.

IF user's command is to move:

IF unlock mode is enabled:

THEN, flag that the movement command has been entered.

END

move the player position if the position is within the game borders.

END

ELIF user's command is to mine:

IF unlock mode is enabled:

THEN, flag that the mine command has been entered.

END

mine the current block if it is not empty (air) by adding it to the inventory and replace it with air

END

ELIF user's command is to place:

THEN, display the inventory for the user.

prompt the user for input to specify the block type or crafted item to place.

IF a valid block type or crafted item number was entered:

THEN, place that block or crafted item when available he wants to be placed in his current position.

END

ELSE:

THEN, notify the user that he entered an invalid type.

END

END

ELIF user's command is to craft:

THEN, display the recipes for the user to know what to choose.

prompt the user for input to get the recipe number.

IF the user has enough blocks for the recipe he wants to craft:

THEN, craft it

END

ELSE:

Tell him he does not have enough resources to craft.

END

END

ELIF user's command is to interact:

THEN, interact with the block where he stands and add it to the inventory and replace it with air in the game world.

END

ELIF user's command is to save:

THEN, prompt the user for the file name.

save the game with the file name the user provided.

When some error happens, i.e. the file does not exist, tell the user that!

END

ELIF user's command is to load:

THEN, prompt the user for the file name.

load the game from the file the user provided.

When some error happens, i.e. the file does not exist, tell the user that!

END

ELIF user's command is to look:

THEN, look around the player's position and display the surrounding blocks.

END

ELIF user's command is to unlock:

THEN, enable the unlock mode.

END

ELIF user's command is to get flag:

THEN, get a flag along with a quote from the provided API.

END

ELIF user's command is to open:

IF unlock mode is enabled and the user has moved and mined a block and crafted:

THEN, flag that the secret door is been unlocked.

reset the world to generate the world with the specified flag.

END

ELSE // the user has invalid passkey.

THEN, disable the unlock mode.

flag that the open command, movement command, mining command and crafting command have not been entered.

END

ELSE // invalid input

notify the user that he entered an invalid input.

IF unlock mode is enabled:

IF user's command is to craft:

THEN, flag that the user has crafted.

END

ELIF user's command is to mine:

THEN, flag the the user has mined.

END

ELIF user's command is to open:

THEN, flag the user has used open command.

END

IF the secret door is unlocked:

THEN, clear the screen.

flag that the user now is in the secret area (flag).

reset the world to generate the new world with the specified flag.

reset the secret door unlocked mode.

fill the inventory with each block.

END

END

ELSE:

Say Goodbye! and do not start the game.

END

**Functionality Exploration**

List of key functionalities explored:

|  |  |  |
| --- | --- | --- |
| No. | Function Name | Description |
| 1 | startGame() | It is the main method in the game, it runs the game and contains its logic. |
| 2 | resetWorld() | It is where the **genereateEmptyWorld()** is called to plot the flag to the game when the secret door is opened and it resets the player’s position to the middle of the world. |
| 3 | saveGame() | It saves the whole game along with the crafted items and inventory to a file named by the player. |
| 4 | loadGame() | It loads a game from a file. |

Note: Provide flowchart and pseudocode for at least 15 functions in the Appendix.

**Finite State Automata (FSA) Design**

* Secret Door Logic Analysis:

Well, first things first, `unlock` should be entered, then we should move and mine and craft then we can use the open command to open the secret door. If we did everything without entering the command `unlock`, then we have an invalid passkey. So, we should first input `unlock` then we can move which we have to do if we want to mine which we have to do also if we want to craft something, BUT this is not the case because in the code it does not matter if we actually crafted something, it only cares if we just typed `c` which is not fun (That’s why I fixed that).

`unlock` -> {`wsad`, `m`, `c`} -> open -> secret door is open!

* FSA Illustration & Description: [Attach FSA diagram]

**Git Collaboration & Version Control**

* Repository Link: <https://gitlab.maastrichtuniversity.nl/bcs1110/javacraft/-/tree/Group33?ref_type=heads>
* Branch Details: List branch names and corresponding members
* Changes & Conflicts: When we are done with something, we push it immediately to the repo. There was one time where a conflict occurred where I was editing on the code so was someone else, so I had to `pull` the changes first, then I made my changes and `add` the file to the staging area, then commit it and finally push it to the repo.

**Extending the Game Code (For Final Submission)**

Two new blocks:

* Diamond ore block
* Dragon egg

One crafting recipe:

* Diamond Sword.

private static final int DIAMOND\_ORE = 5;

private static final int DRAGON\_EGGS = 6;

private static final int CRAFTED\_DIAMOND\_SWORD = 203;

public static void generateWorld() {

        Random rand = new Random();

        int cnt = 0; // to make sure we have only one dragon egg

        for (int y = 0; y < world[0].length; y++) {

            for (int x = 0; x < world.length; x++) {

                int randValue = rand.nextInt(100);

                if (randValue < 20) {

                    world[x][y] = WOOD;

                } else if (randValue < 35) {

                    world[x][y] = LEAVES;

                } else if (randValue < 50) {

                    world[x][y] = STONE;

                } else if (randValue < 70) {

                    world[x][y] = IRON\_ORE;

                } else if (randValue < 72) {

                    world[x][y] = DIAMOND\_ORE;

                } else if (cnt < 1 && randValue < 80) {

                    world[x][y] = DRAGON\_EGGS;

                    cnt++;

                } else {

                    world[x][y] = AIR;

                }

            }

        }

    }

public static void craftDiamondSword() {

        if (inventoryContains(DIAMOND\_ORE, 2) && CraftedItemsContains(CRAFTED\_STICK, 1)) {

            removeItemsFromInventory(DIAMOND\_ORE, 2);

            removeItemsFromCraftedItems(CRAFTED\_STICK, 1);

            addCraftedItem(CRAFTED\_DIAMOND\_SWORD);

            System.out.println("Crafted DEADLY DIAMOND SWORD!!");

        } else {

            System.out.println("Insufficient resources to craft diamond sword.");

        }

    }

**Interacting with Flags API (For Final Submission)**

We got the Canadian flag which was hard to render to game world.

So, plotting the left and right red stripes was easy, just using loops and it is done.

I divided the world into parts so that I can take the left and right part for the red stripes and leave the middle one for the maple leaf (which was the hardest to plot).

However, I made a String array and save each line of the maple leaf on a separate string, then I have 15 strings which is almost equal to game height (I increased it by two so I can get 1 line margin from top and bottom). Then I made an array of integers with the same size as the String array containing the number of spaces should be before drawing each line of the maple leaf. Then, I looped over each string of the string array, and I added a red block to the world with y-coordinates equal to the current integer in the integers array and x-coordinates equal to the current line (string from the String array) plus 1 (to skip plotting in the first line and keep margin by one line). Finally, I plot everything that is not red with white using nested loops!

**Conclusion (For Final Submission)**

We have faced many problems during the project especially using `git`. However, we overcame it, we learned how to use `git`, how to design flowcharts, FSA and pseudocodes. As for the coding, we learned how to manipulate texts in command prompt through Java, how to use an API, making basic internet requests using Java, and drawing something (flags!) using Java into the terminal! We have explored many colours (both foreground and background) and other ANSI codes for special blocks and shapes.

**Appendix**

//! fillInventory() - This will fill the inventory with each block type

FOR each block type from 1 to 6 (included):

LOOP from 0 till the inventory size

add the current block type to inventory.

END

END

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//! mineBlock() - This will mine the current block and add it to the inventory

IF the player stand on a block that is not AIR:

THEN, add the block to the inventory and replace it with air.

END

ELSE

Then, there is no block to mine here.

END

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//! generateEmptyWorld() - This will create a new world for the flag when the secret door is opened

DEFINE stripHeight as worldHeight / 3

// Fill the top of world with red blocks

LOOP from 0 to stripHeight:

THEN, LOOP from 0 to NEW\_WORLD\_WIDTH:

THEN, put a red block at this coordinates.

END

END

// Fill the middle of world with white blocks

LOOP from stripHeight to stripHeight \* 2:

THEN, LOOP from 0 to NEW\_WORLD\_WIDTH:

THEN, put a white block at this coordinates.

END

END

// Fill the bottom of the world with blue block

LOOP from stripHeight \* 2 to NEW\_WORLD\_HEIGHT:

THEN, LOOP from 0 to NEW\_WORLD\_WIDTH:

THEN, put a blue block at this coordinates.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

//! startGame() - This will starts the game and keeps it running until the player exit

WHILE as long as the player did not exit:

clear the screen and display the legend, world and inventory.

IF user's command is to move:

IF unlock mode is enabled:

THEN, flag that the movement command has been entered.

END

move the player position.

END

ELIF user's command is to mine:

IF unlock mode is enabled:

THEN, flag that the mine command has been entered.

END

mine the current block.

END

ELIF user's command is to place a block:

THEN, display the inventory for the user.

prompt the user for input to specify the block type to place.

place that block type he wants to be placed in his current position.

END

ELIF user's command is to craft:

THEN, display the recipes for the user to know what to choose.

prompt the user for input to get the recipe number.

craft the recipe he wants

END

ELIF user's command is to interact:

THEN, interact with the block where he stands.

END

ELIF user's command is to save:

THEN, prompt the user for the file name.

save the game with the file name the user provided.

END

ELIF user's command is to load:

THEN, prompt the user for the file name.

load the game from the file the user provided.

END

ELIF user's command is to look:

THEN, look around the player's position and print him the surrounding blocks.

END

ELIF user's command is to unlock:

THEN, enable the unlock mode.

END

ELIF user's command is to get flag:

THEN, get a flag along with a quote from the provided API.

END

ELIF user's command is to open:

IF unlock mode is enabled and the user has moved and mined a block and crafted:

THEN, flag that the secret door is been unlocked.

reset the world to generate the world with the specified flag.

END

ELSE // the user has invalid passkey.

THEN, disable the unlock mode.

flag that the open command, movement command, mining command and crafting command have not been entered.

END

ELSE // invalid input

notify the user that he entered an invalid input.

IF unlock mode is enabled:

IF user's command is to craft:

THEN, flag that the user has crafted.

END

ELIF user's command is to mine:

THEN, flag the the user has mined.

END

ELIF user's command is to open:

THEN, flag the user has used open command.

END

IF the secret door is unlocked:

THEN, clear the screen.

flag that the user now is in the secret area (flag).

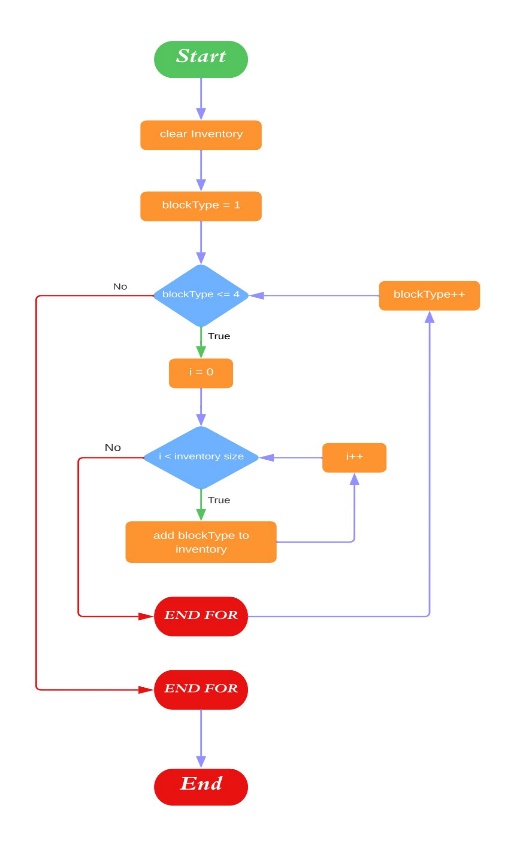
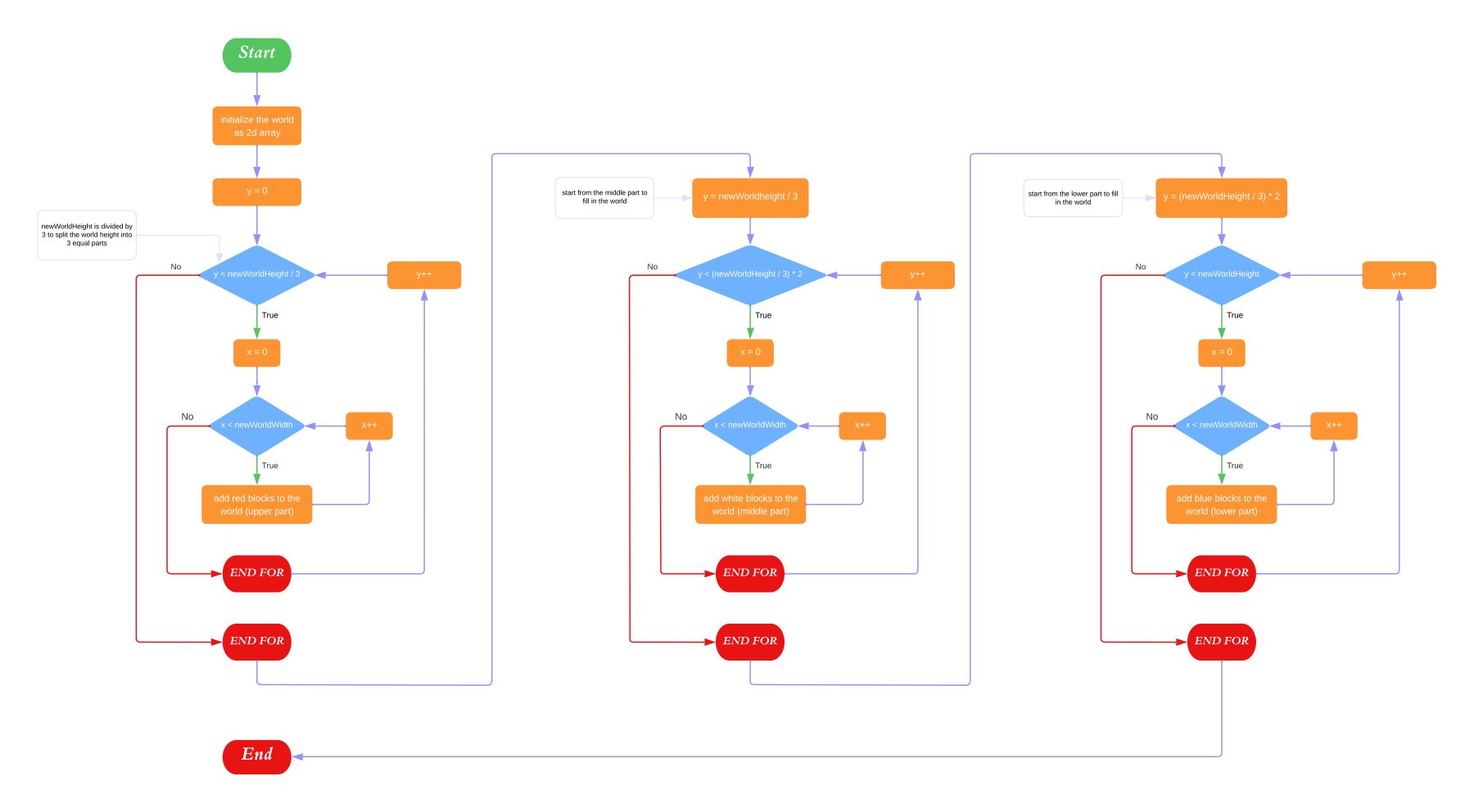
reset the world to generate the new world with the specified flag.

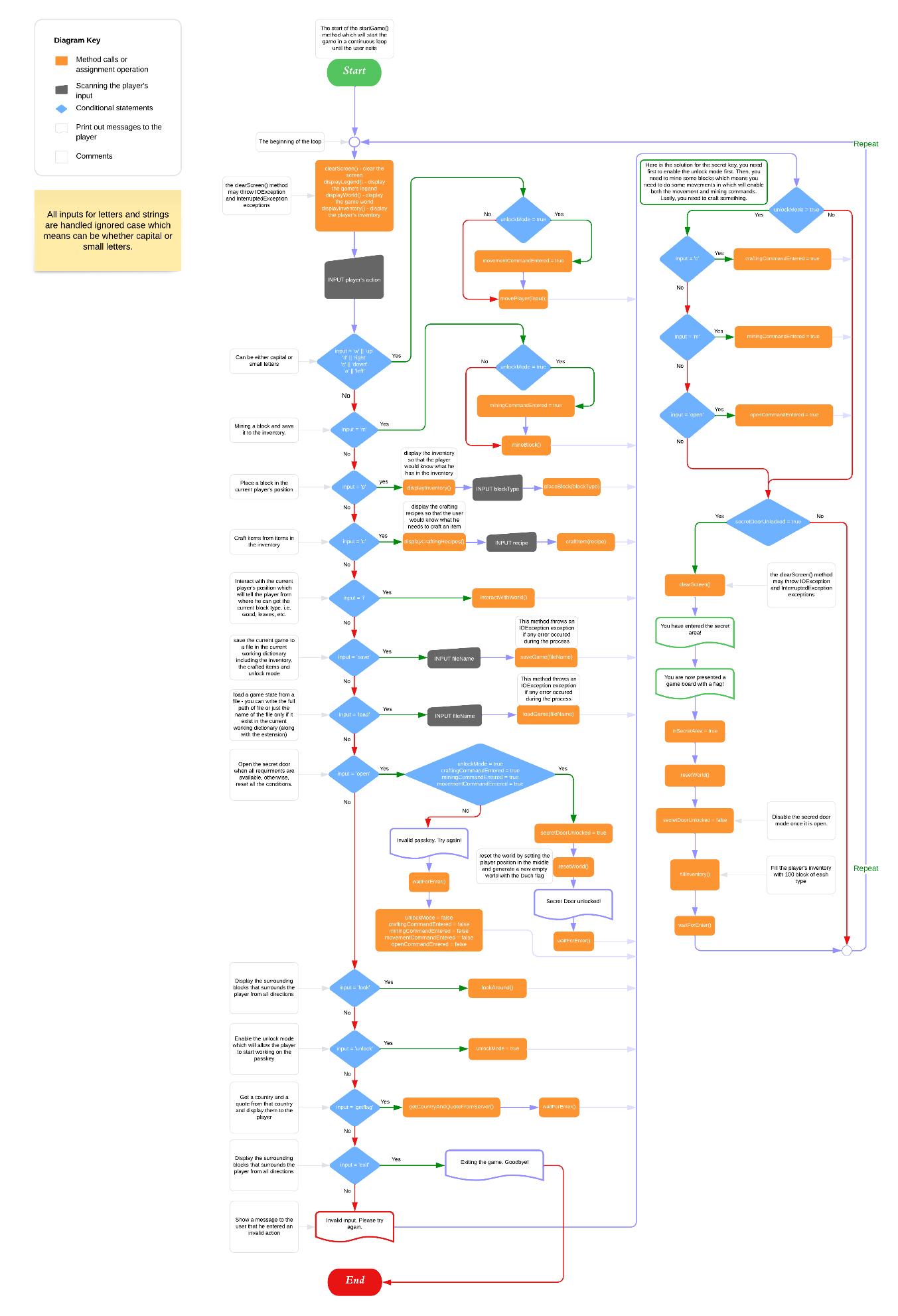
reset the secret door unlocked mode.

fill the inventory with each block.

END

END

A diagram of a flowchart

Description automatically generated

**References**

* ChatGPT – I used it to get ANSI codes for background colours and to solve the encoding problem.