JavaCraft

Provisional Report: Group 43

Saturday, October 7, 2023

Table of contents

Attribute	Details
Group Name	Syntax Error
Group Number	43
TA	José Manuel Ros

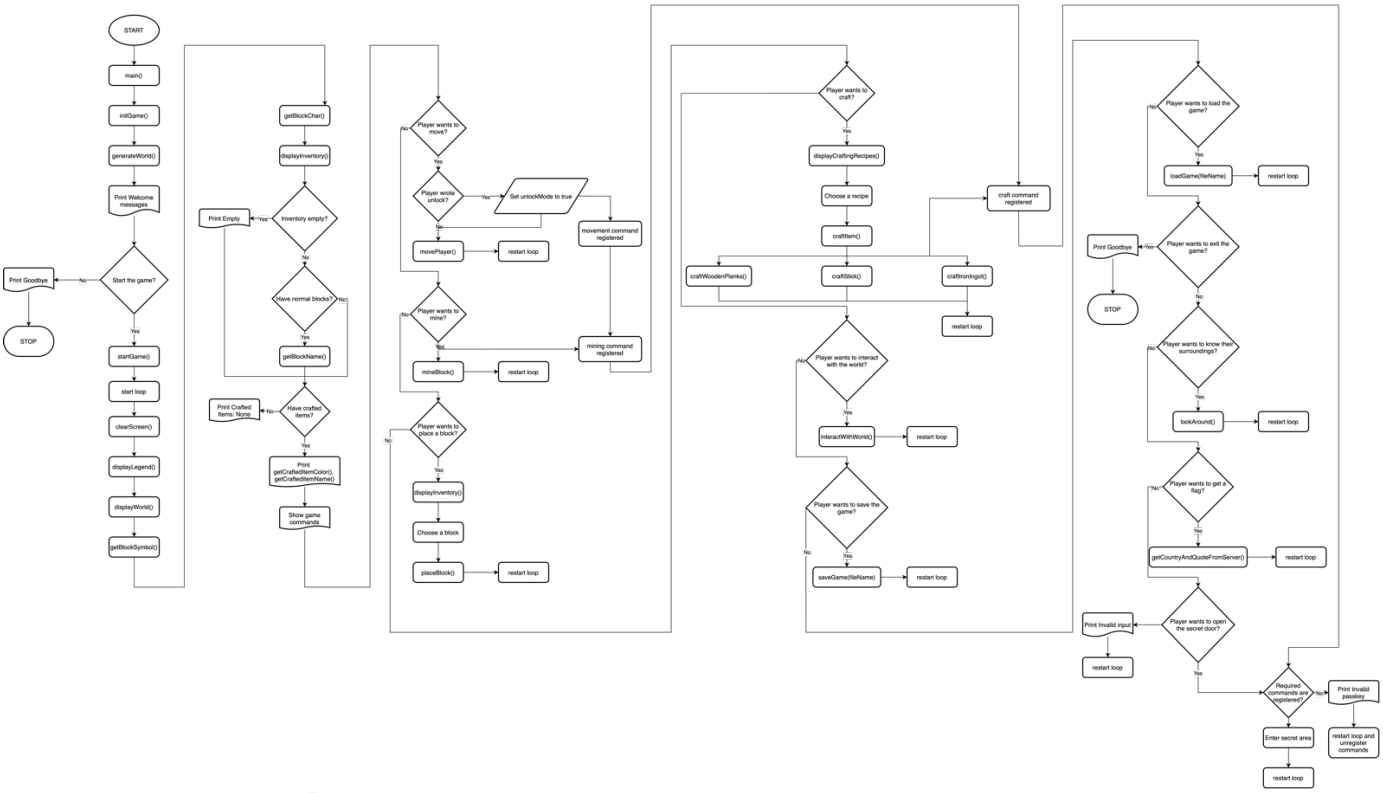
Student Name	Student ID
Botond Moksony	[I6365627]
Vicente Muñoz	[I6346464]
Zhili Yang Wu	[I6362675]
Malo Coquin	[I6302990]

1 Introduction

Name	Tasks
Vicente Muñoz	- Function definitions (25%) - Flowcharts (20%) - Pseudocodes (20%) - Game flowchart (100%) - FSA illustration & design (33%) - Git student (40%) - Git trouble summary (100%)
Botond Moksony	- Function definitions (25%) - Flowcharts (30%) - Pseudocode (30%) - Appendix (100%) - FSA illustration & design (33%) - Git teacher (100%) - Report final revision (100%)
Zhili Yang Wu	- Function definitions (25%) - Flowcharts (40%) (Did startGame flowchart) - Pseudocodes (50%) - Game pseudocode (30%) - FSA description (100%) - FSA illustration & design (13%) - Git student (40%)
Malo Coquin	- Function definitions (25%) - Flowcharts (10%) - Game pseudocode (70%) - FSA illustration & design (20%) - SourceTree student (20%)

2 JavaCraft's Workflow

- Flowchart For Game (also included in the appendix in higher resolution):



Pseudocode For Game:

```
create JavaCraft class
        create  main function
                call initGame function
                call generateWorld function
                print the instructions of the game with the commands
                ask to the player if he want to start the game
                if yes
                        call startgame function
                else
                        print: game not started. Goodbye!

        create initGame function
                set the world size
                set the initial player's position
                create inventory

        create generateWorld function
                create a random world


        create displayWorld function
                print the world map

        create getBlockSymbol function
                return the color of the block and call getBlockChar for the same block

        create getBlockChar function
                return the symbol of the block

        create startGame function
                call clearScreen function
                call displayLegend function
                call displayWorld function
                call displayInventory function
                print the instructions
                do what the player's action

                if the player move
                        call movePlayer function
                else if the player mine
                        call mineBlock function
                else if the player place a block
                        ask witch block he want to place
                        call placeBlock function
                else if the player craft
                        ask witch item he want to craft
                        call craftItem function
                else if the player interact with the world
                        call interactWithWorld function
                else if the player save
                        ask for the file name to save the game
                        call saveGame function
                else if the player load
                        ask witch load he want to load
                        call loadGame function
                else if the player exit
                        print: Exiting the game. Goodbye!
```

```
                    stop the game
            else if the player look
                    call lookAround function
            else if the player unlock
                    set unlockMode to true
            else if the player ask getflag
                    call getCountryAndQuoteFromServer function
                    call waitForEnter function
            else if the player open
                    if all the conditions are true
                            call resetWorld function
                            print: Secret door unlocked!
                            call waitForEnter function
                    else
                            print: Invalid passkey, Try again!
                            call waitForEnter function
                            set all the conditions to false
            else
                    print: Invalid input, Please try again.

            if the first condition is true
                    check if the others conditions are true too
            if secretDoorUnlocked is true
                    call clearScreen function
                    print the context
                    set inSecretArea to true
                    call resetWorld function
                    set secretDoorUnlocked to false
                    call fillInventory function
                    call waitForEnter function

create fillInventory function
            clear the inventory
            add blockType in the inventory

create resetWorld function
            call generateEmptyWorld function
            set the player's initial position

create generateEmptyWorld function
            create a new world
            divide in tree strip the world from the top to the bot
            fill each strip with a different color, to make the dutch flag

create clearScreen function
            clear the screen'

create lookAround function
            print the context
            print the blocks next to the player
            call waitForEnter function

create movePlayer function
            move the player the way he ask

create mineBlock function
            if the block on the player's location is a air block
                    print: No block to mine here.
            if not
                    add the block to his inventory
                    replace the original block by an air block
                    print: "Mined " and call getBlockName function with the blockType in parameter
            call waitForEnter function

create placeBlock function
            check if the inventory contain the block asked
            if yes
                    remove the block from the inventory
                    place the block
                    print: "Placed " and call getBlockName function and print: " at your position."
            else
                    print that the player don't have the block by using getBlockName function
            if the block that the player want to place doesn't exist
                    print: Invalid block number. Please enter a valid block number.
                    print the block's numbers infos
            call waitForEnter function

create getBlockTypeFromCraftedItem function
            return the number of the block asked

create getCraftedItemFromBlockType function
            return the name of the item ask

create displayCraftingRecipes function
            print the crafting recipes

create craftItem function
            if the player choose recipes 1
                    call craftWoodenPlanks function
            if the player choose recipes 2
                    call craftStick function
            if the player choose recipes 3
                    call craftIronIngot function
            else
                    print: Invalid recipes number.
            call waitForEnter function

create craftWoodenPlanks function
            call inventoryContains function with for parameters the recipes of the item
            if the inventoryContain function return true
                    call removeItemsFromInventory function with the recipes of the item
                    call addCraftedItem function with for parameter the item just crafted
                    print: Crafted Wooden Planks.
            else
                    print insufficient resources to craft wooden Planks.

create craftStick function
            call inventoryContains function with for parameters the recipes of the item
            if the inventoryContain function return true
                    call removeItemsFromInventory function with the recipes of the item
                    call addCraftedItem function with for parameter the item just crafted
                    print: Crafted Stick.
            else
                    print insufficient resources to craft Stick.

create craftIronIngot function
            call inventoryContains function with for parameters the recipes of the item
            if the inventoryContain function return true
                    call removeItemsFromInventory function with the recipes of the item
                    call addCraftedItem function with for parameter the item just crafted
                    print: Crafted Iron Ingot.
            else
                    print insufficient resources to craft Iron Ingot.

create inventoryContains function
```

```
        return
                true if inventory contains items
                false if inventory does not contain items

create inventoryContains function for adding count in inventory
        itemCount is equal to 0
        for all i within inventory
                if i is equal to item
                        increment itemCount
                        if itemCount is equal to count
                                inventoryContains is true

create removeItemsFromInventory function
        removedCount is equal to 0
        iterator is equal to inventory with the iteration function called into it
        while iterator has a token
                i is equal to iterator with next method called into it
                if i is equal to item
                        remove item from inventory by iterator
                        increments for removedCount
                if removedCount is equal to count
                        break loop

create addCraftedItem function
        if craftedItems is empty
                create an array list for craftedItems
        add craftedItem to craftedItem

create interactWithWorld function
        switch for blockType
                case wood
                        print where you obtained the wood
                        add wood block to inventory
                case leaves
                        print where you obtained the leaves
                        add leaves block to inventory
                case stone
                        print where you obtained the stone
                        add stone block to inventory
                case iron ore
                        print where you obtained the iron ore
                        add iron ore to inventory
                case air
                        print “Nothing to interact with here”
                default
                        print “Unrecognized block. Cannot interact here”
        call waitForEnter function to wait for user confirmation

create saveGame function
        takes a string of the name of a file as a parameter from which to save the game
        serializes the game state data for saving
        prints where the game state was saved into
        prints whether if there is an error while saving the game
        calls waitForEnter function to ask for user confirmation

create loadGame function
        takes a string of the name of a file as a parameter to load the game from a file
        obtains the game state data from the file and loads the game
        prints the origin of the file from which the game was loaded from
        prints whether if there is an error when loading the game
        calls waitForEnter function to ask for user confirmation

create getBlockName function
        assigns to which type of block does a block belong to
        prints the name for the block

create displayLegend function
        prints out the legend for all the symbols that appear in the game

create displayInventory function
        informs the user about the contents of the inventory
        informs the user about the items that have been crafted by the user

create getBlockColor function
        assigns a color for each type of block that exists in the game

create waitForEnter function
        prints in the terminal “Press enter to continue..”
        scan the enter input of user

create getCraftedItemName function
        prints the name of the items that can be crafted into the terminal

create getCraftedItemColor function
        get the color for the items that can be crafted

create getCountryAndQuoteFromServer function
        try the following for errors
                create a variable that is an URL: “https://flag.ashish.nl/get_flag
                create conn variable responsible for opening the connection to URL
                set the request method “POST”
                set request property with key: “Content-Type”, value: “application/json
                set do output to post requests
                string payload
                call new outputStreamWriter on conn to convert string streams to byte streams
                write payload into the stream
                flush payload into the output stream
                close stream
                create variable reader that reads text for the connection conn to URL
                create a modifiable string called sb
                create a string called line
                while line is equal to line of text from console and it is not empty
                        append sb to line
                string json is set equal to sb converted to String
                countryStart is equal to the first occurrence of “ ” in json plus 11
                countryEnd is equal to the first occurrence of “ ” starting from countryStart
                country is equal to the substring of json from position countryStart to countryEnd
                quoteStart is equal to the first occurrence of “ ” in json plus 9
                quoteEnd is equal to the first occurrence of “ ” in json starting from quoteStart
                quote is equal to the substring of json from quoteStart to quoteEnd
                replace “ ” in quote with “ ”
                print “ ” and country
                print “ ” and quote
        if there is an error
                print where in the code the error occurred
                Print “Error connecting to the server”
```

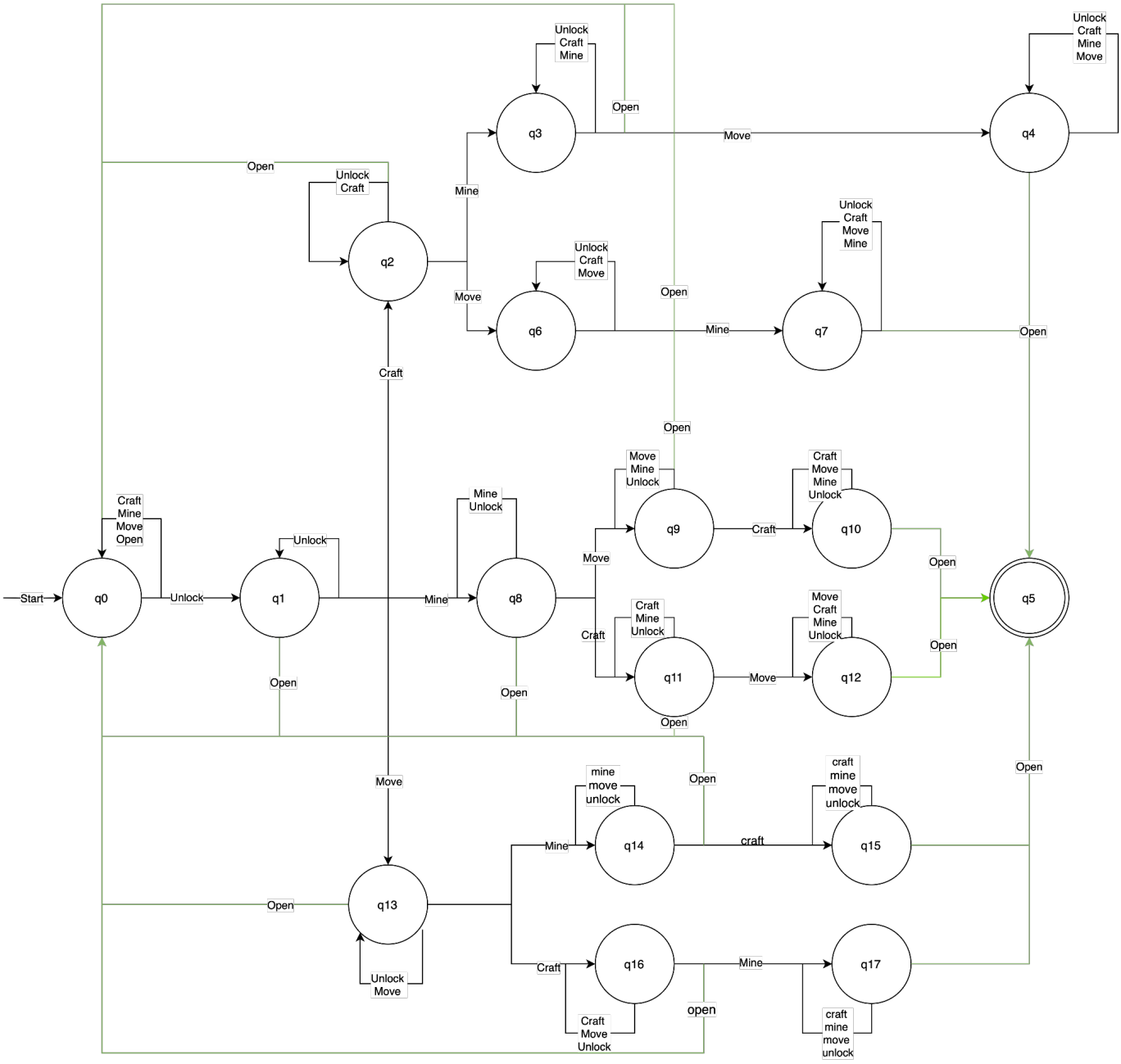
3 Functionality Exploration

List of key functionalities explored:

No.	Function Name	Description
1	main	Print out the game's description and initiate the game after user confirmation
2	initGame	create the world width, set the initial position of the player & create the inventory
3	generateWorld	generate blocks for all the coordinates on the map based on random values
4	displayWorld	display map on terminal
5	getBlockSymbol	function for getting block colors and symbols (some blocks are represented by symbols, e.g.: $)
6	getBlockChar	function for getting the characters for the blocks
7	startGame	ask for the action, & call a function for the specific action we choose. It also can open the secret door but only with some conditions.
8	fillInventory	fills up the inventory with all the existing blocktypes
	resetWorld	calls for the dutch flag function (generateEmptyWord) and moves the player to the middle of the screen
9	generateEmptyWorld	Generates the Dutch flag
10	clearScreen	clear the screen according to the os type
11	lookAround	prints out the blocks around the player
12	movePlayer	Changes player coordinates according to the input
13	mineBlock	It allows the player to mine the blocks to get materials if the specified coordinate is not air
14	placeBlock	It allows the player to put a material in a specific coordinate
15	getBlockTypeFromCraftedItem	Return the blockType from the itemID
16	getCraftedItemFromBlockType	Return the itemID from the blockType
17	displayCraftingRecipes	Prints out the crafting recipes.
18	craftItem	It allows the player to use their materials to create an item.
19	craftWoodenPlanks	It allows the player to use the wood they collected in the game and turn it into wooden planks
20	craftStick	It allows the player to use the wood they collected in the game and turn it into a stick
21	craftIronIngot	It allows the player to use the iron ore they collected in the game and turn it into iron ingots
22	inventoryContains	Checks whether the inventory contains a specific item.
23	inventoryContains	Checks whether the inventory contains a certain amount of items
24	removeItemsFromInventory	Remove a specified number of items from the inventory
25	addCraftedItem	Add itemID to the craftedItems list
26	interactWithWorld	Add block to inventory according to the place where the player stands
27	saveGame	Save the game variables, objects to an external file
28	loadGame	Load variables and objects from a previously saved game file
29	getBlockName	Returns the name of the block from blockType

30	displayLegend	print the legend with the symbols and names of the blocks
31	displayInventory	print the player's inventory with the number of blocks that the player has
32	getBlockColor	return the color of each block
33	waitForEnter	It tells the player to "press enter to continue" whenever they need to, and waits for input.
34	getCraftedItemName	return the name of the item from craftedItem
35	getCraftedItemColor	return the color off block, but only for the iron ingot (brown).
36	getCountryAndQuoteFromServer	Communicates with an external server in order to get a flag of a country and a quote besides it.

4 Finite State Automata (FSA) Design



The secret door is a hidden option within the game that is activated when entering the following commands into the terminal:

- Unlock
- Mine (user has to input “m” in either upper or lower case in order to mine)
- Move (user has to either character from WASD in upper or lower case for moving)
- Craft (user has to select any crafting option in this command)
- Open

For opening the secret door function, the user needs to input these commands in specific orders, where each command is entered one and without any repetition in the following inputs. The main conditions needed to enter the sequence are the following:

- The Unlock command always needs to be the first command to be inputted (although you can input this command in intermediate steps as we will mention later).
- Open command has to be strictly the last command to be inputted.
- In between the first input (unlock) and the last output (open), the remaining commands and/or unlock are to be inputted in any order.

With the input of these commands, we can describe the sequencing of the input of these commands as a change of states within the game, where we can identify an initial state and a final/accepting state and determine a finite state automaton (FSA). Here, the FSA's alphabet is the set of all the commands accepted by the game as valid inputs (as described in the outputs of the game) and the language accepted by this is the set of commands mentioned earlier. By applying the conditions for the sequence of inputs for unlocking the door, we can get the following representation of the language:

Σ = { L ∈ {valid inputs within the game}* | L contains ‘unlock’ as first string, has the following commands {‘unlock’, ‘mine’, ‘move’, ‘craft’} in any order and has ‘open’ as last string}

With this language, we will proceed to describe the steps required for reaching the secret door. First step, when the game locates us into the game right after entering the correct inputs for starting the game, we begin at the initial state q0, from which we move to q1 after entering the ‘unlock’ command. If we enter ‘unlock’ for a second time right after entering it from q0, we will not move from q1 as no new changes or alterations to the game are noticed, meaning we have a loop for ‘unlock’ in this state. From q1 and onwards, we will have loops in the states for all repeating commands that we input, meaning that we can only advance in the FSA by inputting the commands exactly once, as long as the condition that ‘unlock’ is the first command entered and ‘open’ is the last entered. For all cases, if the input ‘open’ is entered in intermediate states, the FSA returns the user to q0 losing all progress (as represented with the green line in the FSA diagram).

Upon opening the secret door the game plots a Dutch flag into the terminal and the player’s inventory gets filled up with all of the block types.

5 Git Collaboration & Version Control

- Repository Link: **https://gitlab.maastrichtuniversity.nl/bcs1110/javacraft/-/tree/group_43**
- Branch Details: **group_43**

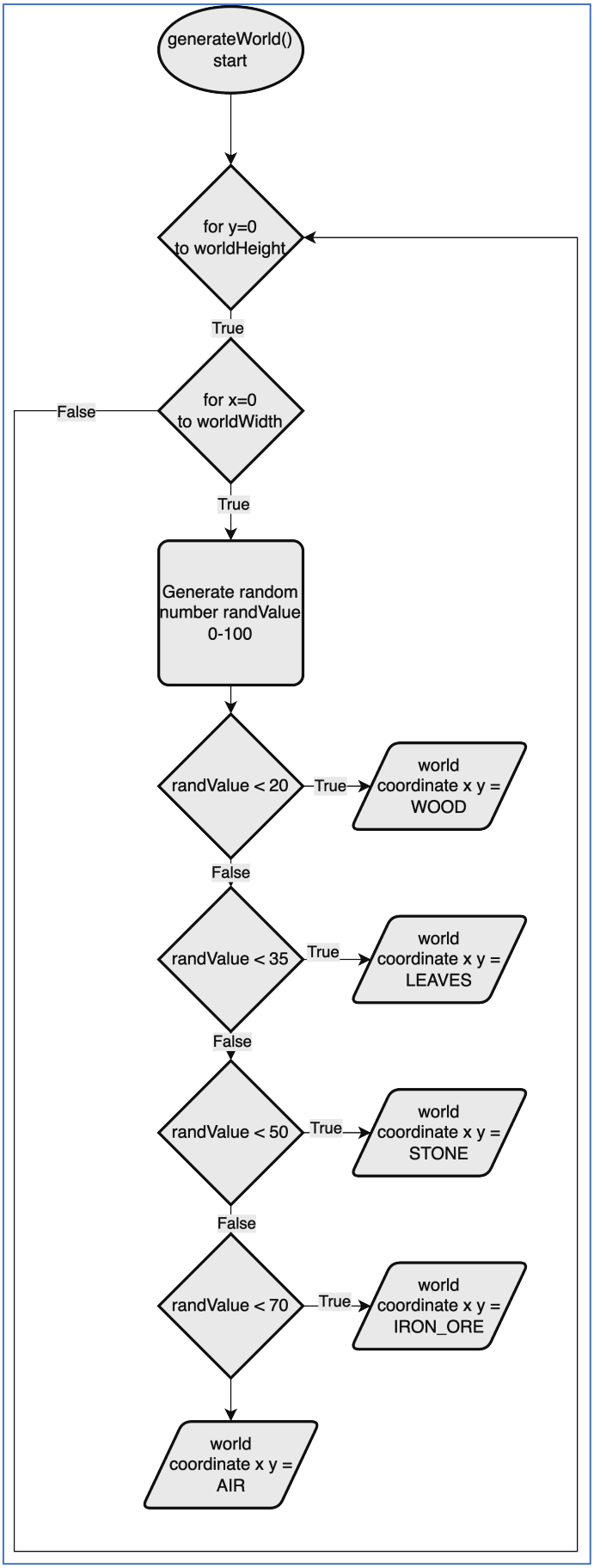
Changes & Conflicts & Problems:

- We had problems at the start understanding how to use git and how to commit/push/pull, but thankfully we had Botond there to help us understand all of this.
- We also had problems at one point when trying to push and pull, in which error messages kept appearing. We solved this by merging the previous branches, then pulling, and finally pushing.
- We had problems when setting up SourceTree, where we needed to create an access token and ssh keys in gitlab. We had many errors so we had to do the process all over again carefully and we made it work.
- Some conflicting branches were very problematic, so we had to copy paste parts of one to the other before effectively pushing them.

APPENDIX

Flowcharts & Pseudocode

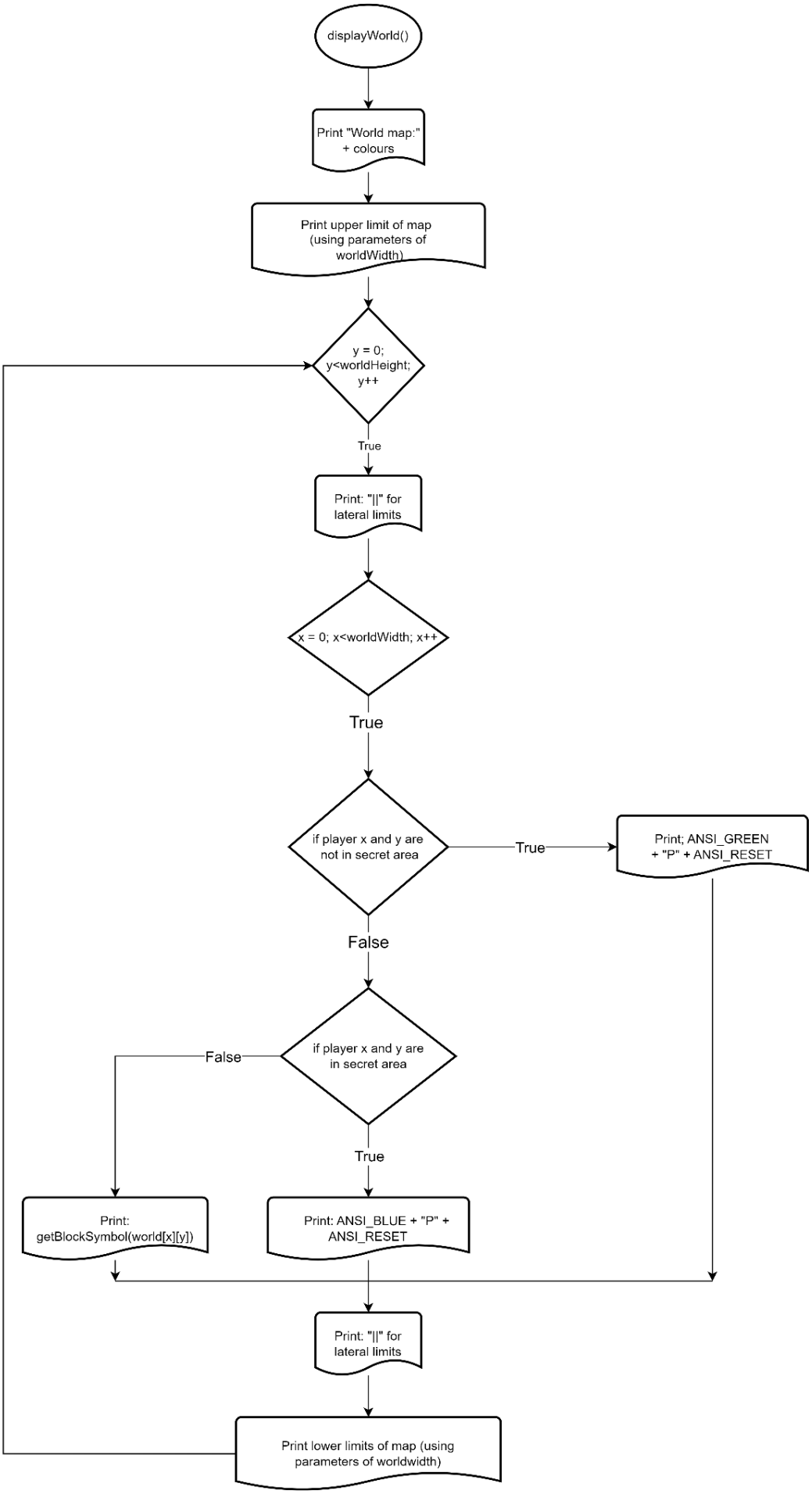
03 – generateWorld



Pseudocodes > 03-generateWorld

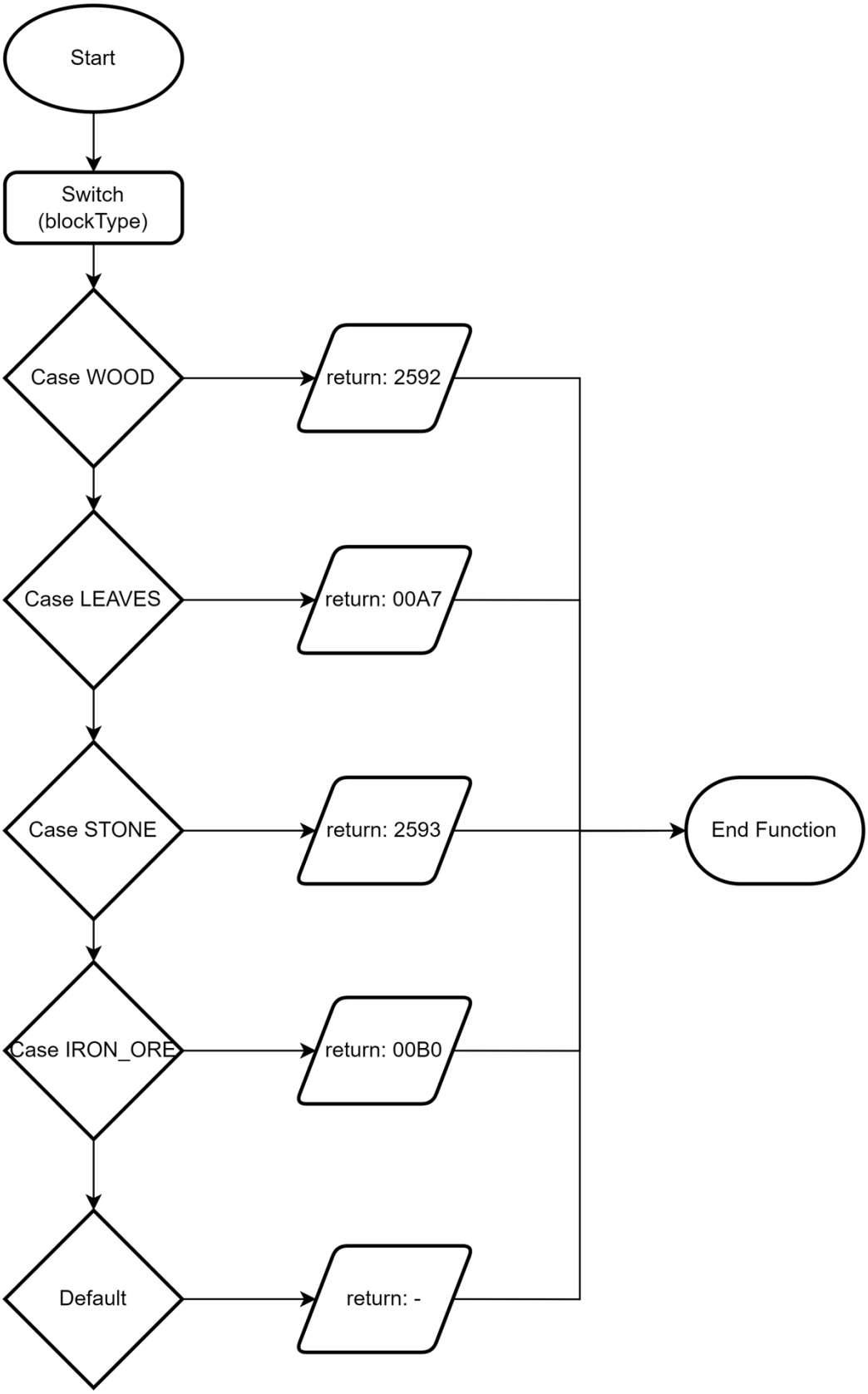
```
FUNCTION generateWorld
    OBJECT Random new rand
    FOR each y coordinate in wordHeight
        FOR each x coordinate in worldWidth
            SET randValue to new random value between 0-100
            IF randValue is smaller than 20 THEN
                SET world xy coordinate to WOOD
            ELSE IF randValue is smaller than 35 THEN
                SET world xy coordinate to LEAVES
            ELSE IF randValue is smaller than 50 THEN
                SET world xy coordinate to STONE
            ELSE IF randValue is smaller than 70 THEN
                SET world xy coordinate to IRON_ORE
            ELSE
                SET world xy coordinate to AIR
```

04 – displayWorld



06 – getBlockChar

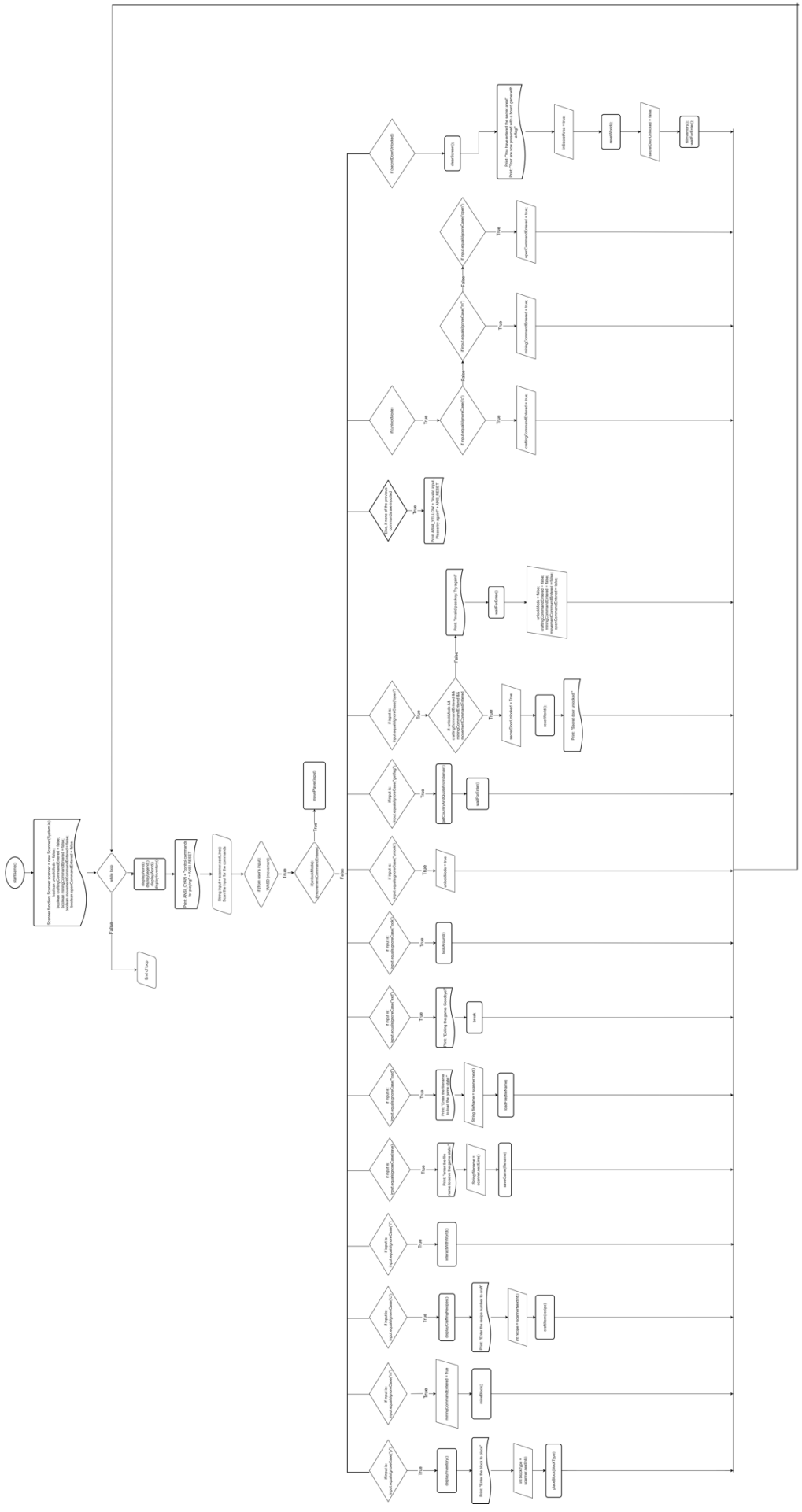
getBlockChar



Pseudocodes > ! 06-getBlockChar

```
1   FUNCTION getBlockChar
2       CASE block type
3           "WOOD":
4               RETURN: 2592
5           "LEAVES":
6               RETURN: 00A7
7           "STONE":
8               RETURN: 2593
9           "IRON_ORE":
10              RETURN: 00B0
11          "DEFAULT":
12              RETURN: nothing
13      ENDCASE
```

07 – startGame

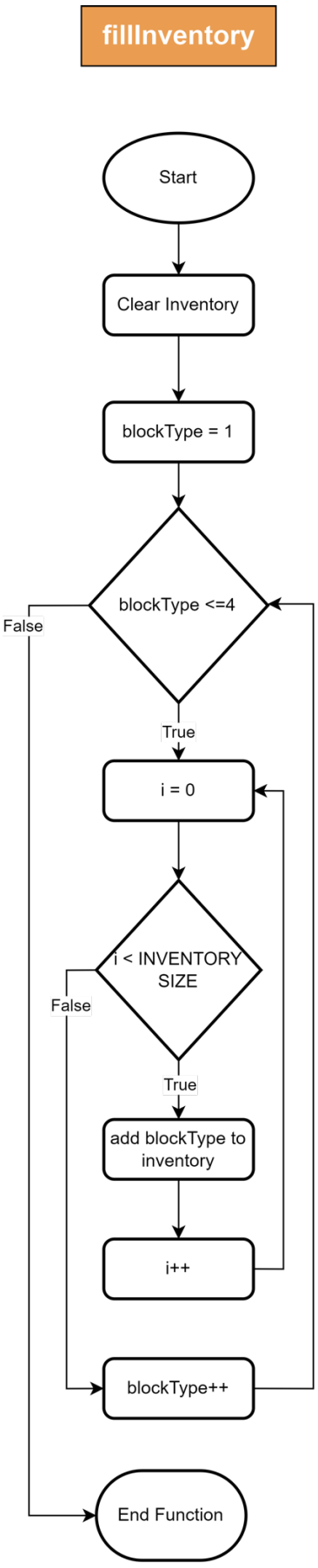


Pseudocodes > 07-startGame

```
START GAME
    scanner is equal to new Scanner(System.in)
    unlockMode is false
    craftingCommandEntered is false
    miningCommandEntered is false
    movementCommandEntered is false
    openCommandEntered is false
    WHILE true
        clearScreen
        displayLegend
        displayWorld
        displayInventory
        PRINT  "Enter your action: 'WASD': Move, 'M': Mine, 'P': Place, 'C': Craft,
        'I': Interact, 'Save': Save, 'Load': Load, 'Exit': Quit, 'Unlock': Unlock Secret Door"
        scanner scans the value for unserinput
        IF any of these commands are entered (ignoring case) ("w" || "up", "s" || "down", "a" || "left", "d" || "right")
            IF it is in unlockMode THEN
                movementCommandEntered is true
                movePLAYER(Input)
            ELSE IF INPUT is "m"
                miningCommandEntered is true
                mineBlock
            ELSE IF INPUT is "p"
                displayInventory
                PRINT "Enter the block type to place: "
                blockType is input(integer)
                placeBlock(blockType)
            ELSE IF INPUT is "c"
                displayCraftingRecipes
                PRINT "Enter the recipe number to craft: "
                recipe is input(integer)
                craftItem(recipe)
            ELSE IF INPUT is "i"
                interactWithWorld
            ELSE IF INPUT is "save"
                PRINT "Enter the file name to save the game state: "
                fileName is input
                saveGame(fileName)
            ELSE IF INPUT is "load"
                PRINT "Enter the file name to load the game state: "
                fileName is input
                loadGame(fileName)
```

```
        ELSE IF INPUT is "exit"
            PRINT "Exiting the game. Goodbye!"
            break
        ELSE IF INPUT is "look"
            lookAround
        ELSE IF INPUT is "unlock"
            unlockMode is true
        ELSE IF INPUT is "getFlag"
            getCountryAndQuoteFromServer
            waitForEnter
        ELSE IF INPUT is "open"
            IF unlockMode && craftingCommandEntered && miningCommandEntered && movementCommandEntered THEN
                secretDoorUnlocked is true
                resetWorld
                PRINT "Secret door unlocked!"
                waitForEnter
            ELSE
                PRINT "Passkey invalid. Try again!"
                unlockMode is false
                craftingCommandEntered is false
                miningCommandEntered is false
                movementCommandEntered is false
                openCommandEntered is false
    ELSE
        PRINT "Invalid input. Please try again" in yellow
    IF unlockMode
        IF INPUT is "c" THEN
            craftingCommandEntered = true
        ELSE IF INPUT is "m"
            miningCommandEntered is true
        ELSE IF INPUT is "open"
            openCommandEntered is true
    IF secretDoorUnlocked
        clearScreen
        PRINT "You have entered the secret area!"
        PRINT "You are not presented with a board game with a flag!"
        inSecretArea is true
        resetWorld
        secretDoorUnlocked is false
        fillInventory
        waitForEnter
```

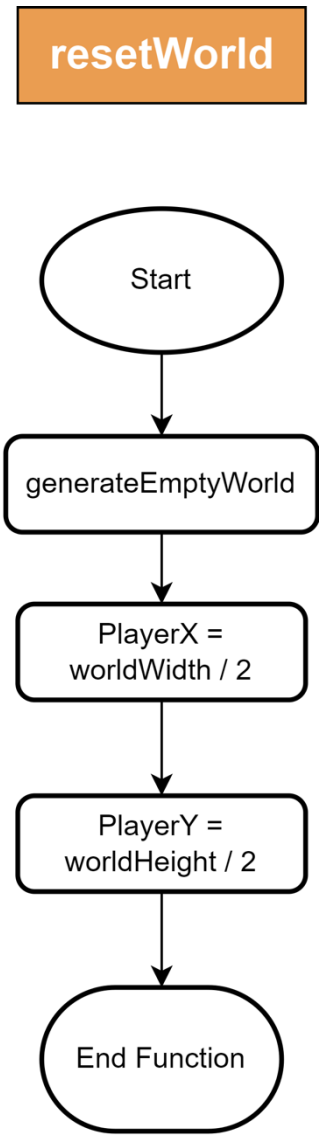
08 – fillInventory



Pseudocodes > 08-fillInventroy

```
1   FUNCTION fillInventory
2       CLEAR inventory
3       FOR each blockType from 1 to 4
4           FOR each in the inventory size
5               ADD block type to the inventory
```

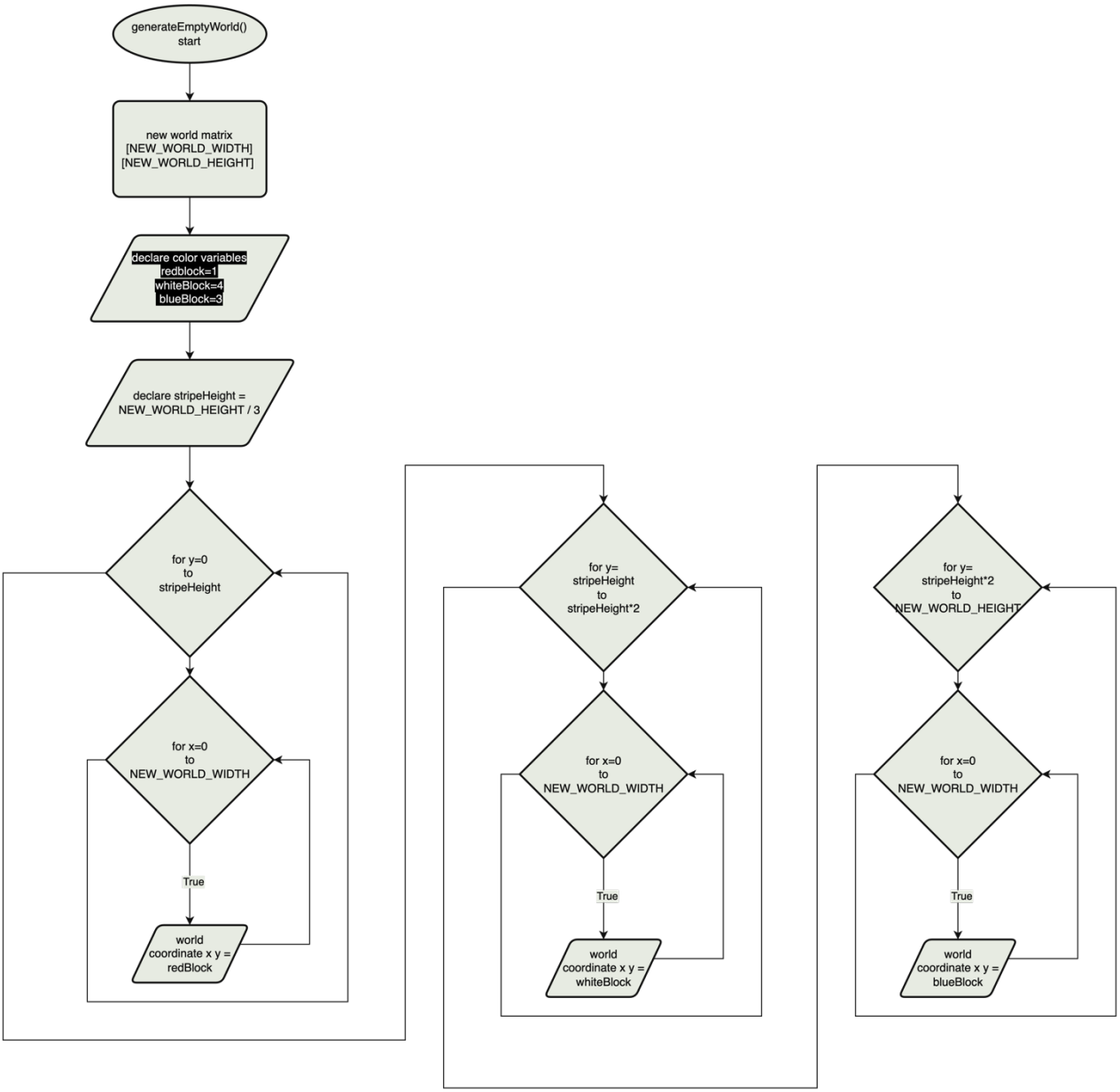
09 – resetWorld



Pseudocodes > 09-resetWorld

```
1   FUNCTION resetWorld
2       GENERATE empty world
3       SET variable playerX to the worldwidth divided by 2
4       SET variable playerY to the worldheight divided by 2
```

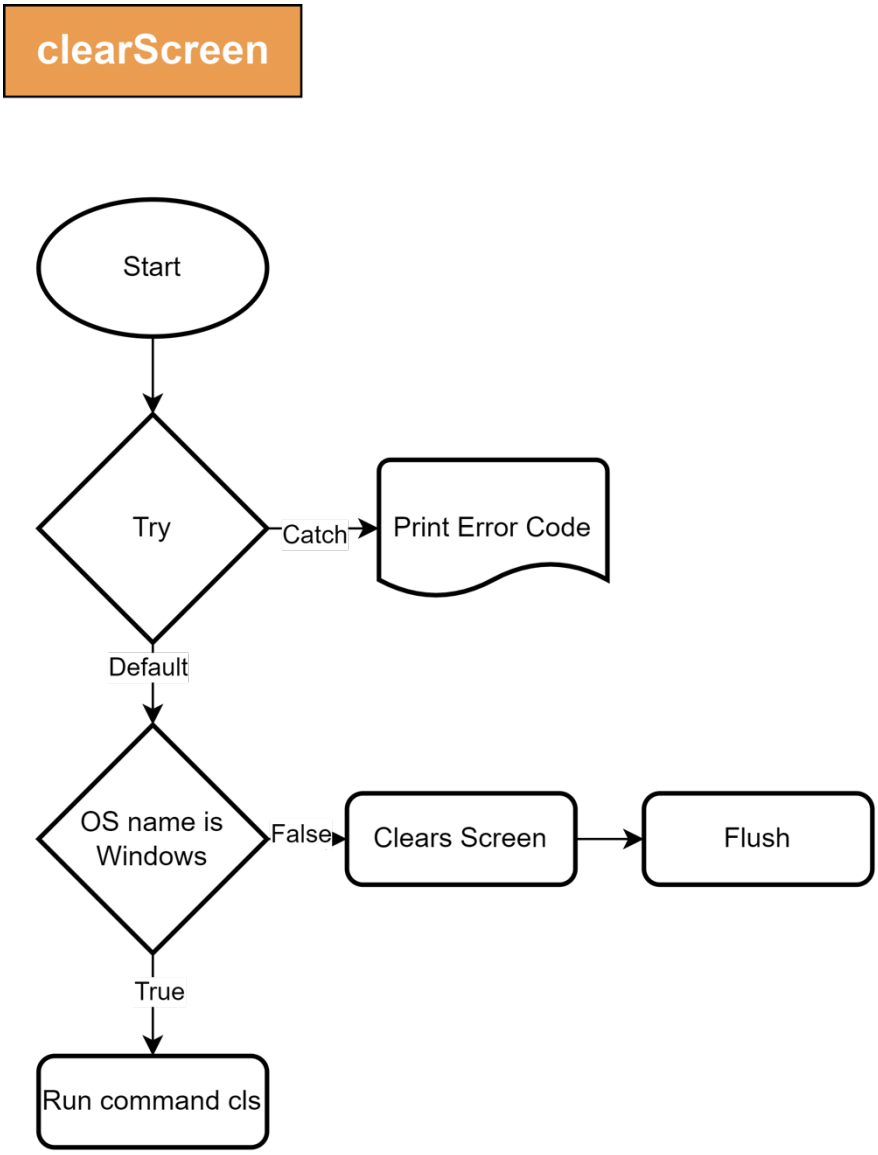
10 – generateEmptyWorld



Pseudocodes > 10-generateEmptyWorld

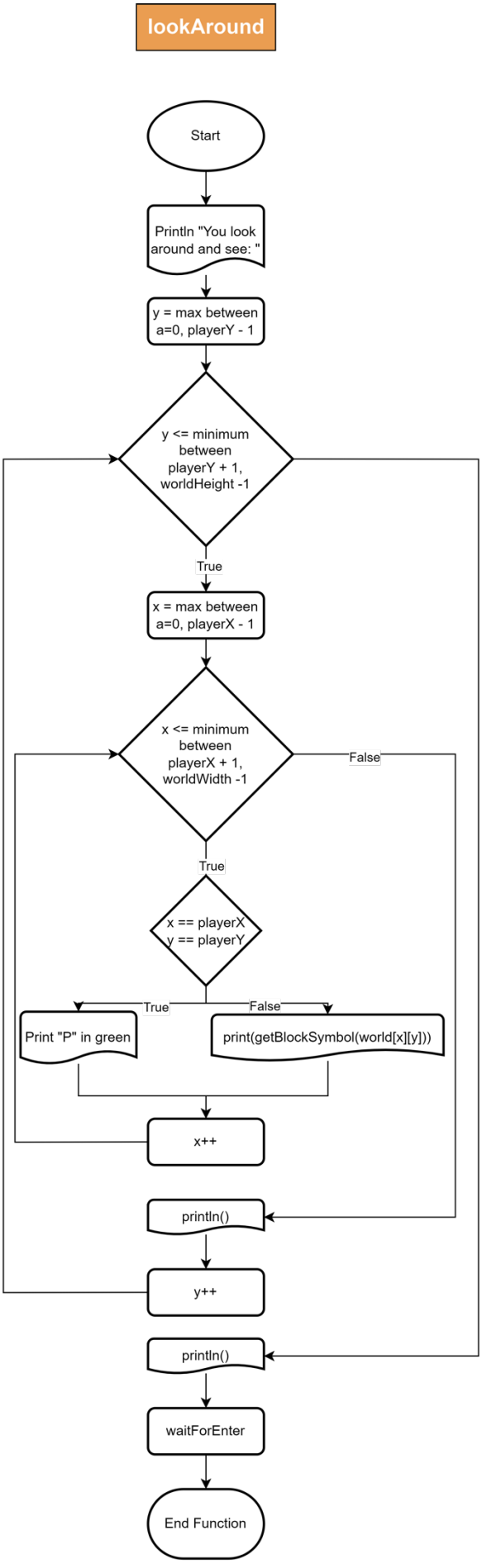
```
FUNCTION generateEmptyWorld
    SET world to interger array (NEW_WORLD_WIDTH, NEW_WORLD_HEIGHT)
    SET redBlock to 1
    SET whiteBlock to 4
    SET blueBlock to 3
    SET stripeHeight to one third of the NEW_WORLD_HEIGHT

    FOR each y coordinate in stripeHeight
        FOR each x coordinate in NEW_WORLD_WIDTH
            SET world xy coordinate to redBlock

    FOR each y coordinate between stripeHeight and 2 times stripeHeight
        FOR each x coordinate in NEW_WORLD_WIDTH
            SET world xy coordinate to whiteBlock

    FOR each y coordinate between 2 times stripeHeight and NEW_WORLD_HEIGHT
        FOR each x coordinate in NEW_WORLD_WIDTH
            SET world xy coordinate to blueBlock
```

11 – clearScreen




```
Pseudocodes > 11-clearScreen
1   FUNCTION clearScreen
2       TRY unless an error happens
3           IF the OS name is Windows
4               RUN command cls
5           ELSE
6               CLEAR the screen
7               FLUSH
8       CATCH when an error happens
9           PRINT error code
```

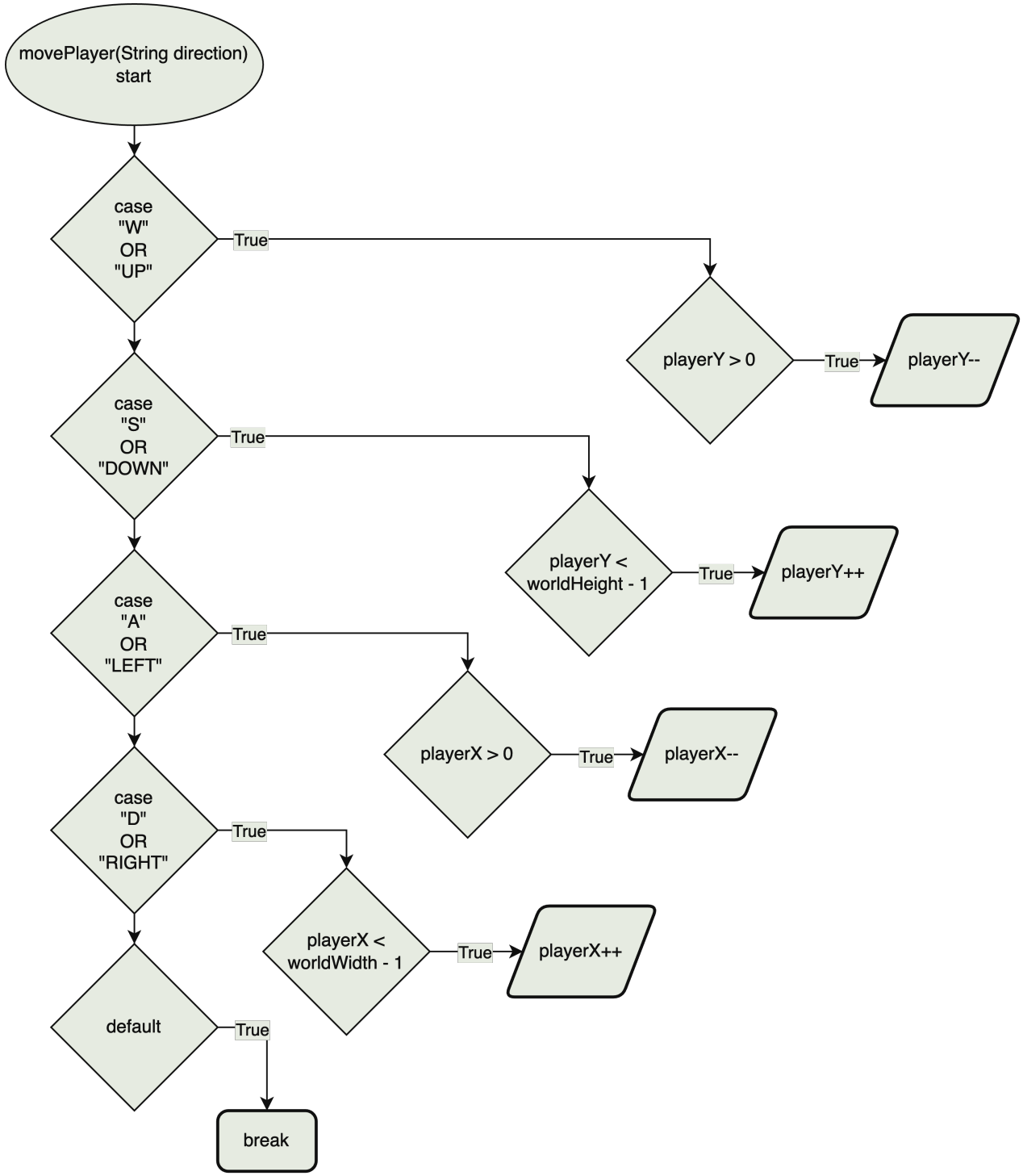
12 – lookAround



Pseudocodes > 12-lookAround

```
FUNCTION lookAround
    PRINT "You look around and see"
    FOR each y coordinate that is lower or equal to the minimum between playerY +1, and the worldheight -1
        FOR each x coordinate that is lower or equal to the minimum between playerX +1, and the worldwidth -1
            IF the x coordinate equals playerX, and the y coordinate equals playerY
                PRINT "P" in green
            ELSE
                PRINT getBlockSymbol(world[x][y])
        PRINT a new line
    PRINT a new line
    WAIT for enter
```

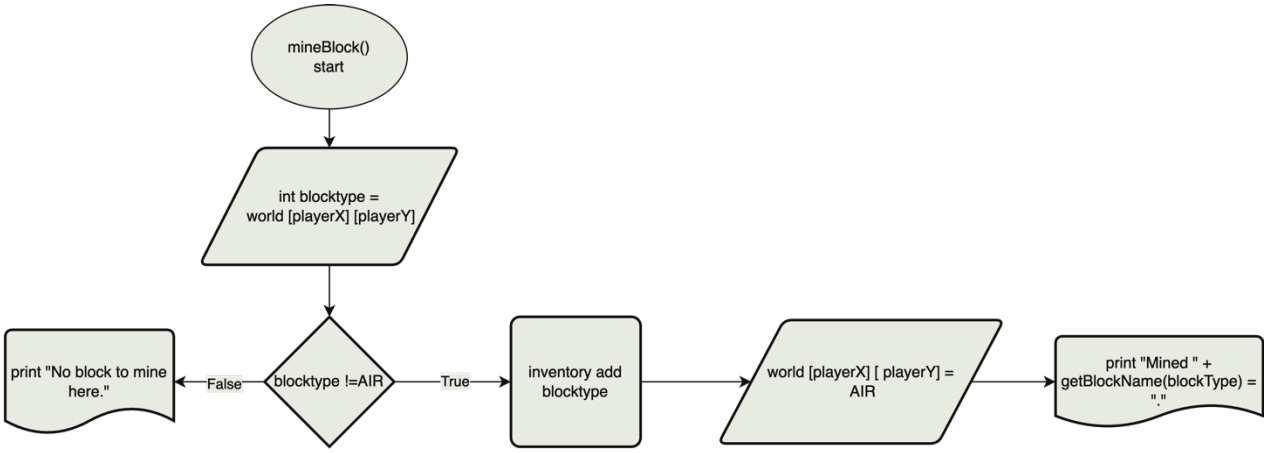
13 – movePlayer



Pseudocodes > 13-movePlayer

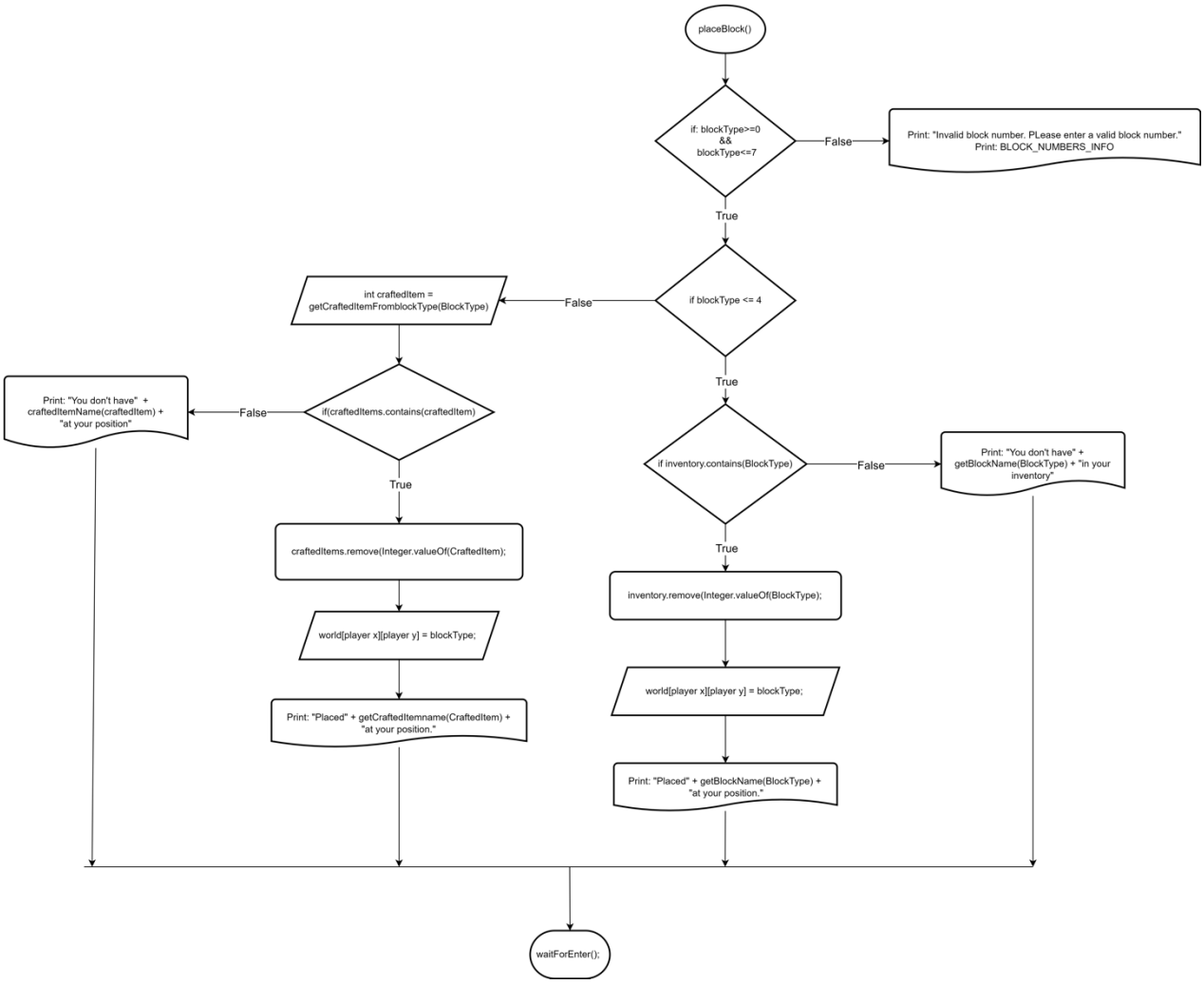
```
FUNCTION movePlayer
    CASE uppercase direction
        "W" OR "UP":
            IF playerY is greater than 0 THEN
                DECREMENT playerY
        "S" OR "DOWN":
            IF playerY is smaller than wordHeight-1 THEN
                INCREMENT playerY
        "A" OR "LEFT":
            IF playerX is greater than 0 THEN
                DECREMENT playerX
        "D" OR "RIGHT":
            IF playerX is smaller than worldWidth-1 THEN
                INCREMENT playerX
        DEFAULT:
    ENDCASE
```

14 – mineBlock




```
Pseudocodes > 14-mineBlock
1  FUNCTION mineBlock
2      SET blockType to world(playerX, playerY)
3      IF blockType is not AIR THEN
4          ADD inventory blockType
5          SET world(playerX,playerY) to AIR
6          PRINT "Mined" GET blockname "."
7      ELSE
8          PRINT "No block to mine here."
```

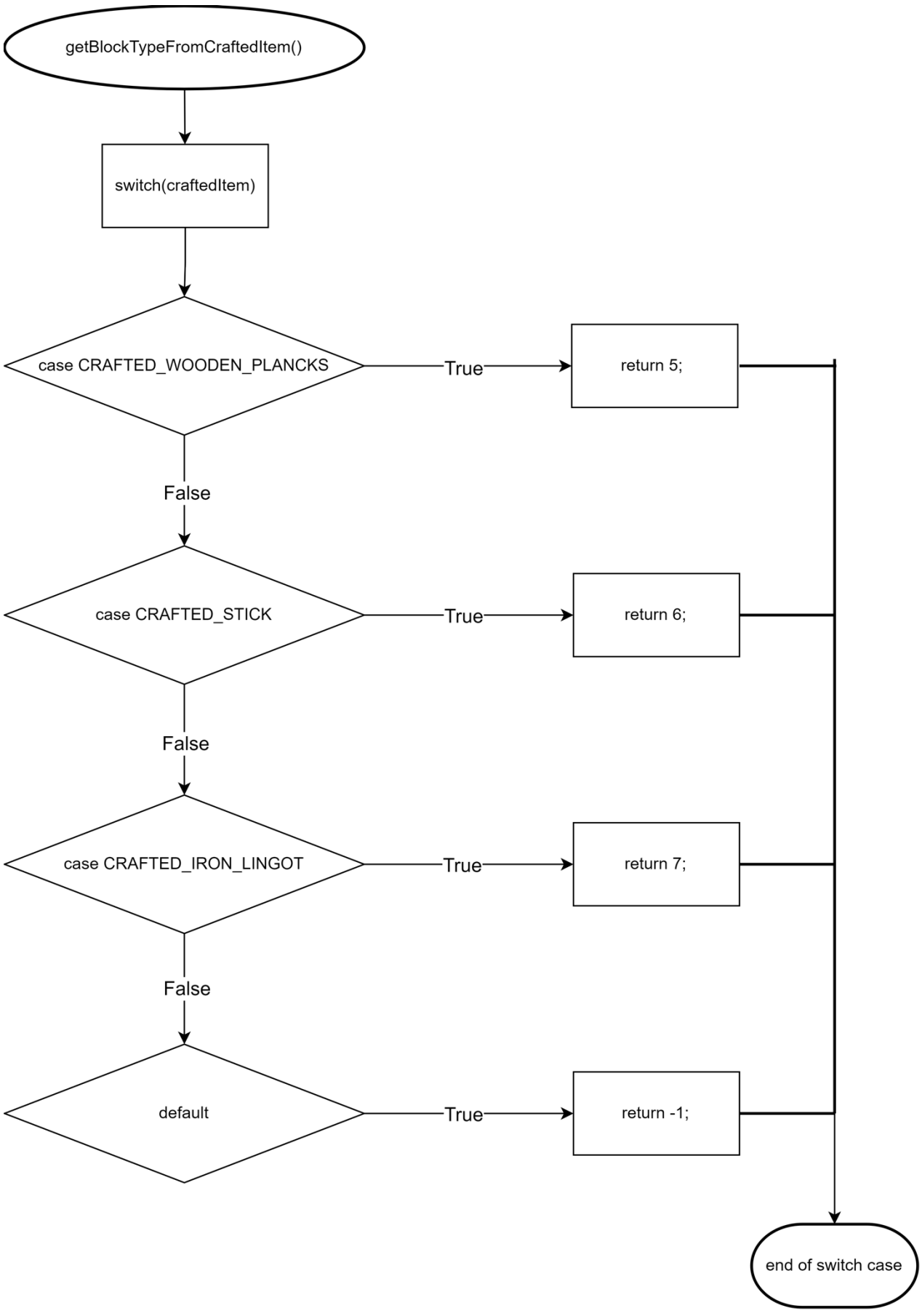
15 – placeBlock



Pseudocodes > 15-placeBlock

```
PLACE BLOCK
    IF blockType is greater or equal to 0 and blockType is also smaller or equal to 7
        IF blockType is smaller or equal to 4
            IF inventory contains blockType
            Remove an integer value of blockType in inventory
            world[player X][player Y] is equal to blockType
            PRINT "Placed " + getBlockName(blockType) + " at your position."
            ELSE
                PRINT "You don't have " + getBlockName(blockType) + " in your inventory."
        ELSE
            craftedItem equals to getCraftedItemFromBlockType(blockType)
            IF craftedItems contains craftedItem THEN
                Remove an integer value in craftedItem
                world[playerX][playerY] is equal to blockType
                PRINT "Placed " + getCraftedItemName(craftedItem) + " at your position."
            ELSE
                PRINT "You don't have " + getCraftedItemName(craftedItem) + " in your crafted items."
    ELSE
        PRINT "Invalid block number. Please enter a valid block number."
        PRINT BLOCK_NUMBERS_INFO
    waitForEnter
```

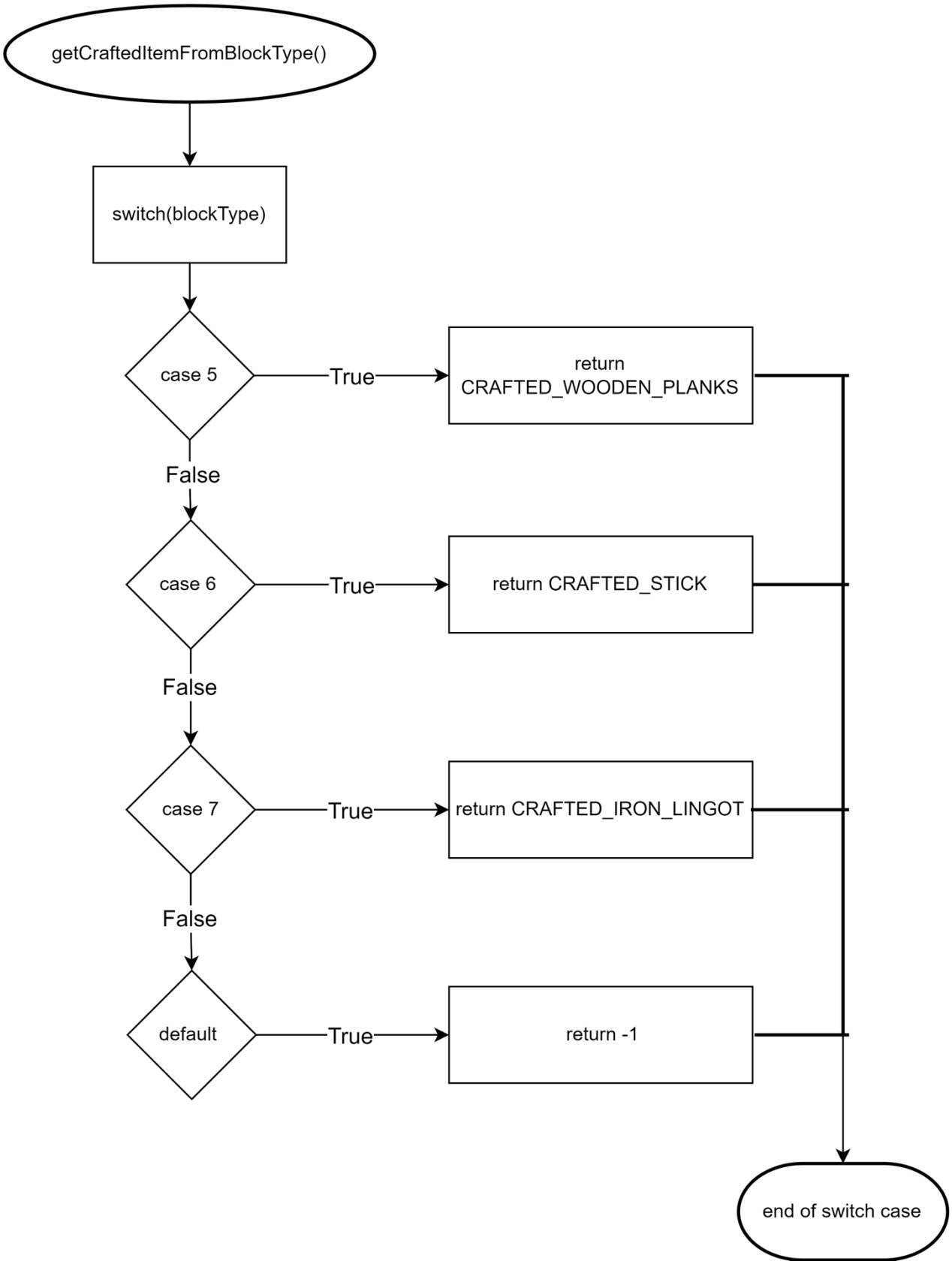
16 – getBlockTypeFromCraftedItem



Pseudocodes > 16-getBlockTypeFromCraftedItem

```
GET BLOCK TYPE FROM CRAFTED ITEM
    SWITCH craftedItem
        CASE CRAFTED_WOODEN_PLANKS:
            return 5
        CASE CRAFTED_STICK:
            return 6
        CASE CRAFTED_IRON_INGOT:
            return 7
        DEFAULT:
            return -1
```

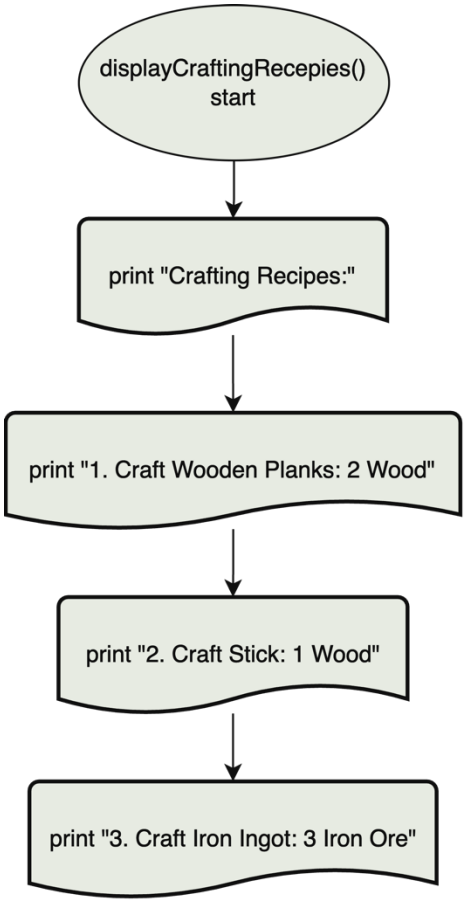
17 – getCraftedItemFromBlockType



Pseudocodes > 17-getCraftedItemFromBlockType

```
GET CRAFTED ITEM FROM BLOCK TYPE
    SWITCH blockType
        CASE 5
            return CRAFTED_WOODEN_PLANKS
        CASE 6
            return CRAFTED_STICK
        CASE 7
            return CRAFTED_IRON_INGOT
        DEFAULT
            return -1
```

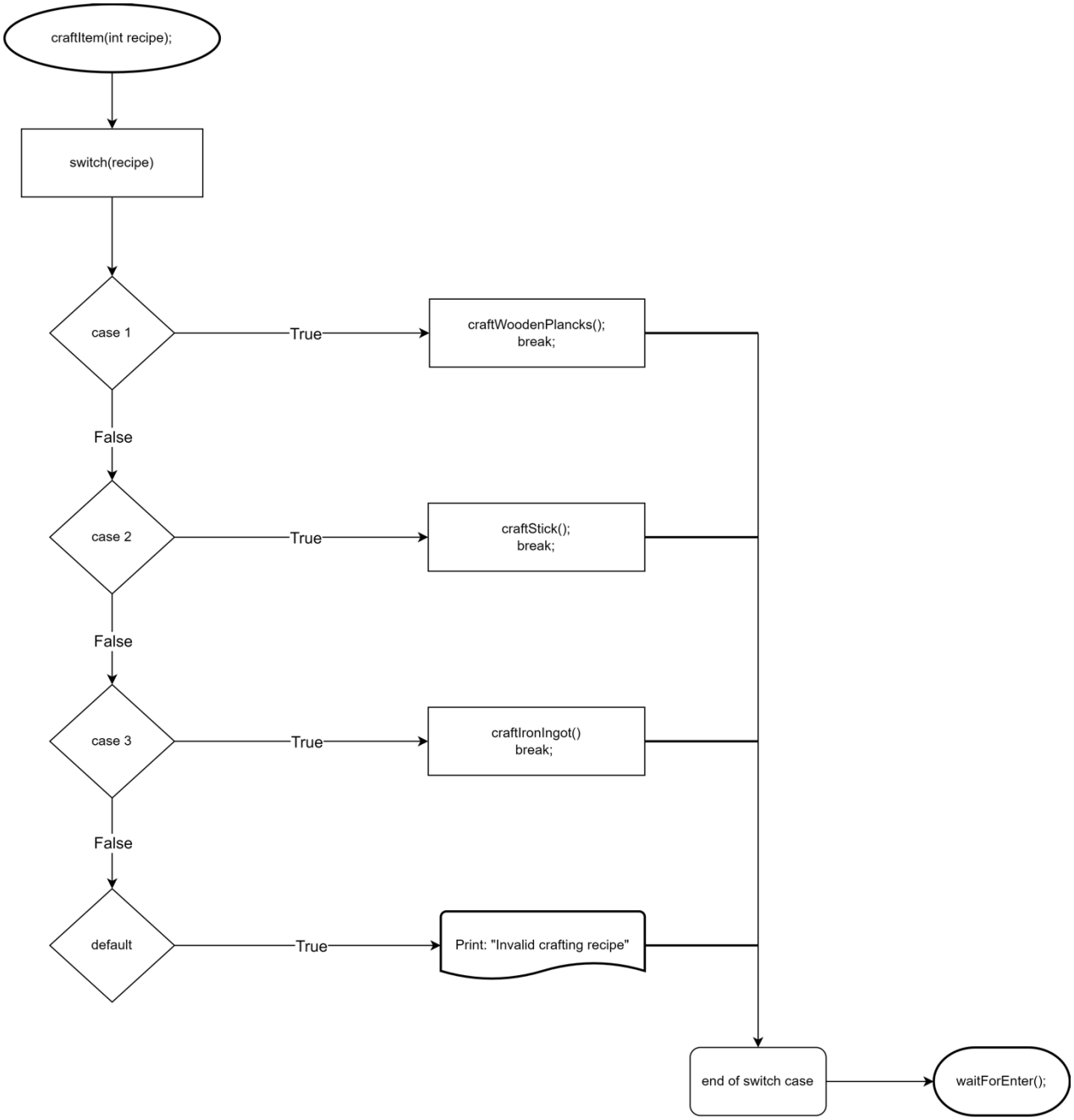
18 – displayCraftingRecepies



Pseudocodes > 18-displayCraftingRecipes

```
1   FUNCTION displayCraftingRecipes
2       PRINT "Crafting Recipes:"
3       PRINT "1. Craft Wooden Planks: 2 Wood"
4       PRINT "2. Craft Stick: 1 Wood"
5       PRINT "3. Craft Iron Ingot: 3 Iron Ore"
```

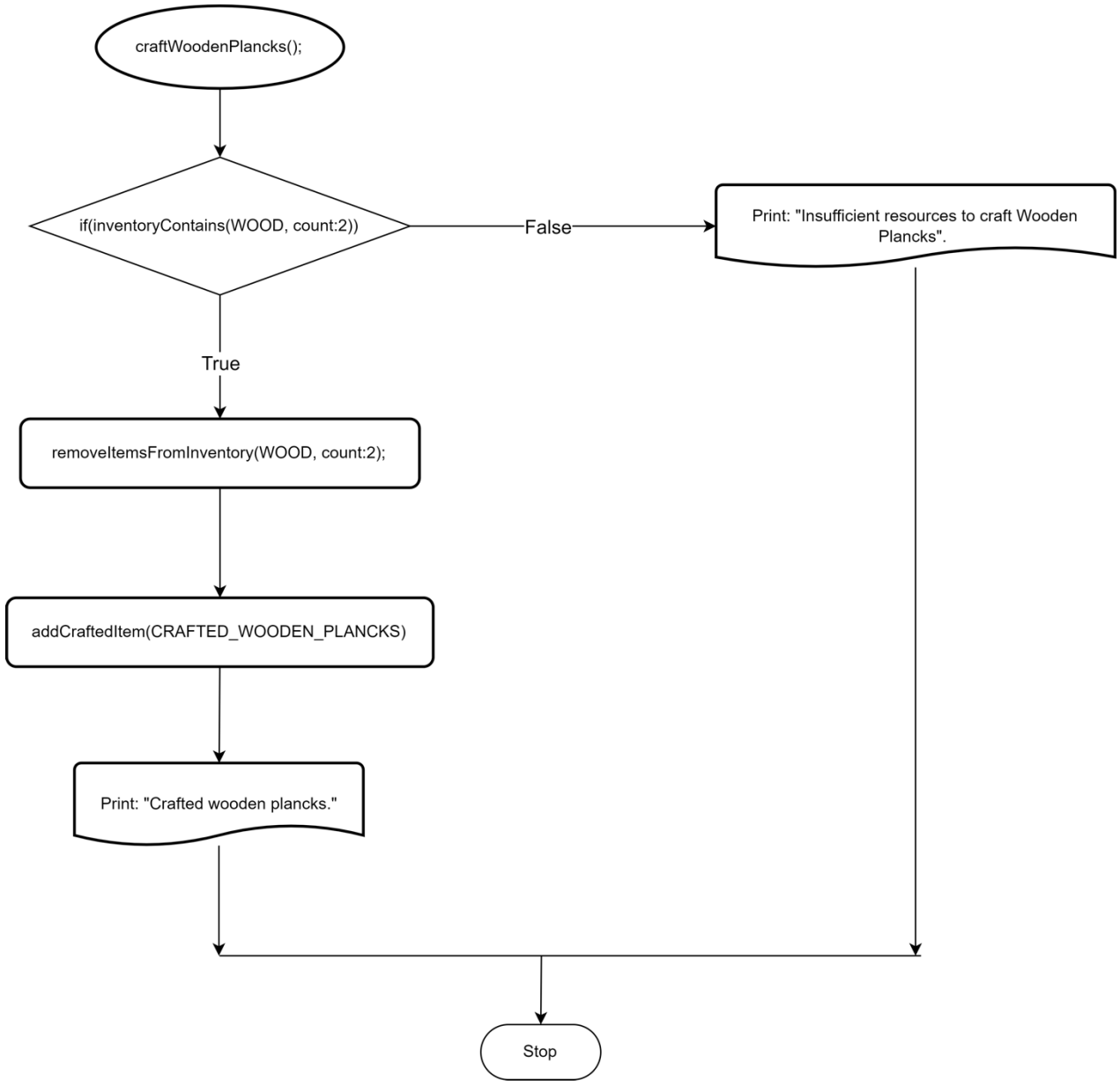
19 – craftItem



Pseudocodes > 19-craftItem

```
CRAFT ITEM
    SWITCH recipe
        CASE 1
            craftWoodenPlanks
            break
        CASE 2
            craftStick
            break
        CASE 3
            craftIronIngot
            break
        DEFAULT
            PRINT "Invalid recipe number."
    waitForEnter
```

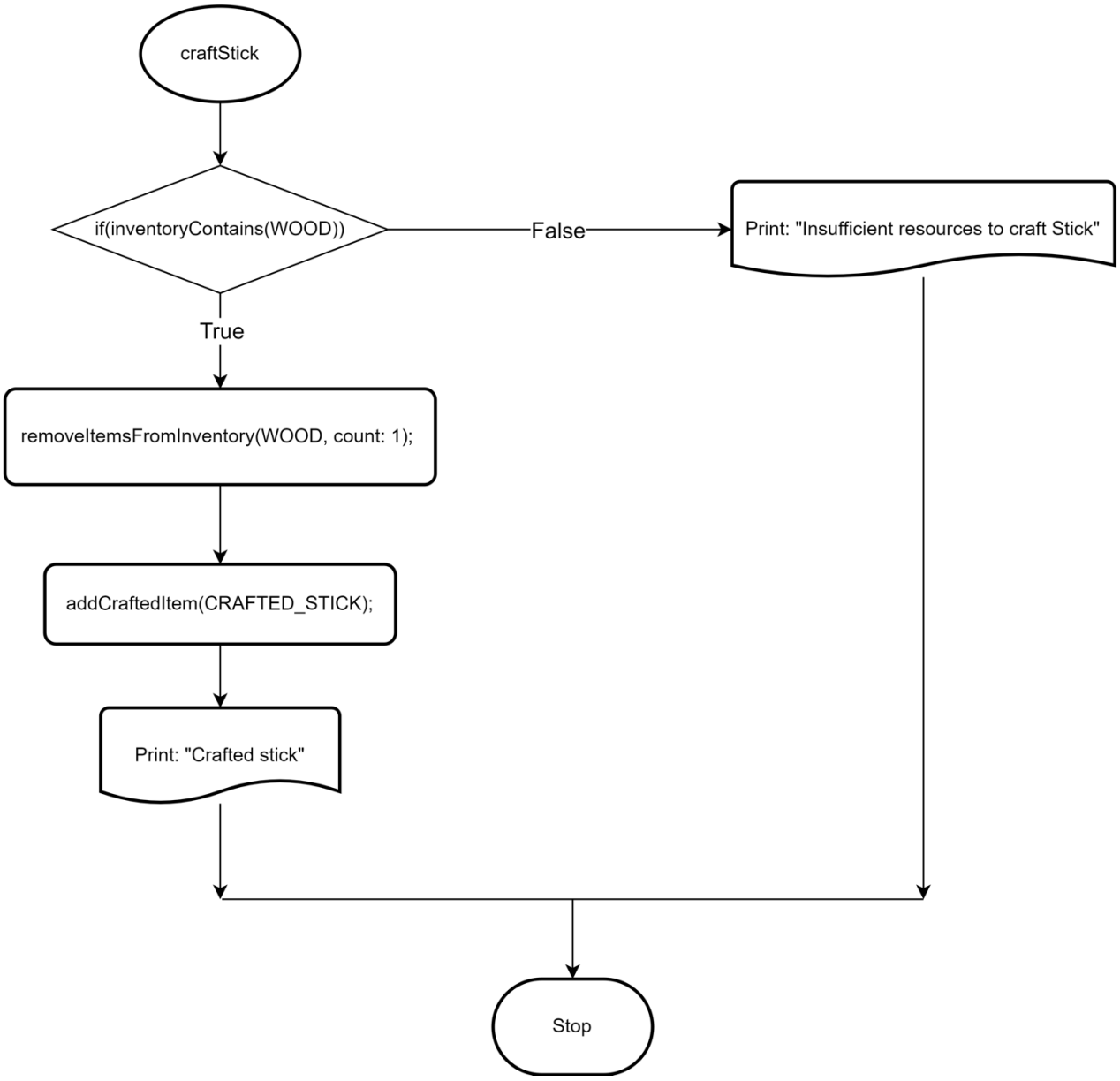
20 – craftWoodenPlanks



Pseudocodes > 20-craftWoodenPlanks

```
CRAFT WOODEN PLANKS
    IF inventory contains wood (2 units min) THEN
        Remove 2 units of wood from inventory
        addCraftedItem(CRAFTED_WOODEN_PLANKS)
        PRINT "Crafted wooden planks"
    ELSE
        PRINT "Insufficient resources to craft Wooden Plancks. "
```

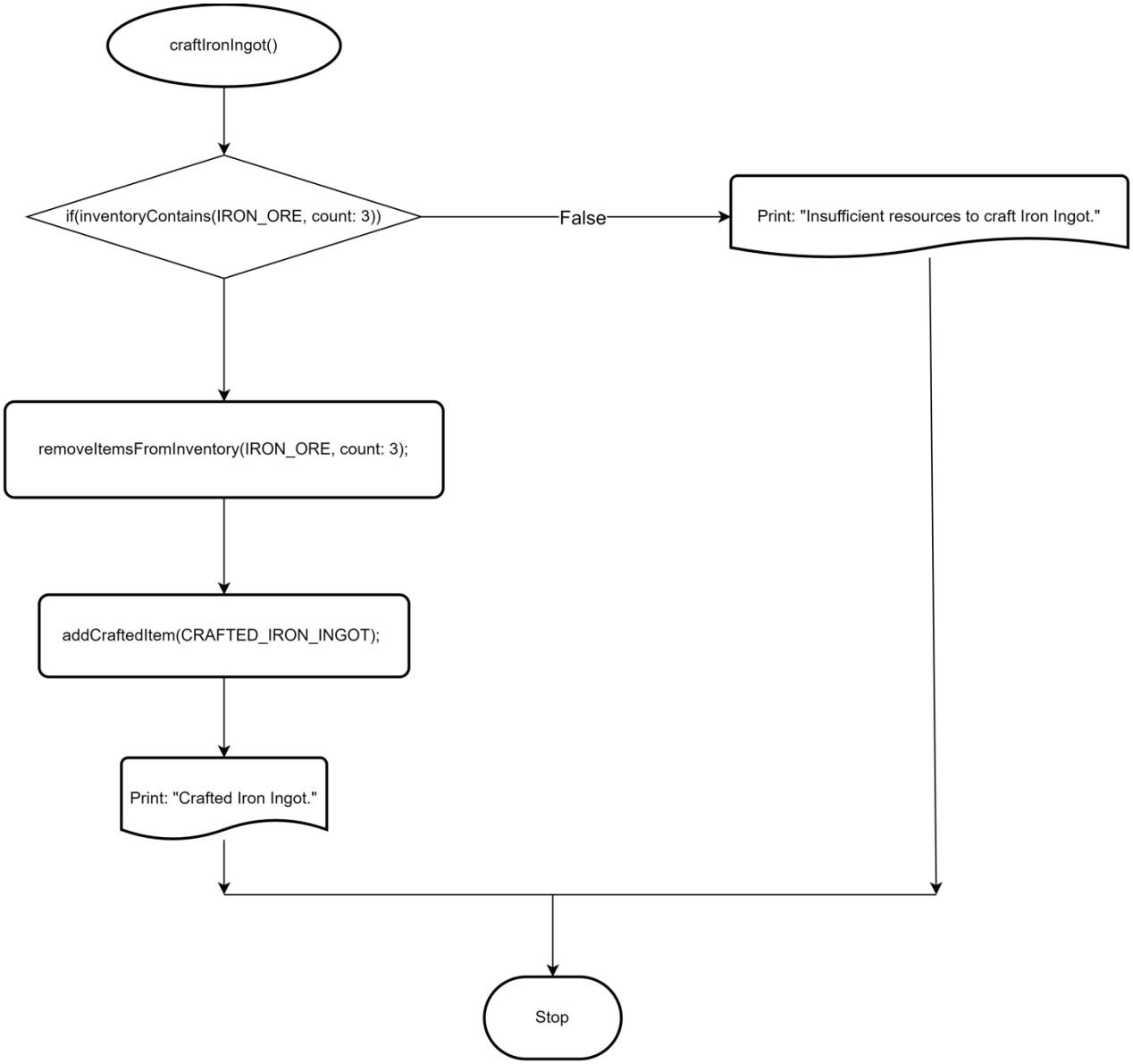
21 – craftStick



Pseudocodes > 21-craftStick

```
1   CRAFT STICK
2       IF  inventory contains wood THEN
3           Remove 1 unit of of wood from inventory
4           addCraftedItem(CRAFTED_STICK)
5           PRINT "Crafted Stick."
6       ELSE
7           PRINT "Insufficient resources to craft Stick."
```

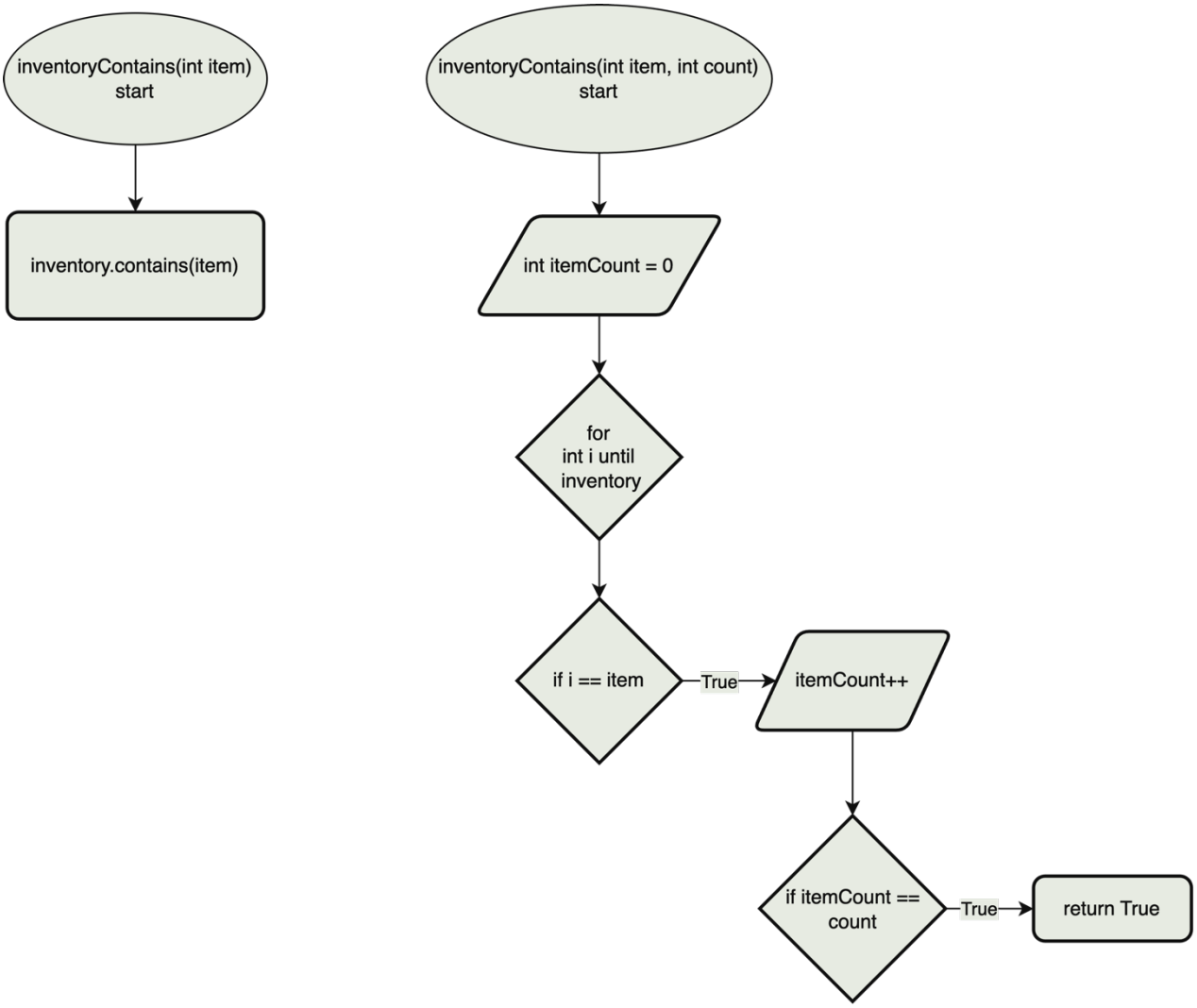
22 – craftIronIngot



Pseudocodes > 22-craftIronIngot

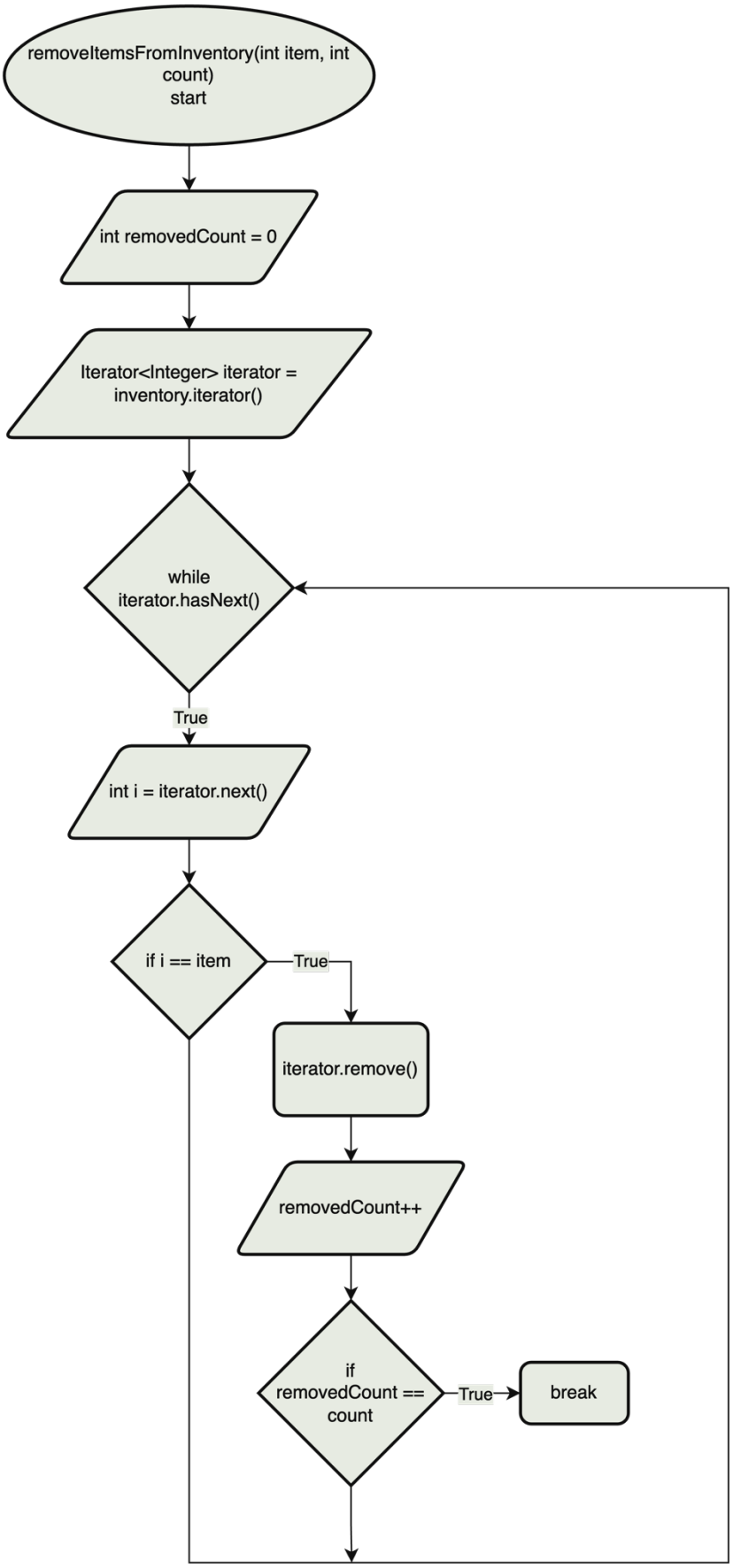
```
CRAFT_IRON_INGOT
    IF inventory contains iron ore (3 units min) THEN
        Remove 3 units of iron ore from inventory
        addCraftedItem(CRAFTED_IRON_INGOT)
        PRINT "Crafted Iron Ingot."
    ELSE
        PRINT "Insufficent resources to craft Iron Ingot"
```

23-24 – inventoryContains




```
Pseudocodes > 23-24-inventoryContains
  1   FUNCTION inventoryContains(integer item)
  2       RETURN wheter inventory contains(item)
  3
  4   FUNCTION inventoryContains(integer item, integer count)
  5       SET itemCount to 0
  6       FOR each i in inventory
  7           IF i equals to item THEN
  8               INCREMENT itemCount
  9               IF itemCount equals count THEN
 10                   RETURN true
 11       RETURN false
```

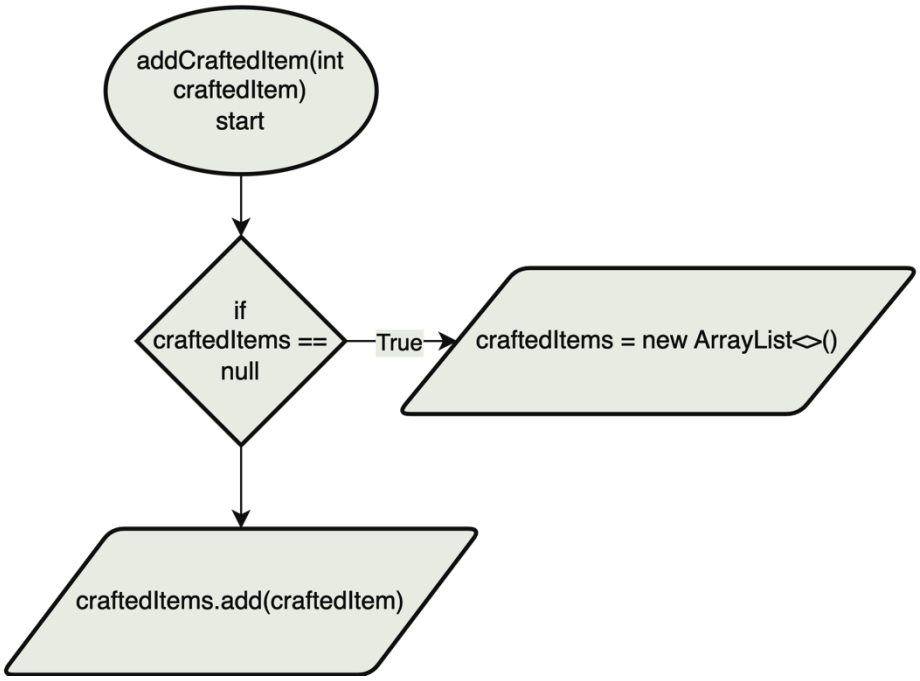
25 – removeItemsFromInventory



Pseudocodes > 25-removeItemsFromInventory

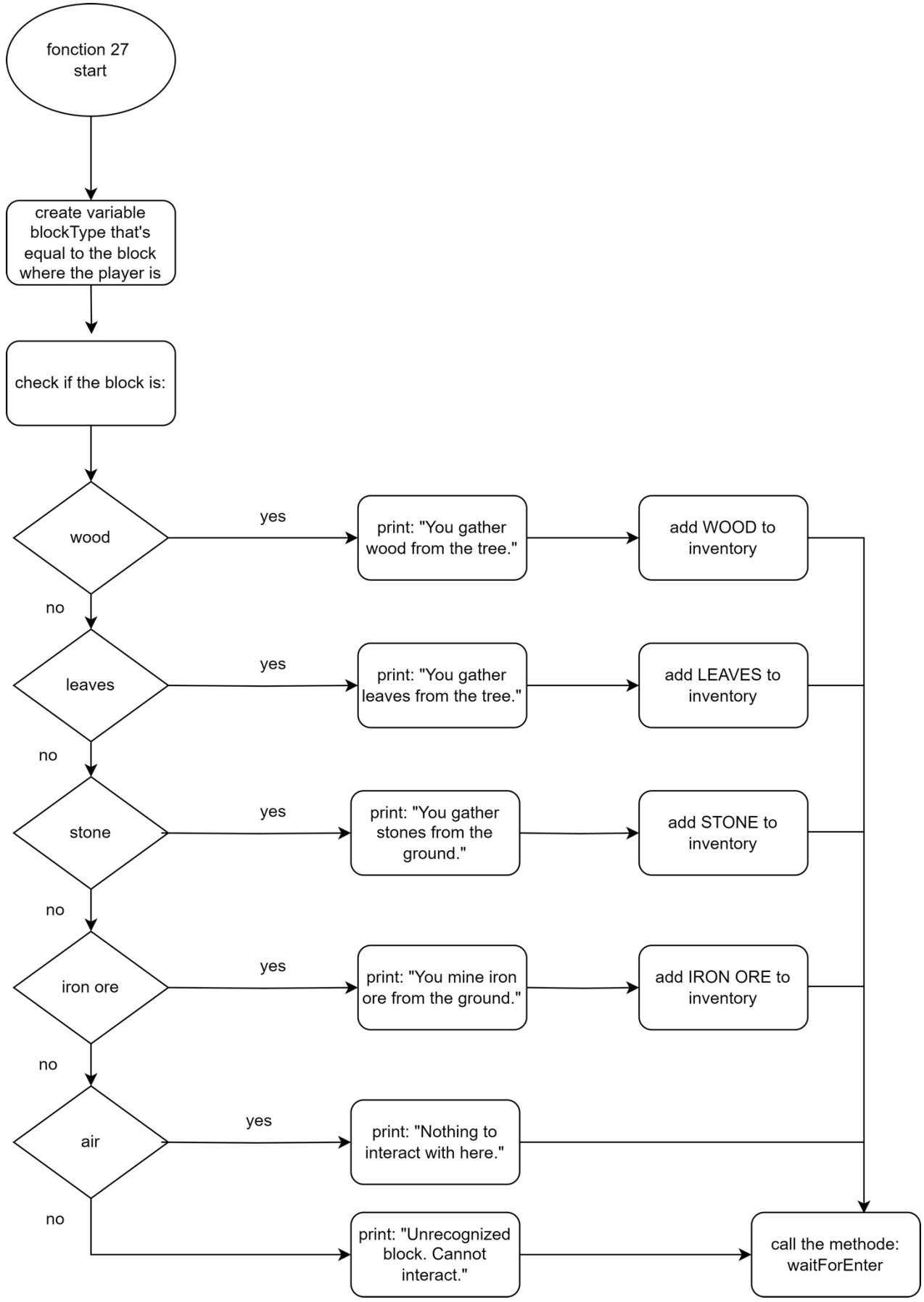
```
FUNCTION removeItemsFromInventory
    SET removedCount to 0
    NEW iterator inventory.iterator
    WHILE iterator has next item
        SET i to next iterator item
        IF i equals to item
            REMOVE iterator item
            INCREMENT removedCount
            IF removedCount equals to count THEN
                BREAK
```

26 – addCraftedItem

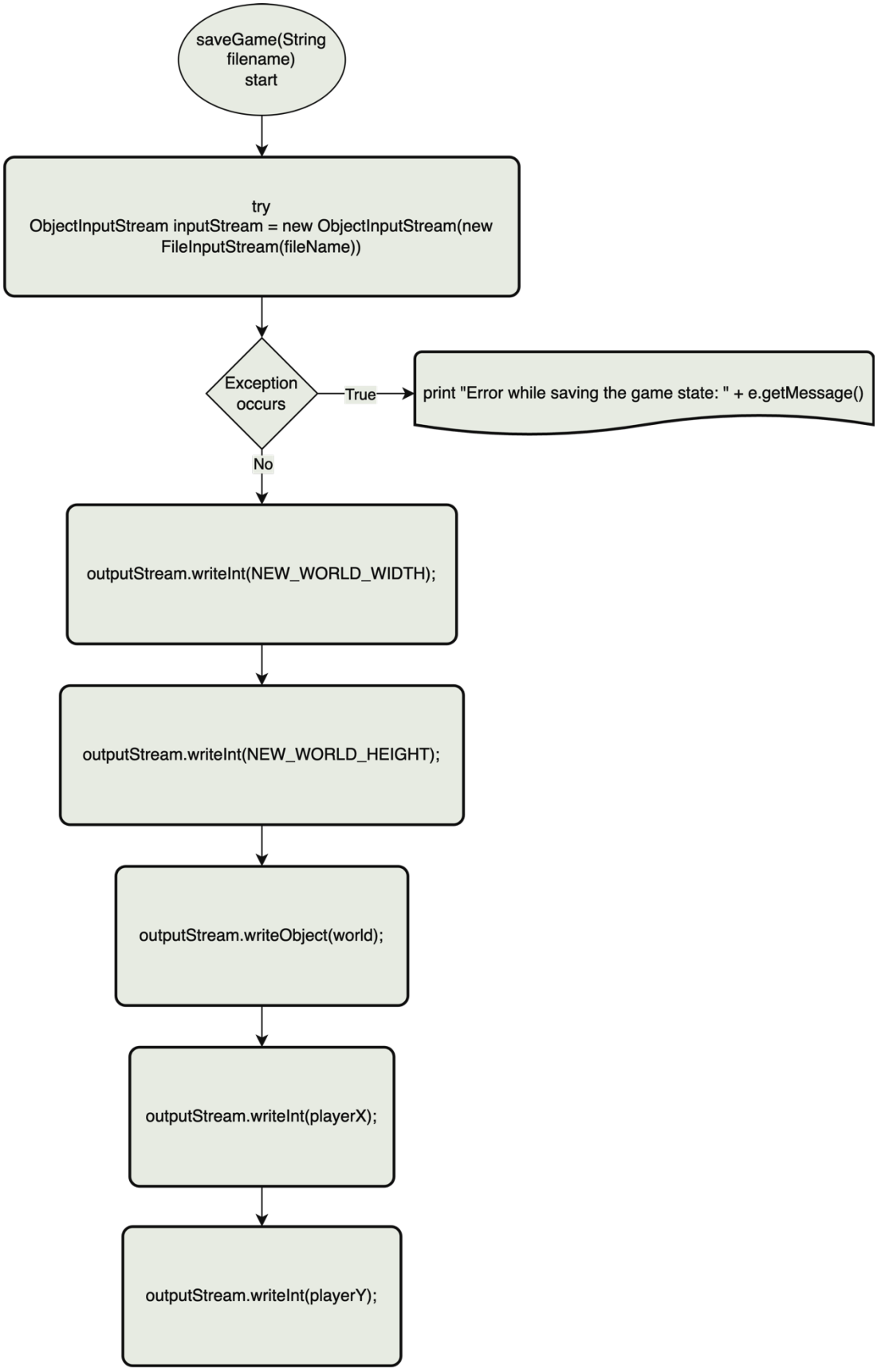



```
Pseudocodes > 26-addCraftedItem
1   FUNCTION addCraftedItem
2       IF craftedItems equals to null THEN
3           NEW ArrayList craftedItems
4       ADD craftedItem to craftedItems
```

27 – interactWithWorld

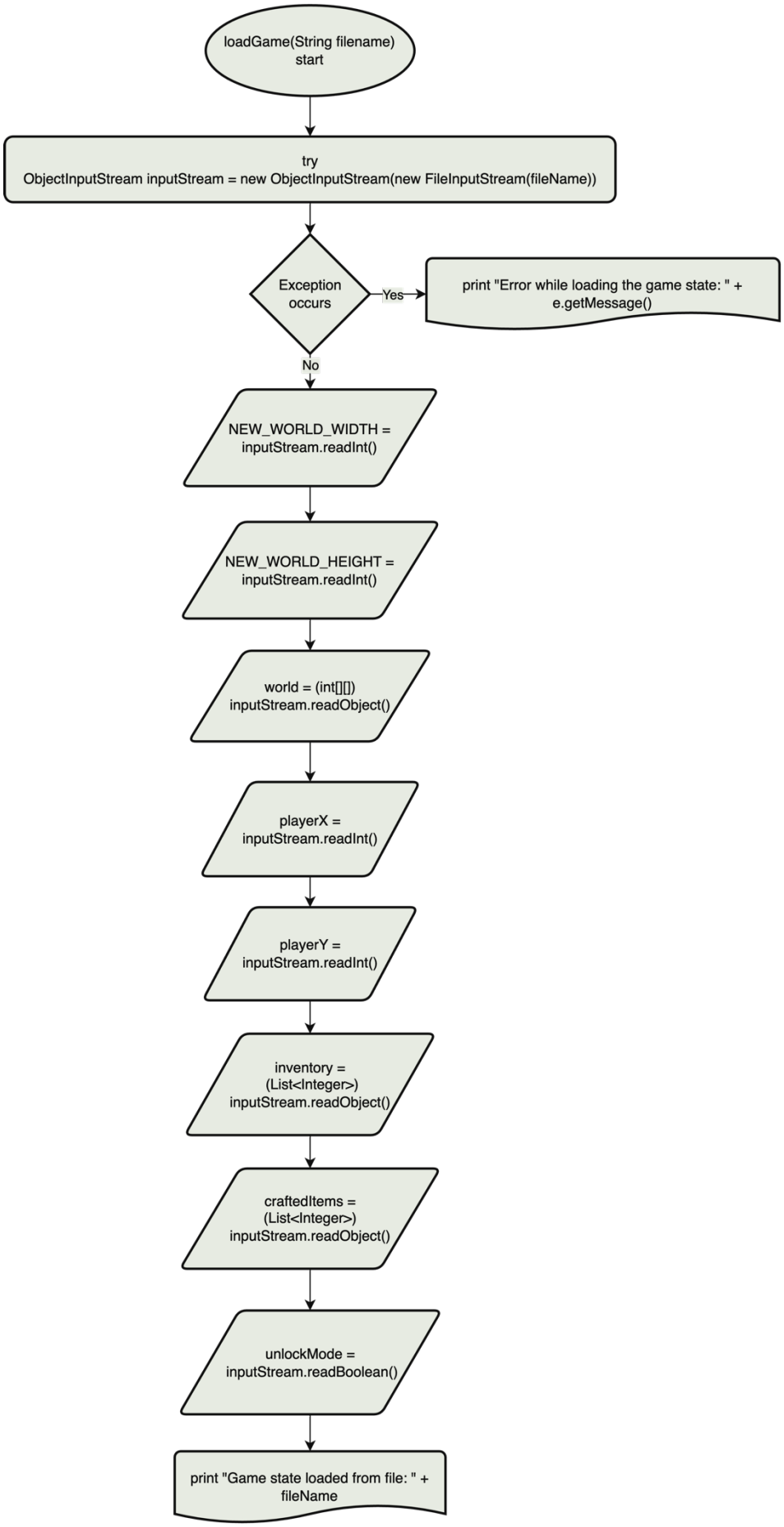


28 – saveGame


saveGame(String filename) start
try ObjectInputStream inputStream = new ObjectInputStream(new FileInputStream(fileName))
Exception occurs
True
print "Error while saving the game state: " + e.getMessage()
No
outputStream.writeInt(NEW_WORLD_WIDTH);
outputStream.writeInt(NEW_WORLD_HEIGHT);
outputStream.writeObject(world);
outputStream.writeInt(playerX);
outputStream.writeInt(playerY);

```
FUNCTION saveGame(string fileName)
    TRY NEW ObjectOutputStream(NEW FileOutputStream(fileName)) outputStream
        WRITE NEW_WORLD_WIDTH to outputStream
        WRITE NEW_WORLD_HEIGHT to outputStream
        WRITE world to outputStream
        WRITE playerX to outputStream
        WRITE playerY to outputStream
        WRITE inventory to outputStream
        WRITE craftedItems to outputStream
        WRITE unlockMode to outputStream
        PRINT "Game state saved to file: " + fileName
    CATCH
        PRINT "Error while saving the game state: " + error message
    waitForEnter()
```

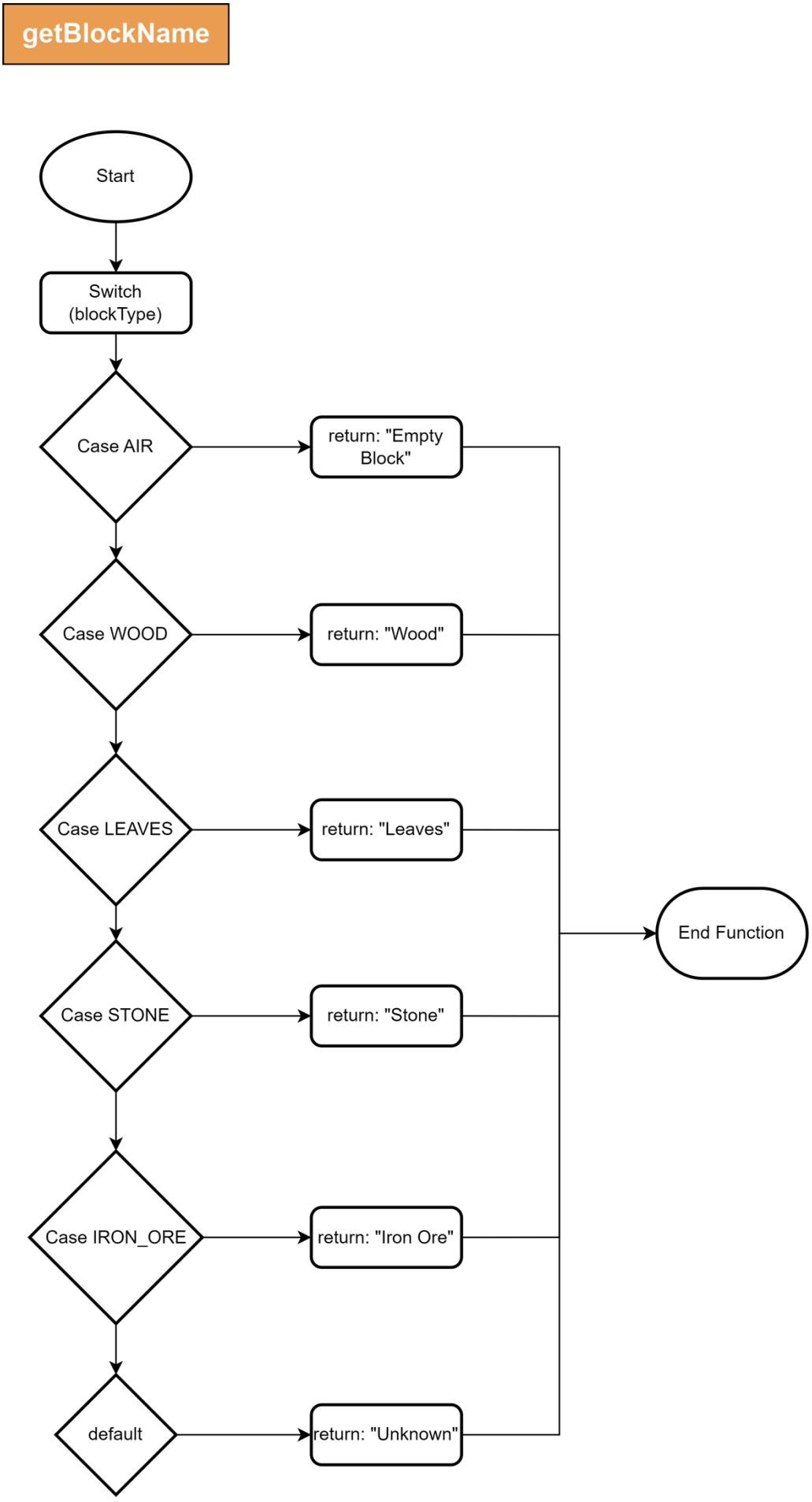
29 – loadGame

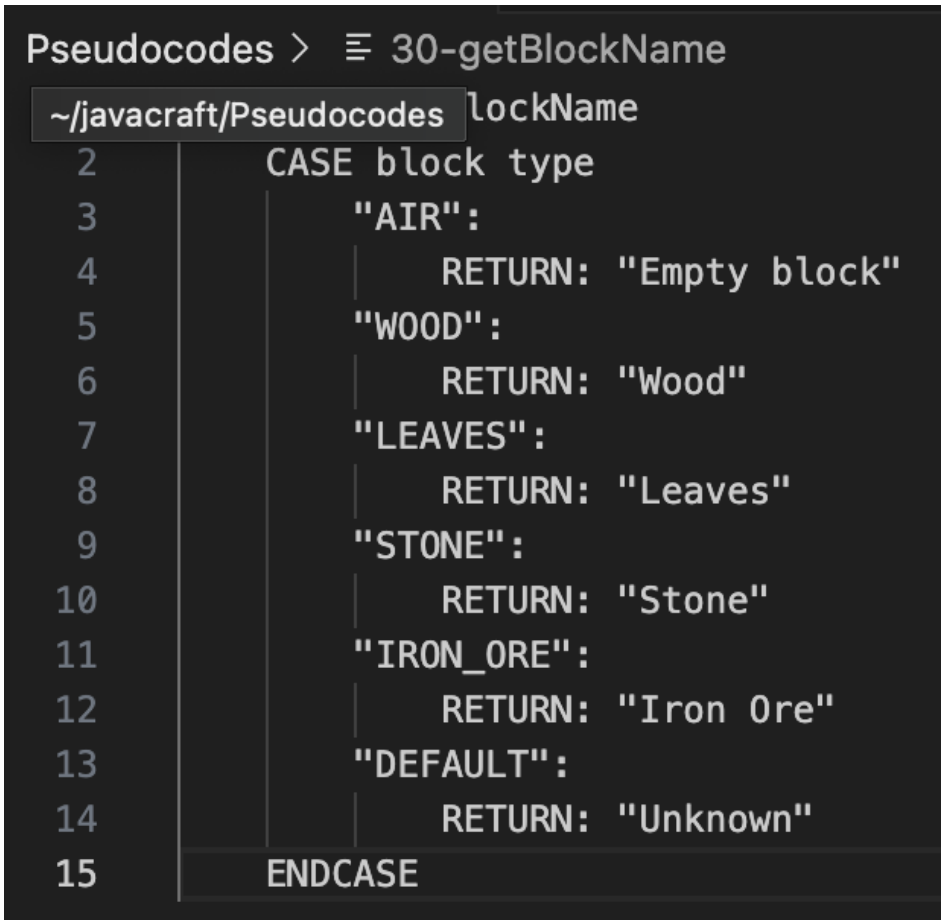

loadGame(String filename)
start
try
ObjectInputStream inputStream = new ObjectInputStream(new FileInputStream(fileName))
Exception occurs
Yes
print "Error while loading the game state: " + e.getMessage()
No
NEW_WORLD_WIDTH = inputStream.readInt()
NEW_WORLD_HEIGHT = inputStream.readInt()
world = (int[][]) inputStream.readObject()
playerX = inputStream.readInt()
playerY = inputStream.readInt()
inventory = (List<Integer>) inputStream.readObject()
craftedItems = (List<Integer>) inputStream.readObject()
unlockMode = inputStream.readBoolean()
print "Game state loaded from file: " + fileName

Pseudocodes > 29-loadGame

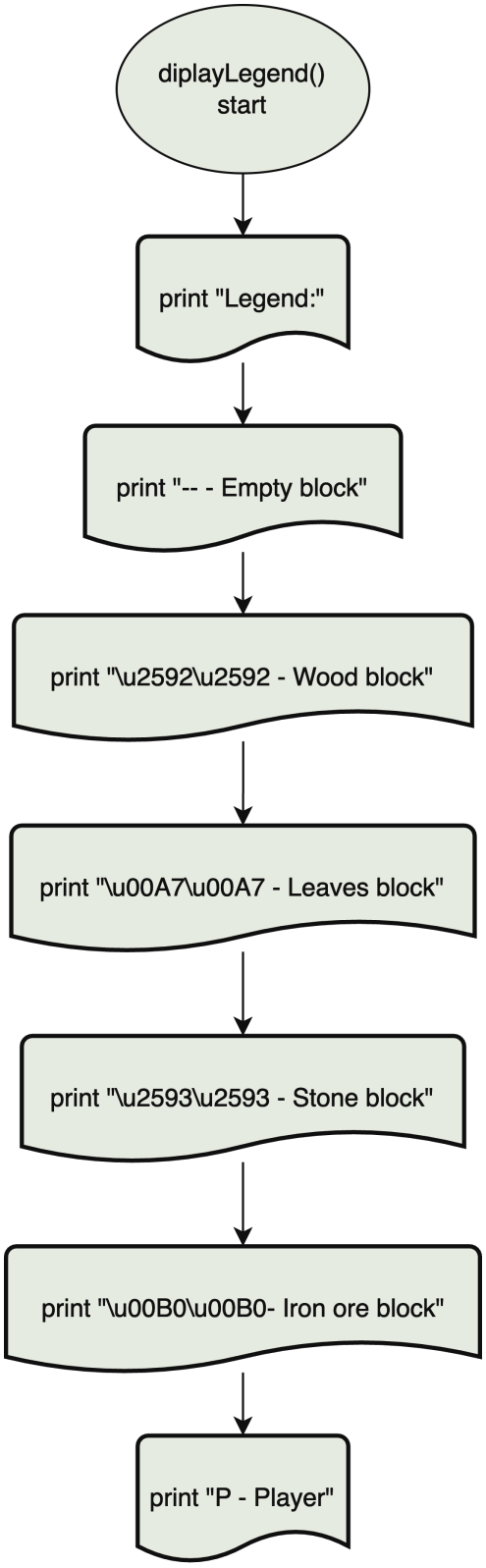
```
FUNCTION loadGame(string fileName)
    TRY NEW ObjectInputStream(NEW FileInputStream(fileName)) inputStream
        SET NEW_WORLD_WIDTH to READ inputStream
        SET NEW_WORLD_HEIGHT to READ inputStream
        SET world to READ inputStream
        SET playerX to READ inputStream
        SET playerY to READ inputStream
        SET inventory to READ inputStream
        SET craftedItems to READ inputStream
        SET unlockMode to REAF inputStream
        PRINT "Game state loaded from file: " + fileName
    CATCH
        PRINT "Error while loading the game state: " + error message
    waitForEnter()
```

30 – getBlockName




Pseudocodes > 30-getBlockName
~/javacraft/Pseudocodes
lockName
CASE block type
"AIR":
RETURN: "Empty block"
"WOOD":
RETURN: "Wood"
"LEAVES":
RETURN: "Leaves"
"STONE":
RETURN: "Stone"
"IRON_ORE":
RETURN: "Iron Ore"
"DEFAULT":
RETURN: "Unknown"
ENDCASE

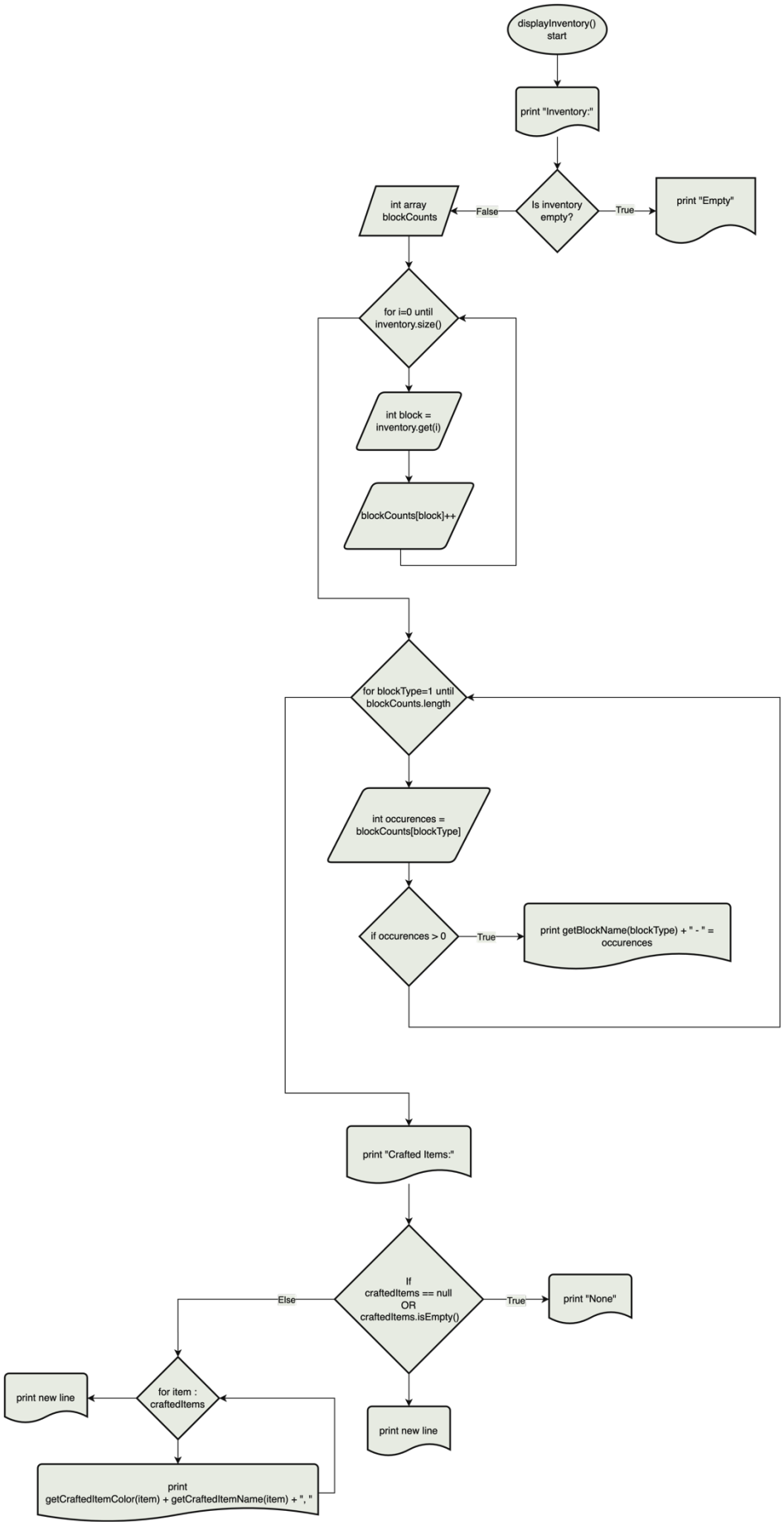
31 – displayLegend



Pseudocodes > 31-displayLegend

```
1   FUNCTION displayLegend
2       PRINT "Legend:" in blue
3       PRINT "-- - Empty block" in white
4       PRINT "▒▒ - Wood block" in red
5       PRINT "§§ - Leaves block" in green
6       PRINT "▒▒ - Stone block" in blue
7       PRINT "°°- Iron ore block" in white
8       PRINT "P - Player" in blue
```

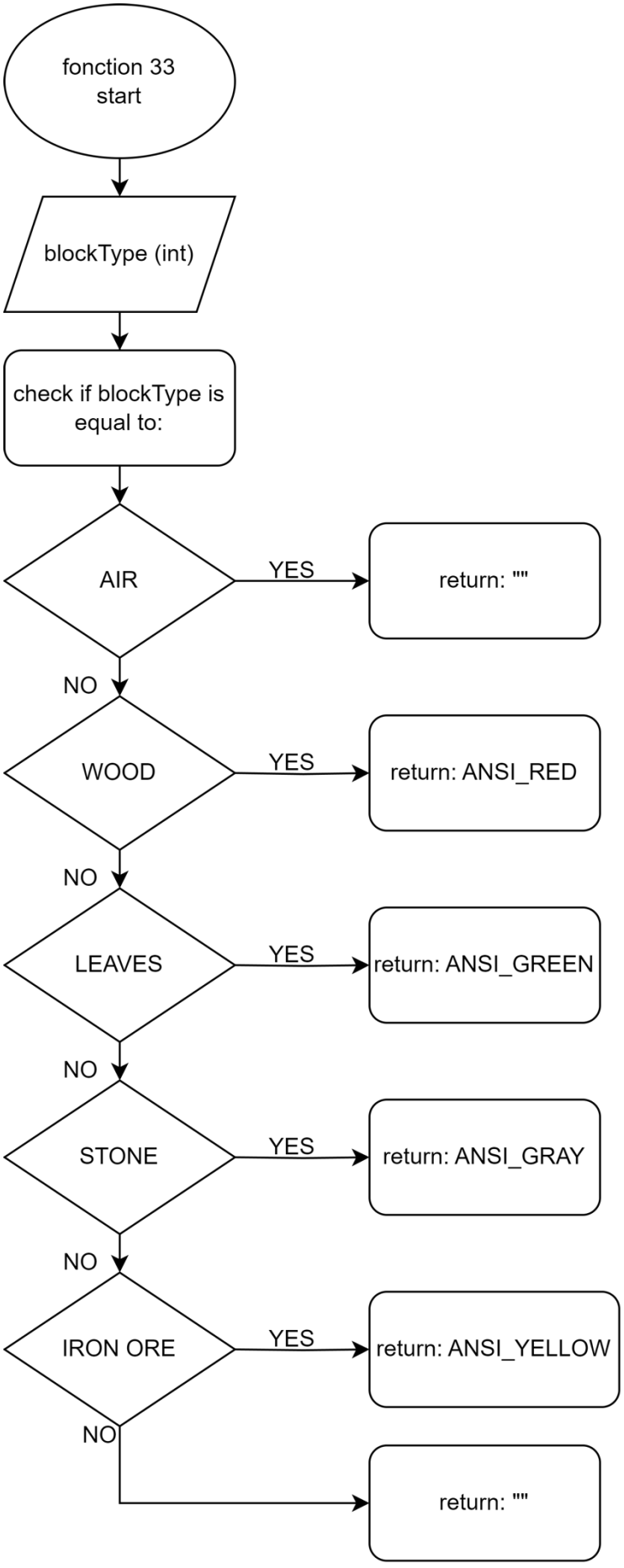
32 – displayInventory



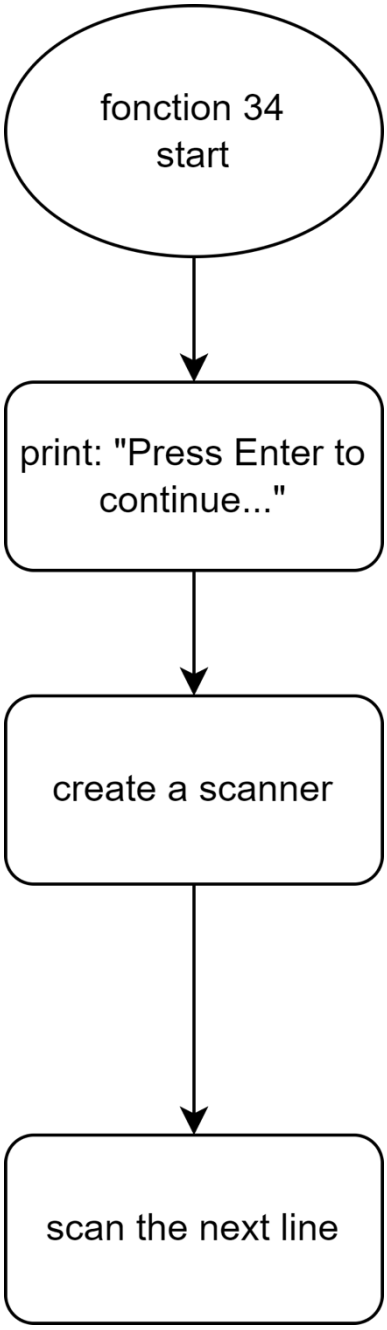
Pseudocodes > 32-displayInventory

```
FUNCTION displayInventory
    PRINT "Inventory:"
    IF inventory is empty THEN
        PRINT "Empty" in yellow
    ELSE
        SET blockCounts to NEW integer array[5]
        FOR each i in inventory
            SET block to GET inventory(i)
            INCREMENT blockCounts[block]
        FOR each blockType in blockCounts
            SET occurrences to blockCounts[blockType]
            IF occurrences bigger than 0 THEN
                PRINT getBlockName(blockType) + " - " + occurrences
    PRINT "Crafted Items:"
    IF craftedItems equals to null THEN
        PRINT "None" in yellow
    ELSE
        FOR each item in craftedItems
            PRINT getCraftedItemColor(item) + getCraftedItemName(item) + ", "
        PRINT NEW LINE
    PRINT NEW LINE
```

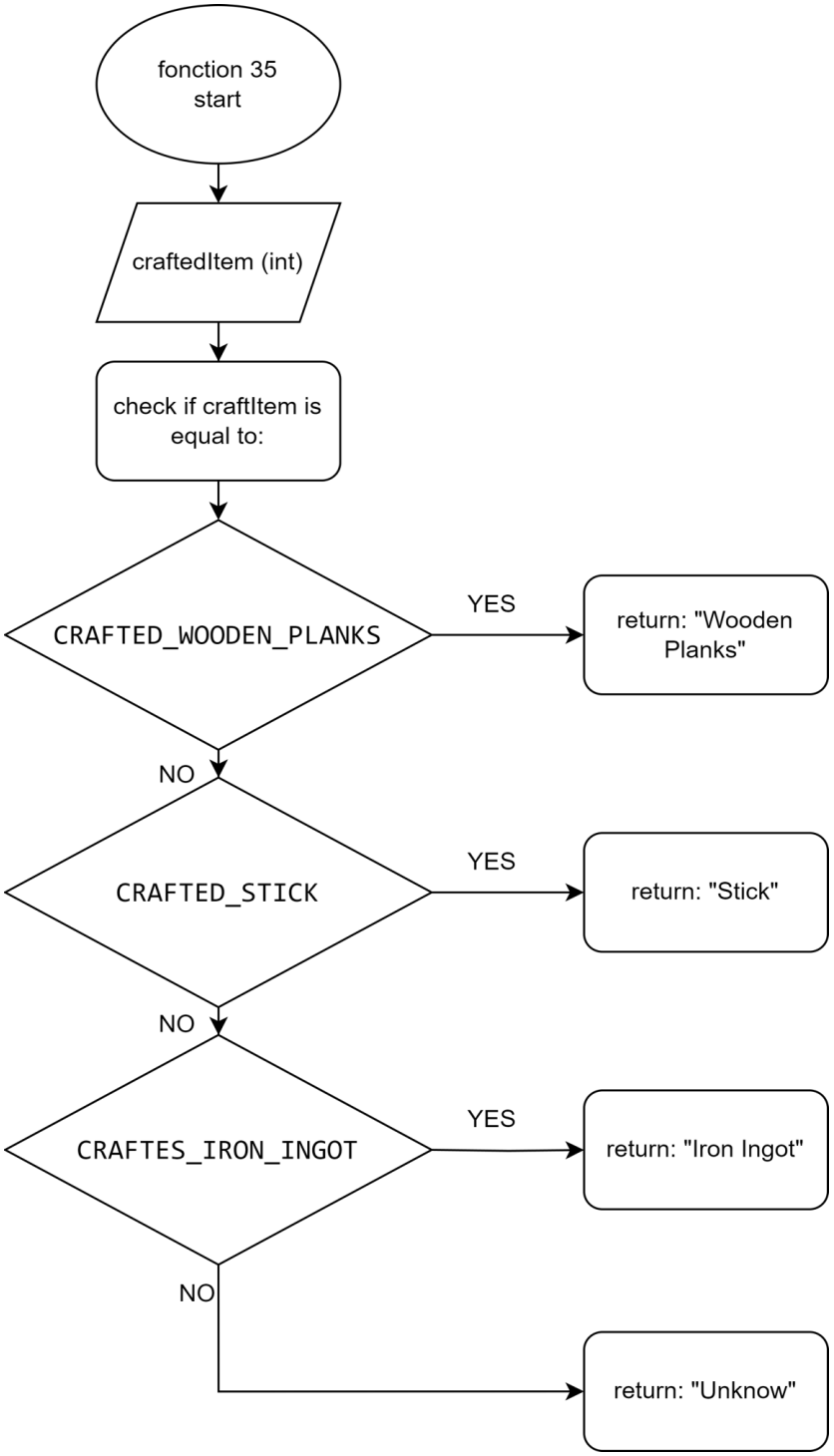
33 – getBlockColor



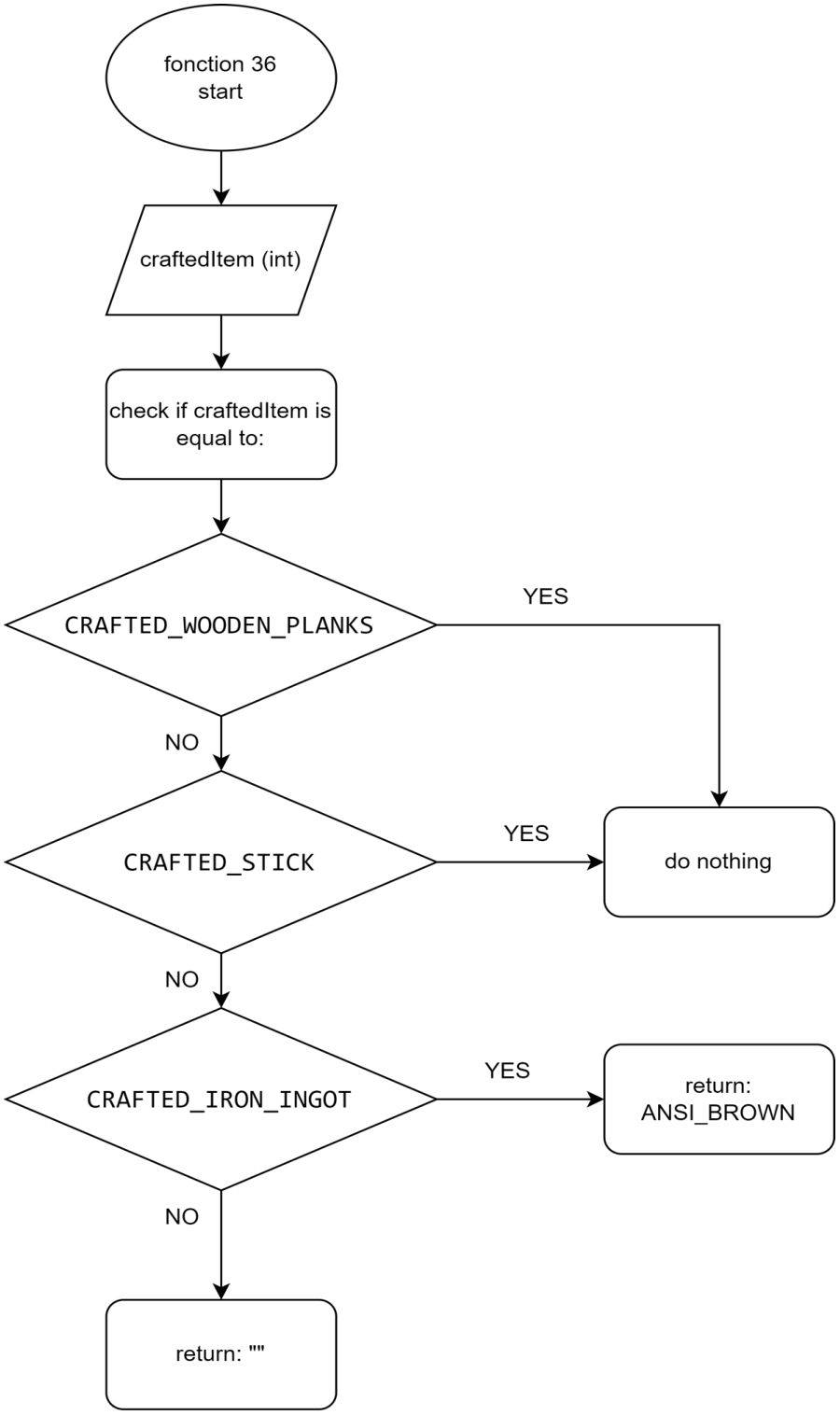
34 – waitForEnter



35 – getCraftedItemName



36 – getCraftedItemColor



Flowchart for the game: