Maastricht University

JavaCraft Project

BCS1110, Introduction to Computer Science

Group 11

Full Name	Student ID
Long Luong	I6359380
Élisa Donéa	I6356213
Chris Munteanu	I6344912
Alexia Raportaru	I6355814

Professors: Dr. Ashish Sai, Dr. Thomas Bitterman

Maastricht, October 8, 2023

Contents

1 | Introduction

JavaCraft is a a multifaceted text-based Java game inspired by Minecraft. The game is a relatively complex Java program that brings over 35 functions to create a diverse gameplay experience. This team project is an academic exercise in computer science, logical thinking, and collaboration. Working in teams of four, we used our creativity, analytical skills, and technical skills to analyse and expand upon the existing JavaCraft game. The source code of JavaCraft can be found in the following link: https://gitlab.maastrichtuniversity.nl/bcs1110/javacraft/-/raw/main/JavaCraft.java

Below is a table that describes what each team member did on the project, divided by section:

Section	Who did what
Introduction	Long (100%)
JavaCraft's Workflow	Alexia (100%)
Functionality Exploration	Long (60%), Élisa (40%)
Secret Door FSA Design	Élisa (50%), Chris (50%)
Git Collaboration & Version Control	Alexia (25%), Élisa (25%), Chris (25%), Long (25%)
Pseudocode & flowcharts for 15 different functions	Alexia (31%), Élisa (31%), Chris (31%), Long (7%)
Report	Long (100%)

2 | JavaCraft's Workflow

In order to fully understand the mechanism of the game, a flowchart of the entire game is provided below. The flowchart is accompanied by pseudocode.

start

generate world

program greets player and displays action commands

start game? — false → exit program → stop

start game? — true → game starts

program displays world

display inventory, action commands

this shape is representative for a loop, after a command is succesfully operated (e.g. mine a block) the program loops back to this

correct action commands entered? — true

('w' || 'up' || 's' || 'down' || 'a' || 'left' || 'd' || 'right') entered? — true → is unlockMode true? — true → movementCommandEntered becomes true → move player according to movement command entered

'm' entered? — true → is unlockMode true? — true → miningCommandEntered becomes true → mine block that player is currently on

'p' entered? — true → displayInventory method is called → is inventory empty? — false → display blocks present in inventory / — true → Empty

program asks for the block type to place → correct block type entered? — true → program places block type entered on the player's position

'c' entered? — true → is unlockMode true? — true → craftingCommandEntered becomes true

displays crafting recipes → program asks for the recipe number to craft → correct recipe number entered? — true → are required items currently in the player's inventory? — true → craft item and add to inventory

'i' entered? — true → player interacts with the block they are currently on → add block type the player is currently on to inventory

'save' entered? — true → program asks for the file name to save the game state and saves the game state under that name

'load' entered? — true → program asks for the name of the file to load → is there a loadable file with that name? — true → game loads the file / — false → Error while loading the game state

'exit' entered? — true → exits program → stop

'look' entered? — true → displays the blocks surrounding the player

'unlock' entered? — true → unlockMode becomes true

'getflag' entered? — true → program tries to get flag from server → succesfully gets flag? — true → aquired flag from server / — false → Error connecting to the server

'open' entered? — true → Has the player entered crafting, mining and movement commands before? — true → secretDoorUnlocked becomes true → program resets world and loads the flag acquired in the getCountryAndQuoteFromServer method → add 100 of each non-craftable items to player's inventory / — false → Invalid passkey. Try again!

false → Invalid input. Please try again!

```
generate world
print welcome message and instructions
```

```
ask player if they want to start the game
if choice is equal to 'y' then start game
   clear screen
   display world

   loop
   display inventory, display legend

      if input is 'wasd' then move player accordingly
         if unlockMode is true then movementCommandEntered becomes true

      else if input is 'm' then mine the block the player is currently standing on
         if unlockMode is true then miningCommandEntered becomes true

      else if input is 'p' then function displayInventory and ask player which block they want
         to place                                    if inventory is empty display "Empty"
       else display inventory
         if input is correct then place block on the player's current position
         else inform player they do not have that specific block in their inventory

      else if input is 'c' then display crafting recipes
         ask for recipe number in order to craft
            if input is correct then check whether the player has the items necessary for the
               crafting recipe
               if the player has the necessary items then proceed crafting and add the item to
                  the player's inventory
               else inform the player they have insufficient resources to craft
         if unlockMode is true then craftinCommandEntered becomes true

      else if input is 'i' then player interacts with the block they are currently standing on
         if the block they are currently on is not air then add the item to their inventory

      else if input is 'save' then ask player for a name for the current state file they want
         to save
         save file

      else if input is 'load' then ask player for the name of the file they want to load
         if there is a file with that name then load file
         else inform player there has been an error loading the game state

      else if input is 'exit' then exit program

      else if input is 'look' then inform the player of the block types surrounding them

      else if input is 'unlock' then unlockMode becomes true

      else if input is 'getflag' then try to get flag from the server
         if unsuccesful then inform the player that there has been an error connecting to the
            server

      else if input is 'open' then check if the booleans craftingCommandEntered,
         movementCommandEntered, miningCommandEntered true
         if they are true then secretDoorUnlocked becomes true
            reset world
            load flag world
            add 100 of leaves, stone, iron ore and wood to player's inventory

      else inform player that their input is invalid
   end if
else print "Game not started. Goodbye!" and exit program
end if
```

3 | Functionality Exploration

This section describes several functions found in the JavaCraft.java file. Out of the functions, there are fifteen who have a flowchart and pseudocode.

Table 3.1: A table that describes functions used in javacraft

No.	Function Name	Description
1	`void generateWorld`	assigns integer to every tile of the world
2	`void initGame`	creates world with width `worldWidth` and height `worldHeight`
3	`void main`	main function
4	`void startGame`	starts the game
5	`void movePlayer`	moves player horizontally or vertically
6	`void mineBlock`	mines block player is on if block is not air
7	`String getBlockSymbol`	returns symbol of `blockType`
8	`void resetWorld`	clears the world and sets player position in middle
9	`void generateEmptyWorld`	generates an empty world
10	`void clearScreen`	clears terminal
11	`void lookAround`	prints out adjacent squares to player
12	`void fillInventory`	completely fills up inventory of player
13	`void displayLegend`	displays a legend of what each tile represents
14	`void displayWorld`	prints out all tiles of the world
15	`void displayInventory`	prints out obtained items & crafted items
16	`void loadGame`	loads the game from file `fileName`
17	`void saveGame`	saves the game in file `fileName`
18	`void interactWithWorld`	interacts with item player is standing on
19	`void addCraftedItem`	adds item `craftedItem` to array `craftedItems`
20	`void removeItemsFromInventory`	removes item `item` `count` times from inventory
21	`boolean inventoryContains`	returns boolean of whether `item` is in inventory
22	`void placeBlock`	places block `blockType` at player position
23	`void displayCraftingRecipes`	prints out available crafting recipes
24	`void craftItem`	crafts an item based on argument `recipe`
25	`void craftIronIngot`	crafts an iron ingot
26	`void craftStick`	crafts a stick
27	`void waitForEnter`	waits for operator to press Enter
28	`String getBlockTypeFromCraftedItem`	returns integer of `craftedItem`
29	`String getCraftedItemFromBlockType`	returns integer of `blockType`
30	`String getBlockName`	returns name of `blockType`
31	`String getBlockColor`	returns color of `blockType`
32	`String getCraftedItemName`	returns name of `craftedItem`
33	`String getCraftedItemColor`	returns color of `craftedItem`
34	`char getBlockChar`	returns char of `blockType`
35	`void craftWoodenPlanks`	crafts wooden planks
36	`void getCountryAndQuoteFromServer`	makes HTTP request and writes data to server and prints country and quote

4 | Finite State Automata (FSA) Design

The following describes the Secret Door Logic through an FSA illustration and a description:
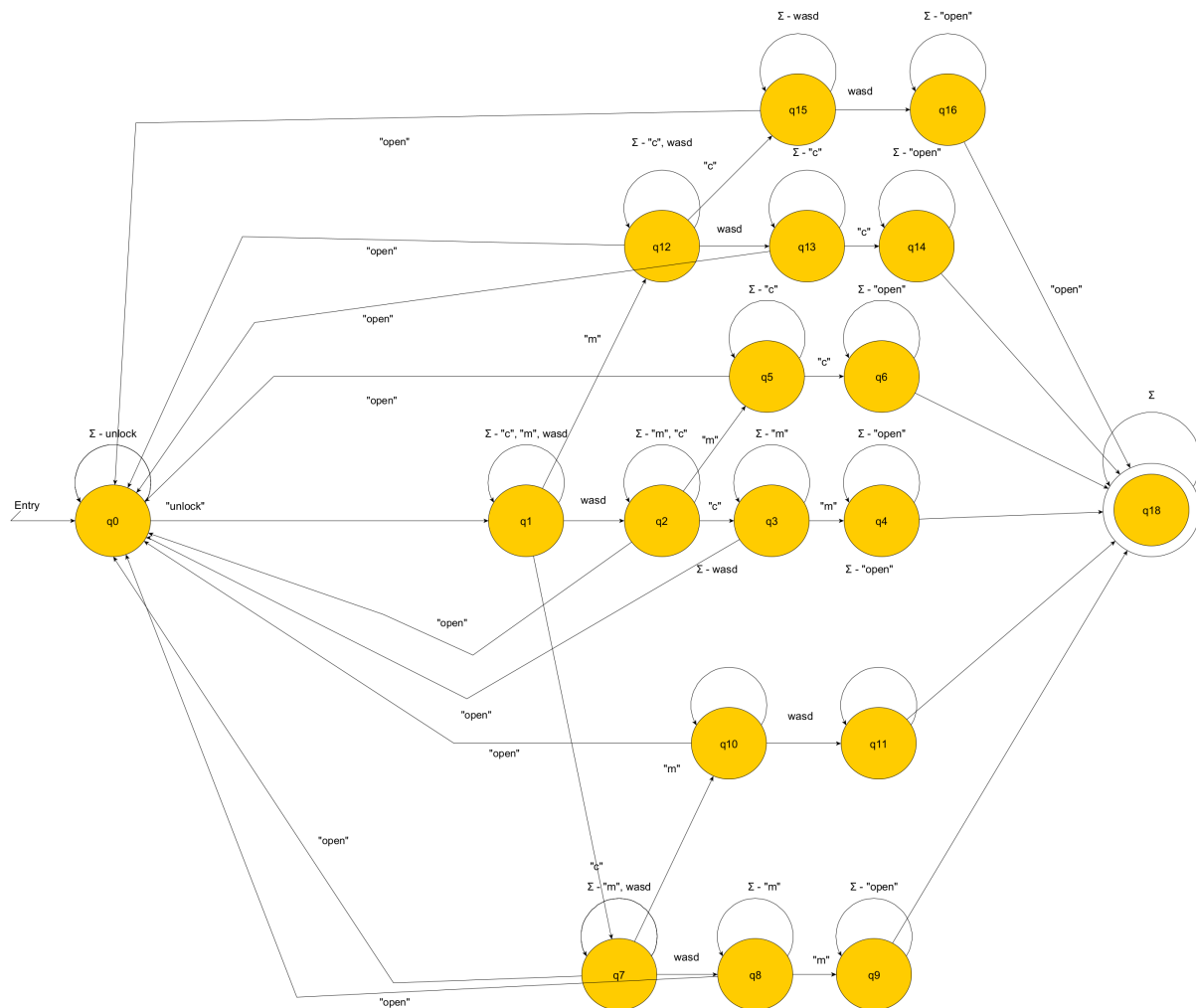
We have 18 finite set of states and in our alphabet we have "open" , "wasd", "c", "m" and "unlock" On step one Q0 you have to write unlock if not you will remain in Q0 After you have arrived in Q1 you have the choice in between wasd, c, m because in able to unlock the secret door you must c, craft then m, mine or wasd to move in any order that's why after Q1 it divides in 3 because the order is not important. If you type a move command (w,a,s or d) you enter Q2, the only options you have from now are mine or craft if not you stay in Q2 unless you type open which sends you back to Q0 and have to start again. If you type c you go into Q3 so the last thing you have to write is mine if you write anything else you stay in Q3 expect "open" which sends you in Q0 after mining you enter Q4. From there your only option is to "open" which unlocks the secret door. Now because the order doesn't matter after Q2 you can also type m for mine which sends you in Q5 where the only option is c, "craft" if you type anything else you go back to Q0 after you type C you entre Q6 where you can only type "open" which will unlock the secret door. We have created 2 other branches based on the same principal so we can cover all the possible ways you can acces the secret door

Note: in our alphabet "wasd" stands for "w", "a", "s" or "d"

5 | Git Collaboration & Version Control

The link to the JavaCraft branch can be found here: https://gitlab.maastrichtuniversity.nl/bcs1110/javacraft/-/tree/group11

All the files of the project can be found in the repository.
Everything was done on one branch called group11. There were no conflicts during the process.

6 | References

A | Appendix: pseudocode and flowcharts

This appendix provides the full blocks of pseudocode and its flowcharts.
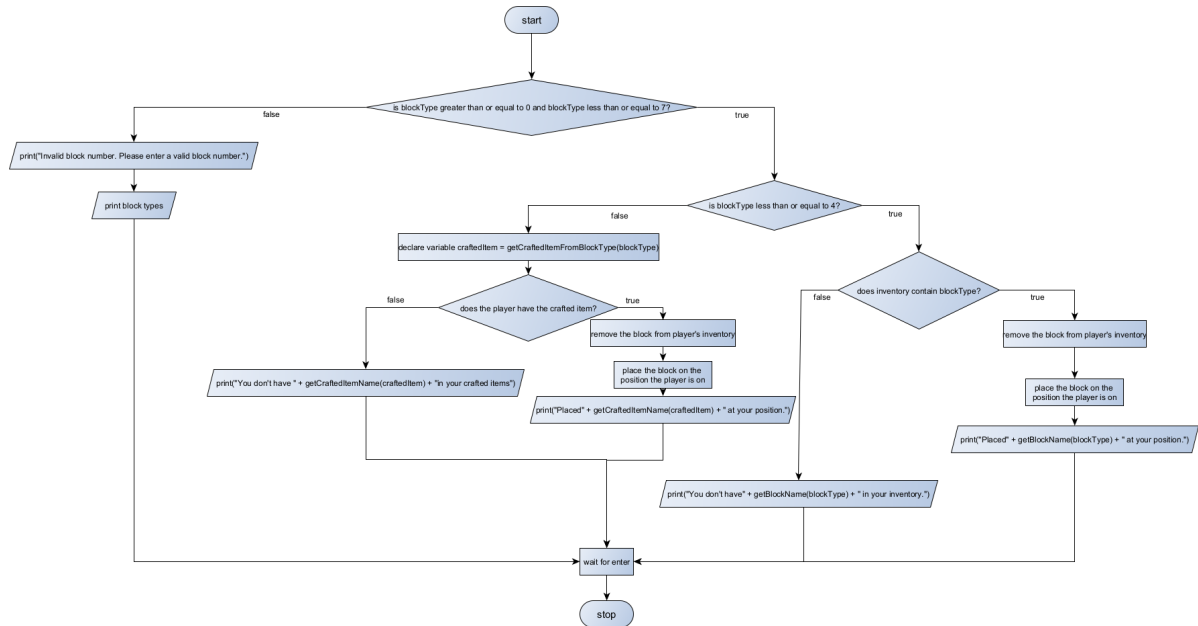

```
function placeBlock(blockType)

if blockType is greater than or equal to 0 and blockType is less than or equal to 7 then
    if blockType less than or equal to 4 then
        if inventory contains blockType then
            inventory.remove(blockType)
            world[playerX][playerY] = blockType
            print("Placed" + getBlockName(blockType) + " at your position.")

        else print("You don't have " + getBlockName(blockType) + " in your inventory")
        end if
    else
        craftedItem = getCraftedItemFromBlockType(blockType)
        if craftedItems contains craftedItem then
            craftedItems.remove(craftedItem)
            world[playerX][playerY] = blockType
            print("Placed" + getCraftedItemName(craftedItem)+ " at your position.")
        else print("You don't have " + getCraftedItemName() + " in your inventory.")
        end if
    end if
else print("Invalid block number. Please enter a valid block number.")
print(BLOCK_NUMBERS_INFO)
end if
waitForEnter()
end function
```
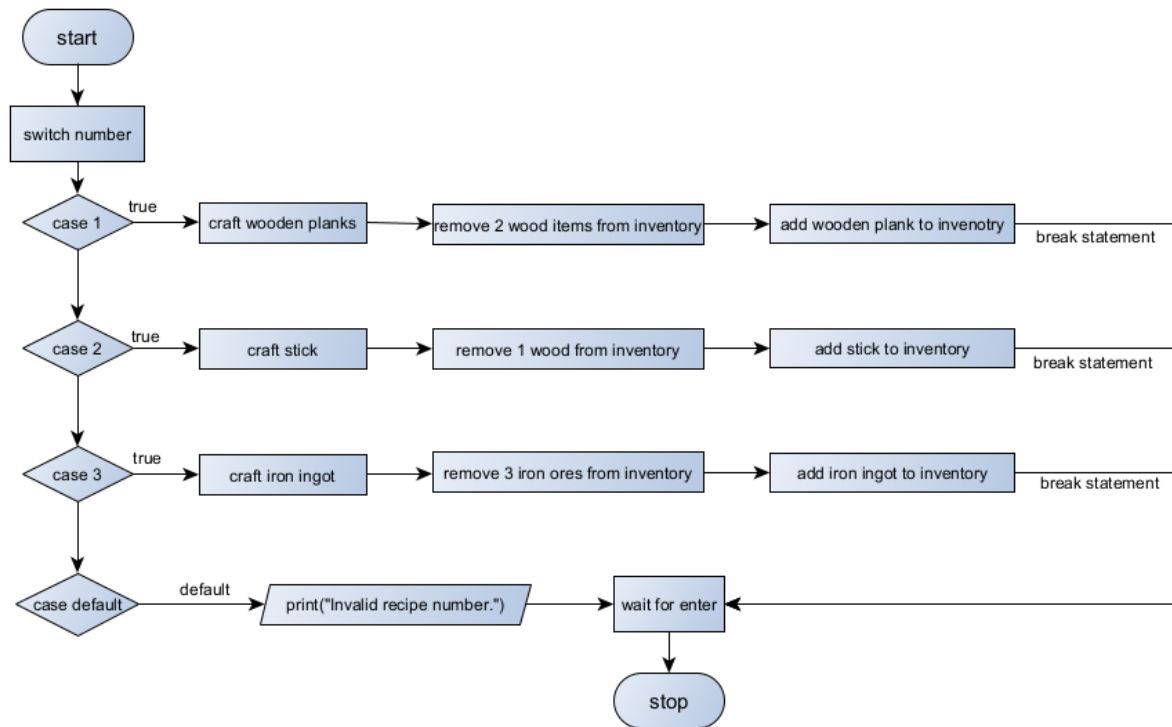
```
function craftItem(recipe)

switch(recipe)
case 1:
   craftWoodenPlanks()
end if
case2:
   craftStick()
end if
case3:
   craftIronIngot()
end if
default:
   print("Invalid recipe number.")
end switch
waitForEnter()
end function
```
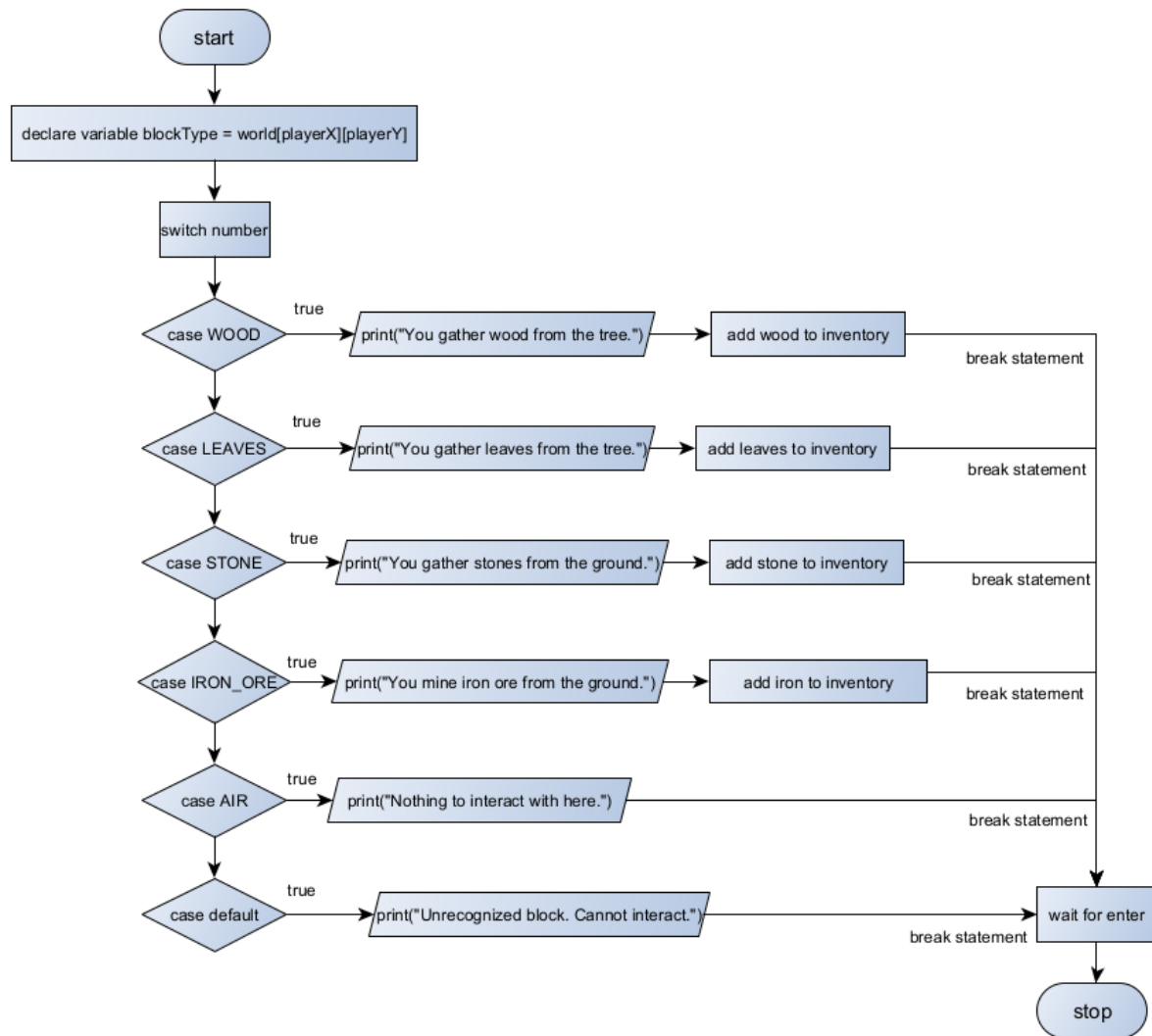
```
function interactWithWorld()

blockType = world[playerX][playerY]
switch(blockType):
case WOOD:
   print("You gather wood from the tree.")
   inventory.add(WOOD)
end if
case LEAVES:
   print("You gather leaves from the tree.")
   inventory.add(LEAVES)
end if
case STONE:
   print("You gather stones from the ground.")
   inventory.add(STONE)
end if
case IRON_ORE:
   print("You mine iron ore from the ground.")
   inventory.add(IRON_ORE)
end if
case AIR:
   print("Nothing to interact with here.")
end if
default: print("Unrecognized block. Cannot interact.")
end switch
```

```
waitForEnter()
end function
```
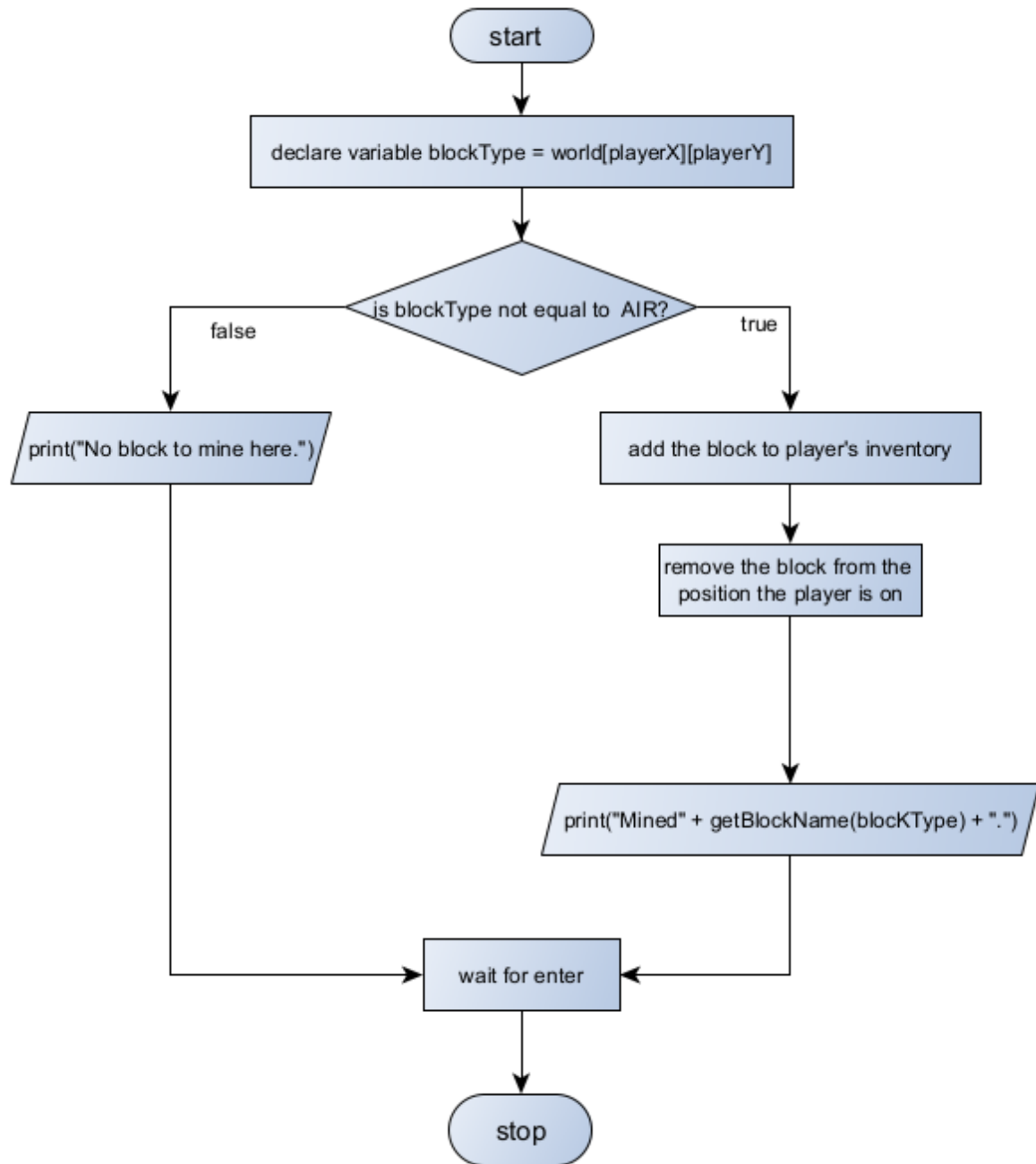
```
function mineBlock()

blockType = world[playerX][playerY]
if blockType is not equal to AIR then
    inventory.add(blockType)
    world[playerX][playerY] = AIR
    print("Mined " + getBlockName(blockType) + ".")
else print("No block to mine here.")
end if
waitForEnter()
end function
```
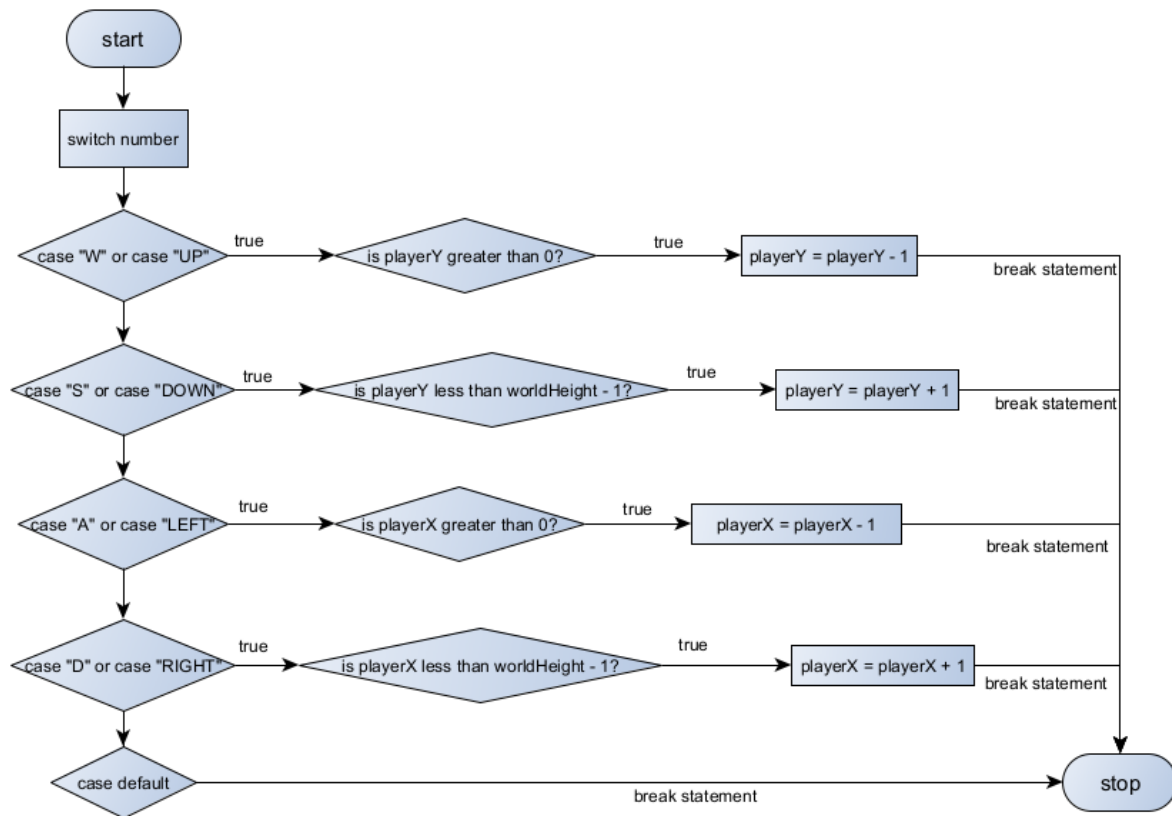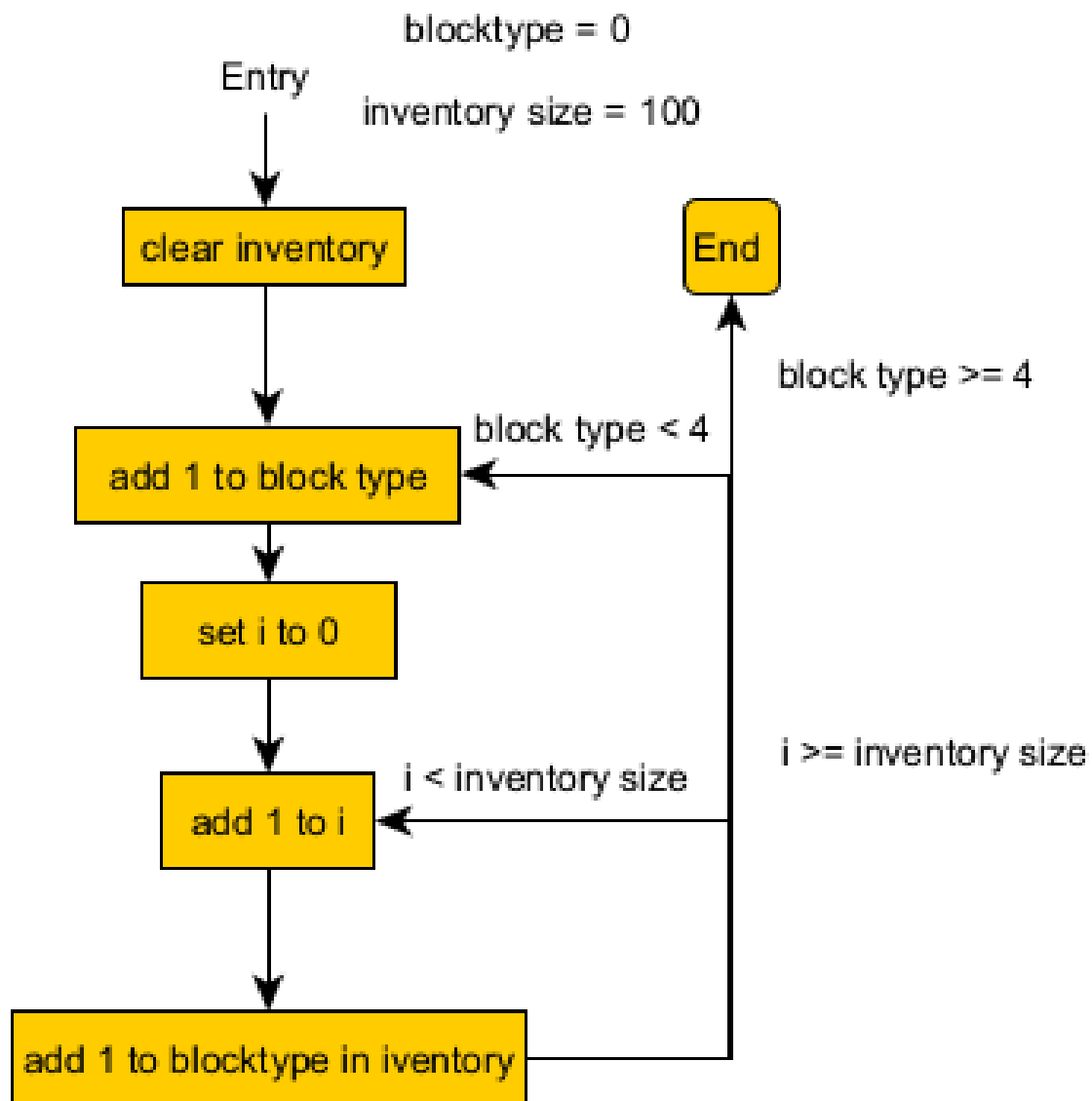
```
function movePlayer(direction):

direction = uppercase(direction) //converts direction to uppercase for consistency
switch(direction):
case "W" or "UP":
   if playerY > 0 then playerY = playerY - 1
   end if
case "S" or "DOWN":
   if playerY < worldHeight - 1 then playerY = playerY + 1
   end if
case "A" or "LEFT":
   if playerX > 0 then playerX = playerX - 1
   end if
case "D" or "RIGHT":
   if playerX < worldWidth - 1 then playerX = playerX + 1
   end if
default: //do nothing
end switch
end function
```
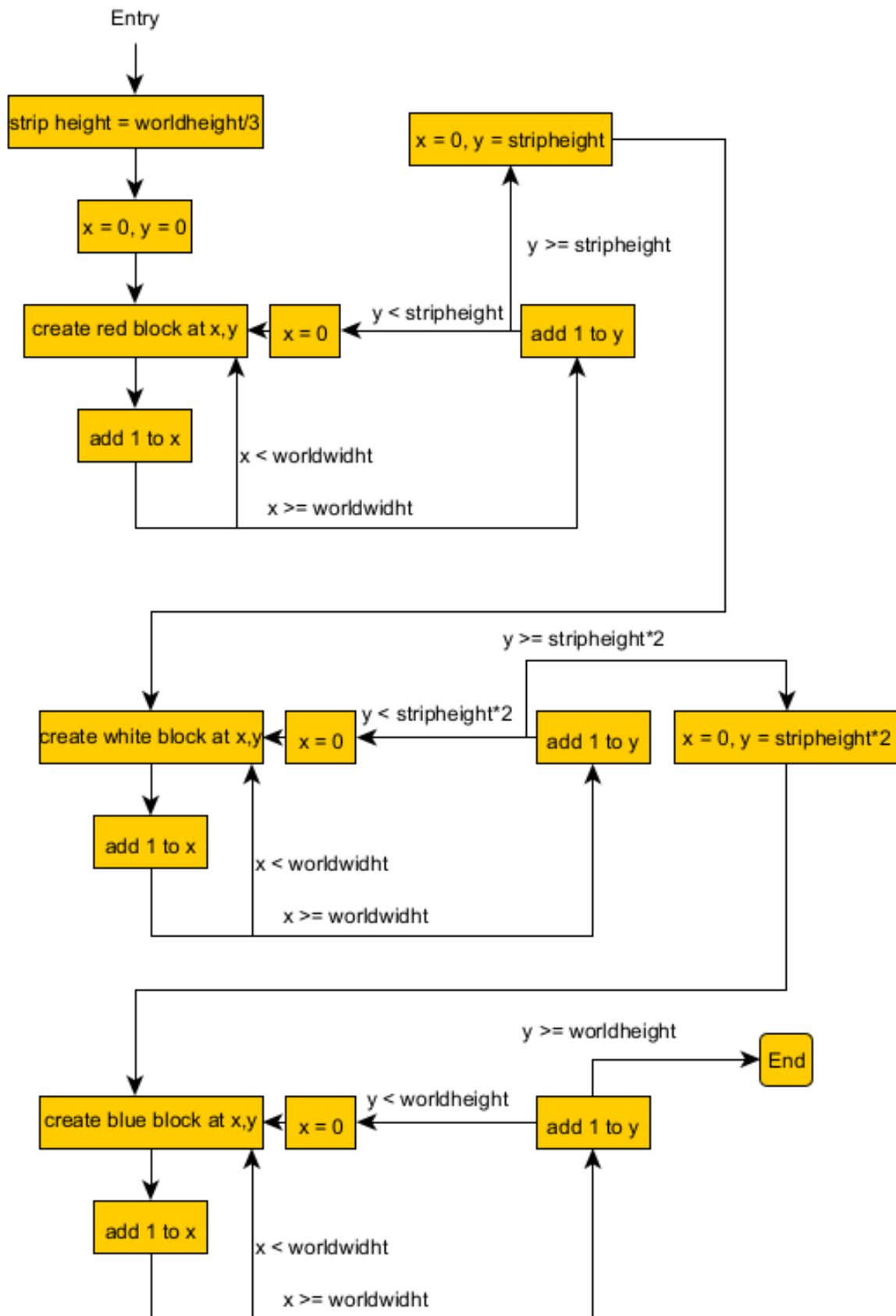
blocktype = 0

Entry

inventory size = 100

clear inventory

End

block type >= 4

block type < 4

add 1 to block type

set i to 0

i < inventory size

i >= inventory size

add 1 to i

add 1 to blocktype in iventory

```
fillInventory()
Set INVENTORY_SIZE = 100

Clear inventory
FOR blockType = 0; blockType <= 4; blockType ++:
   FOR i = 0; i < INVENTORY_SIZE; i++:
      add block blockType to inventory

end function
```


Entry

strip height = worldheight/3

x = 0, y = 0

create red block at x,y ◄ x = 0 ◄ y < stripheight ◄ add 1 to y

x = 0, y = stripheight

y >= stripheight

add 1 to x

x < worldwidht

x >= worldwidht

create white block at x,y ◄ x = 0 ◄ y < stripheight*2 ◄ add 1 to y ◄ x = 0, y = stripheight*2

y >= stripheight*2

add 1 to x

x < worldwidht

x >= worldwidht

create blue block at x,y ◄ x = 0 ◄ y < worldheight ◄ add 1 to y

y >= worldheight

End

add 1 to x

x < worldwidht

x >= worldwidht

`function generateEmptyWorld()`

```
Set 2d array world = new int[NEW_WORLD_WIDTH][NEW_WORLD_HEIGHT]
Set redBlock = 1
Set whiteBlock = 4
Set blueBlock = 3
Set stripHeight = NEW_WORLD_HEIGHT / 3

FOR y = 0; y < stripHeight; y++:
    FOR x = 0; x < NEW_WORLD_WIDTH; x++:
        world[x][y] = redBlock

FOR y = stripHeight; y < stripHeight * 2; y++:
    FOR x = 0; x < NEW_WORLD_WIDTH; x++:
        world[x][y] = whiteBlock

FOR y = stripHeight * 2; y < NEW_WORLD_HEIGHT; y++:
    FOR x = 0; x < NEW_WORLD_WIDTH; x++:
        world[x][y] = blueBlock
end function
```
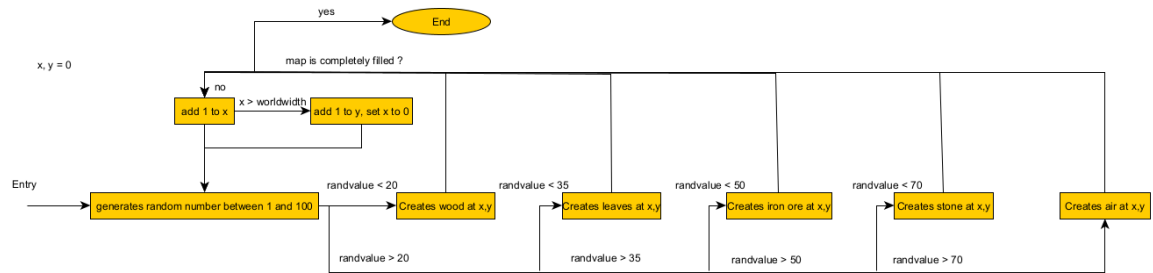
```
function generateWorld()

FOR y = 0; y < WORLD_HEIGHT; y++:
  FOR x = 0; x < WORLD_WIDTH; x++:
    creates random number between 1 and 100
    if random number < 20
      creates wood at x, y
    else if random number < 35
      creates leaves at x, y
    else if random number < 50
      creates stone at x, y
    else if random number < 20
      creates iron ore at x, y
    else create air at x, y
end function
```
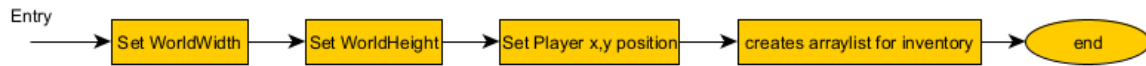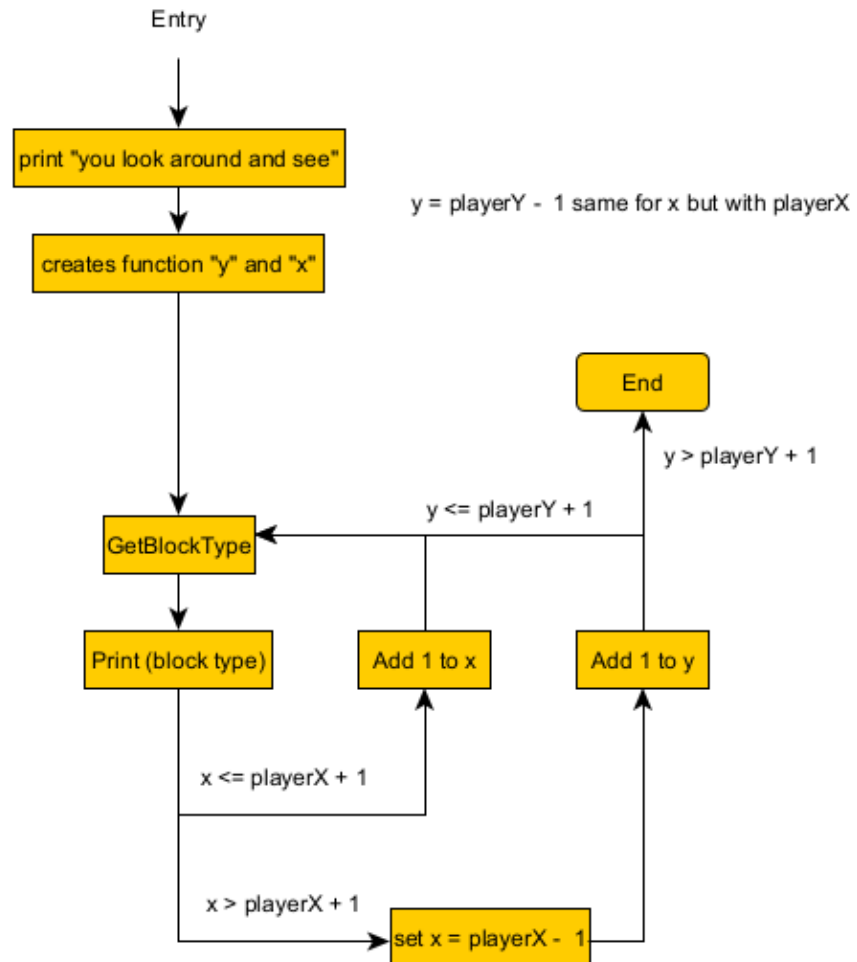
```
function initGame()

Set worldwidth
Set worldheight
Set world = [worldwidht][worldheight]
Set playerx = worldwidght / 2
Set playery = worldheight / 2
Creates array list inventory
```
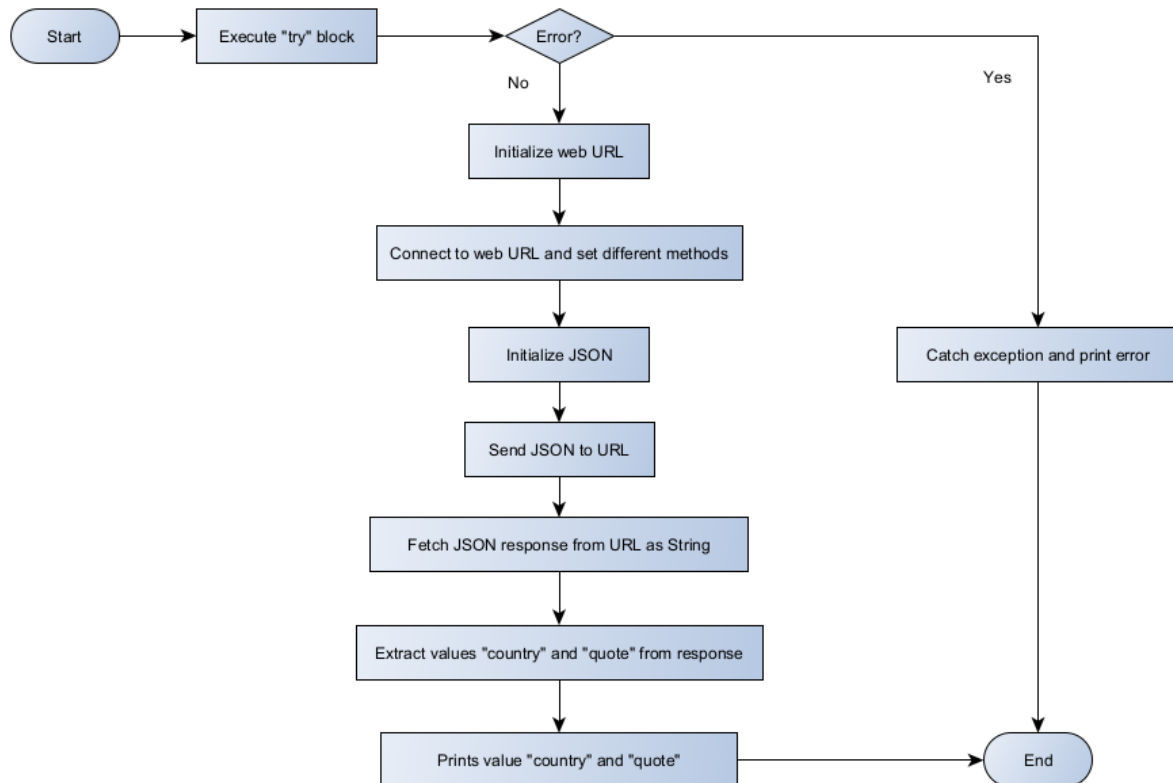
```
function lookAround()
Set playerX be x position of player
Set playerY be y position of player

print("You look around and see:")
FOR y = Math.max(0, playerY - 1); y <= Math.min(playerY + 1, worldHeight - 1); y++:
  FOR x = Math.max(0, playerX - 1); x <= Math.min(playerX + 1, worldWidth - 1); x++:
    if x == playerX and y == playerY:
      print("P");
    else:
      print(block at position [x][y])
  print empty line
print empty line
end function
```

```
function getCountryAndQuoteFromServer():
TRY:
Set link = "https://flag.ashish.nl/get_flag"
Setup a connection to link
Set request method of connection to "POST"
Set request property of connection to "Content-Type" as json
Enable output of connection

let payload be stringified json
let writer be OutputStreamWriter of connection
    Write payload to writer
    Flush writer
    Close writer

    let reader be BufferedReader of connection
    let sb be StringBuilder
    let line be empty string
    WHILE (line is not null):
    let line read next line of reader
    Append line line to sb
        json = ConvertToString(sb)

        let countryStart = FindSubstringIndex(json, " ") + 11
        let countryEnd = FindSubstringIndex(json, " ", countryStart)
        let country = Substring(json, countryStart, countryEnd)

    let quoteStart = FindSubstringIndex(json, " ") + 9
    let quoteEnd = FindSubstringIndex(json, " ", quoteStart)
    let quote = Substring(json, quoteStart, quoteEnd)

    quote = ReplaceSpaces(quote)
    Print("Country: " + country)
    Print("Quote: " + quote)
    CATCH Exception AS e:
```
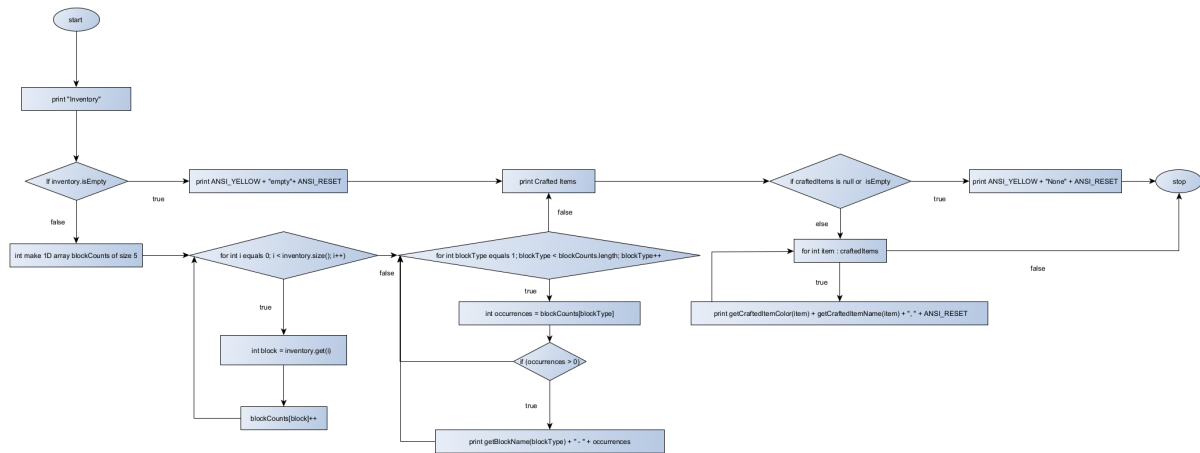
```
stackTrace = GetStackTrace(e)
Print("Error connecting to the server")
Print(stackTrace)
end function
```



```
function displayInventory:

print "Inventory"
if inventory.isEmpty
    print ANSI_YELLOW + " empty" + ANSI_RESET

else:
    make a 1D array blockCounts of size 5
    for int i = 0; i < inventory.size(); i++
    int block = inventory.get(i)
    blockCounts[block]

For int blockType = 1; blockType < blockCounts.length; blockType++
    int occurrences = blockCounts[blockType]
If occurrences > 0:
    Print getBlockName(blockType) + " - " + occurrences
print "Crafted Item"
if craftedItems is null or craftedItems is empty
    print ANSI_YELLOW_ + "none" + ANSi_RESET
else
    for int item : crafted item
    print getCraftedItemColor(item) + getCraftedItemName(item) + ", " + ANSI_RESET
```
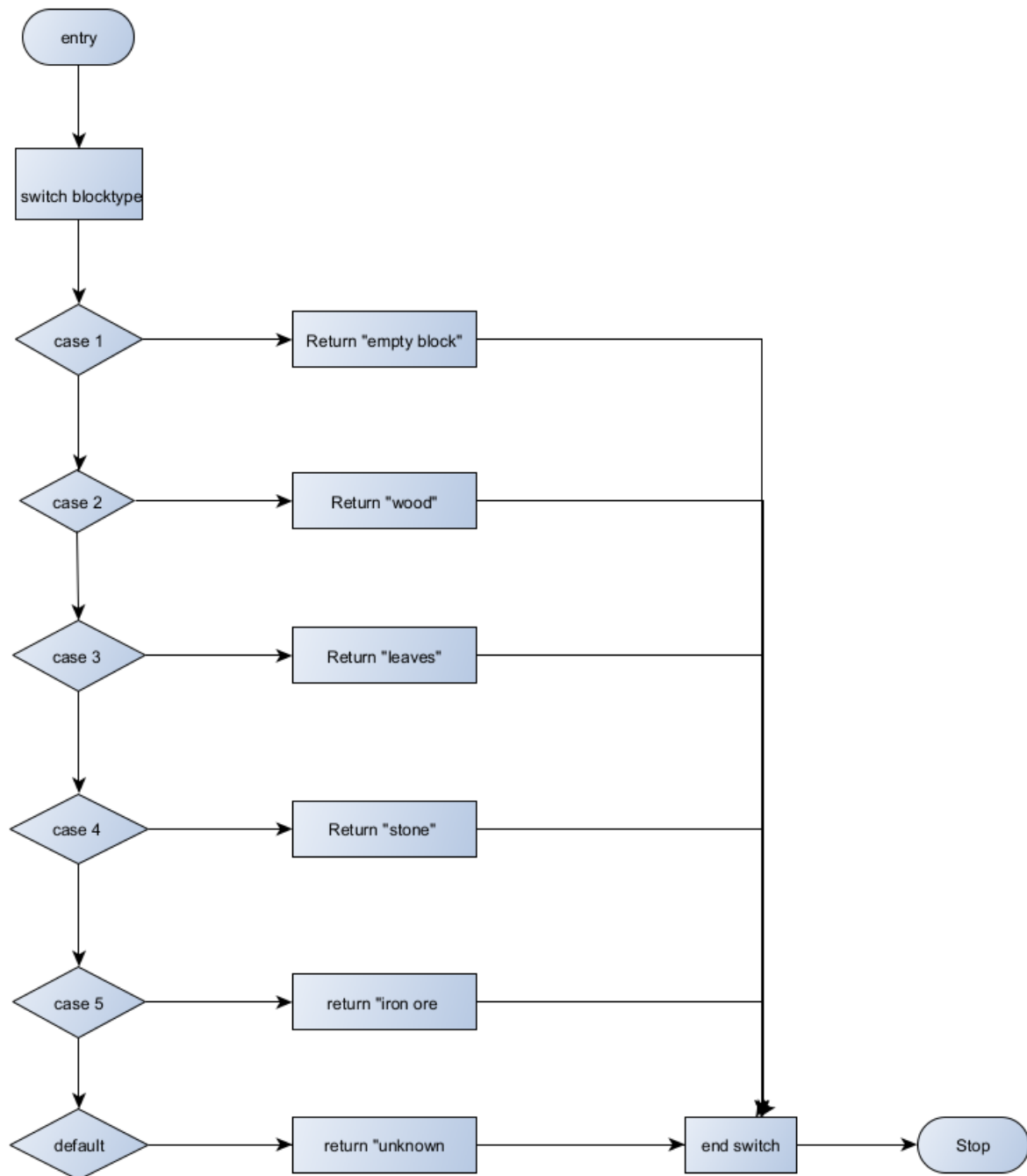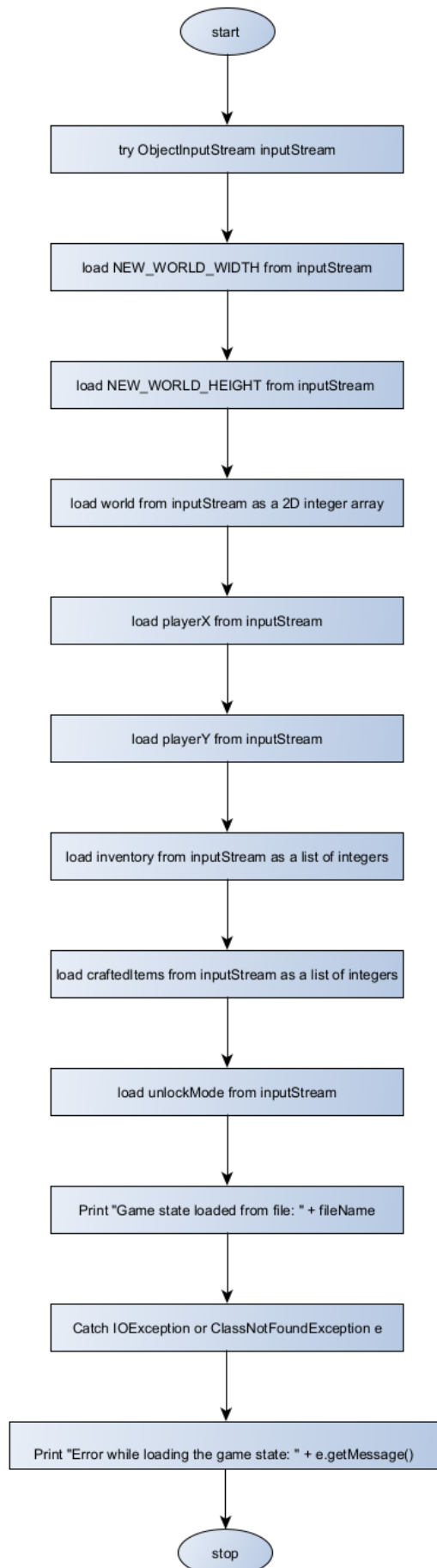
Function getBlockName(blockType):

Switch blockType
Case AIR:
 Return "Empty Block"
Case WOOD:
Return "Wood"
Case LEAVES:
Return "Leaves"
Case STONE:
Return "Stone"
 Case IRON_ORE:
Return "Iron Ore"
Default:
 Return "Unknown"

start

try ObjectInputStream inputStream

load NEW_WORLD_WIDTH from inputStream

load NEW_WORLD_HEIGHT from inputStream

load world from inputStream as a 2D integer array

load playerX from inputStream

load playerY from inputStream

load inventory from inputStream as a list of integers

load craftedItems from inputStream as a list of integers

load unlockMode from inputStream

Print "Game state loaded from file: " + fileName

Catch IOException or ClassNotFoundException e

Print "Error while loading the game state: " + e.getMessage()

stop

```
  Function loadGame(fileName)
Try Create ObjectInputStream inputStream using FileInputStream(fileName)
load NEW_WORLD_WIDTH from inputStream
load NEW_WORLD_HEIGHT from inputStream
load world from inputStream as a 2D integer array
load playerX from inputStream
load playerY from inputStream
load inventory from inputStream as a list of integers
load craftedItems from inputStream as a list of integers
load unlockMode from inputStream
Print "Game state loaded from file: " + fileName
 Catch IOException or ClassNotFoundException e
Print "Error while loading the game state: " + e.getMessage()
```
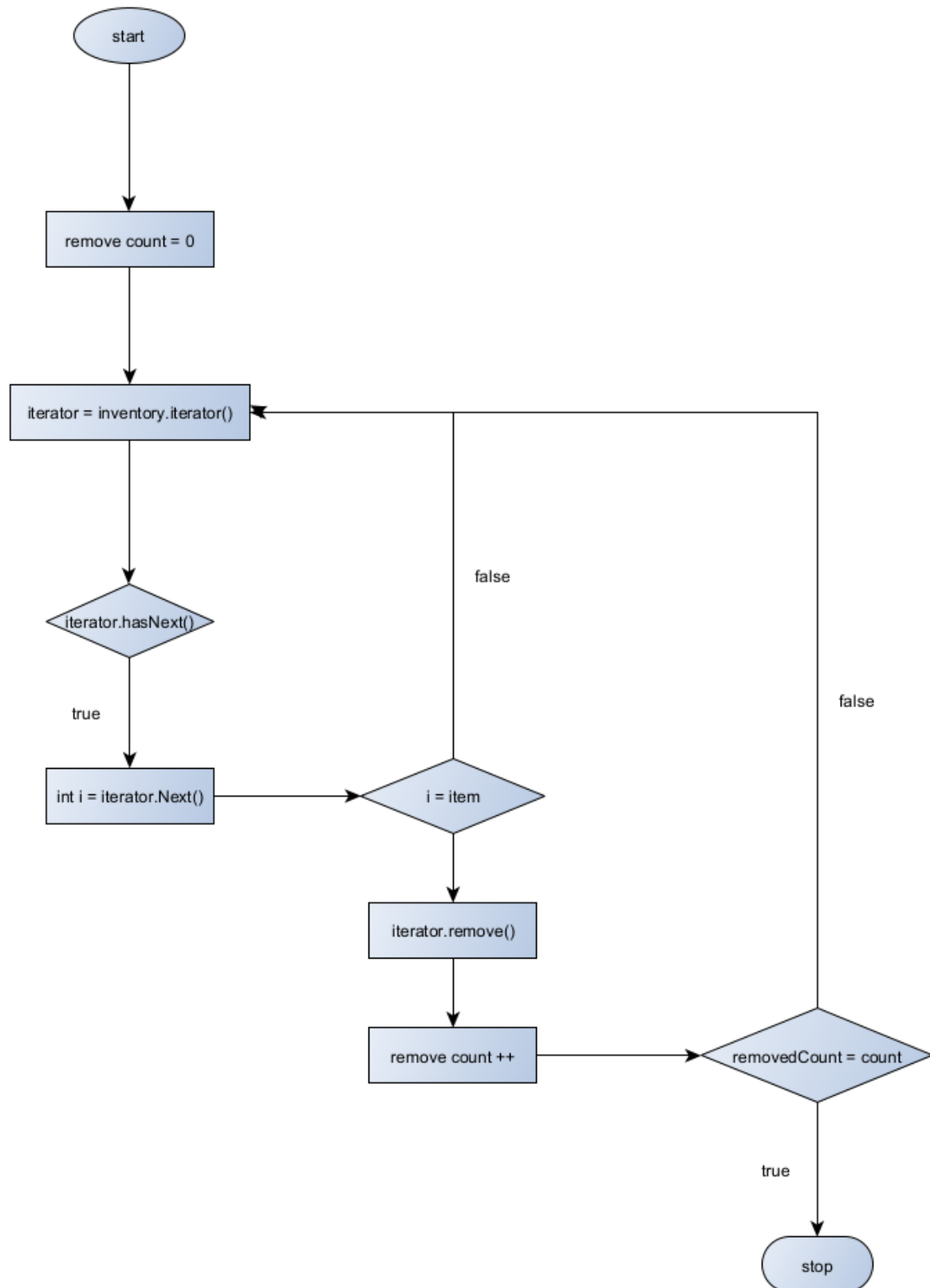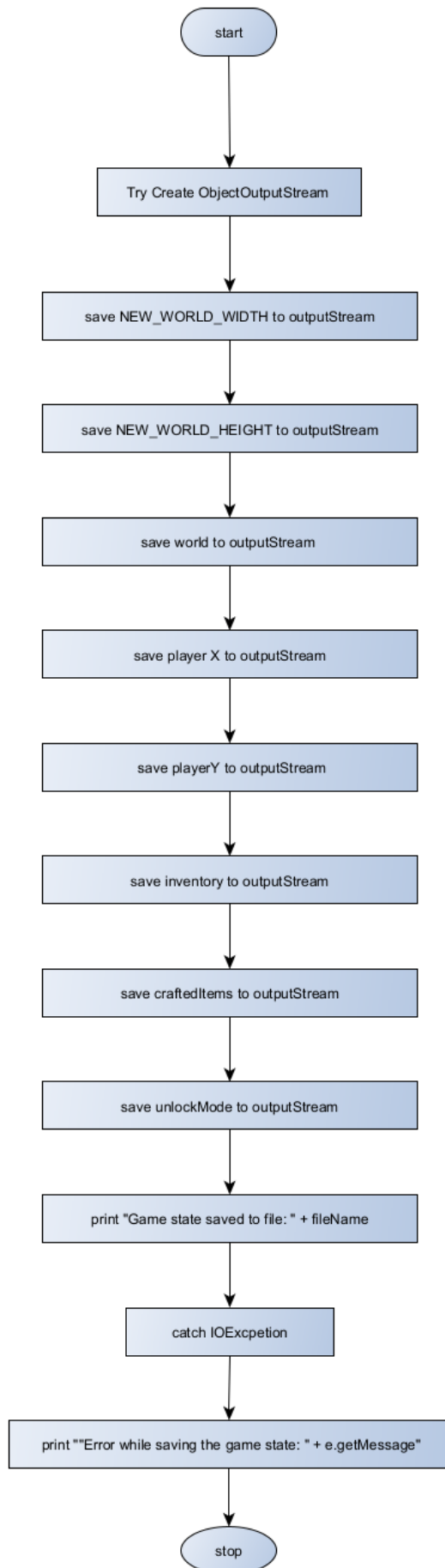
```
function removeItemsFromInventory(item, count):

removedCount = 0
iterator = createIterator(inventory)
While iterator.hasNext()
    i = iterator.next()
```

```
If i is equal to item iterator.remove()
   removedCount = removedCount + 1
If removedCount is equal to count
    break
```

start

Try Create ObjectOutputStream

save NEW_WORLD_WIDTH to outputStream

save NEW_WORLD_HEIGHT to outputStream

save world to outputStream

save player X to outputStream

save playerY to outputStream

save inventory to outputStream

save craftedItems to outputStream

save unlockMode to outputStream

print "Game state saved to file: " + fileName

catch IOExcpetion

print ""Error while saving the game state: " + e.getMessage"

stop

```
  function saveGame(fileName):

Try  ObjectOutputStream outputStream
 save NEW_WORLD_WIDTH to outputStream
save NEW_WORLD_HEIGHT to outputStream
 save world to outputStream
save playerX to outputStream
 save playerY to outputStream
 save inventory to outputStream
save craftedItems to outputStream
save unlockMode to outputStream
 Print "Game state saved to file: " + fileName Catch IOException e
Print "Error while saving the game state: " + e.getMessage()
```