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 7, 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 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.

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.
Insert flowchart of game Insert pseudocode for game

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

Secret Door Logic Analysis: Describe the secret doors functionality • FSA Illustration and Description: Attach FSA diagram

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 | Team Overview

Below describes what each team member did on the project:

6.1 | Long Luong

Project leader
LaTeX Document owner and composer
Created 1 flowchart with pseudocode:
1. getCountryAndQuoteFromServer
Finalised the Function Exploration part
Helped others with git issues
Refined some pseudocode that others have made

6.2 | Élisa Donéa

Designed FSA for secret door with Chris
Initiated with the Function Exploration
Create 5 flowcharts with pseudocode:
1. fillInventory
2. generateEmptyWorld
3. generateWorld
4. initGame
5. lookAround

6.3 | Chris Munteanu

Designed FSA for secret door with Élisa
Create 5 flowcharts with pseudocode:
1. displayInventory
2. getBlockName
3. loadGame
4. removeItemsFromInventory
5. saveGame

6.4 | Alexia Raportaru

Create 5 flowcharts with pseudocode:
1. placeBlock
2. craftItem
3. interactWithWorld
4. mineBlock
5. movePlayer
Created an impressive flowchart of the entire game along with its pseudocode explanation

6.5 | Explanation

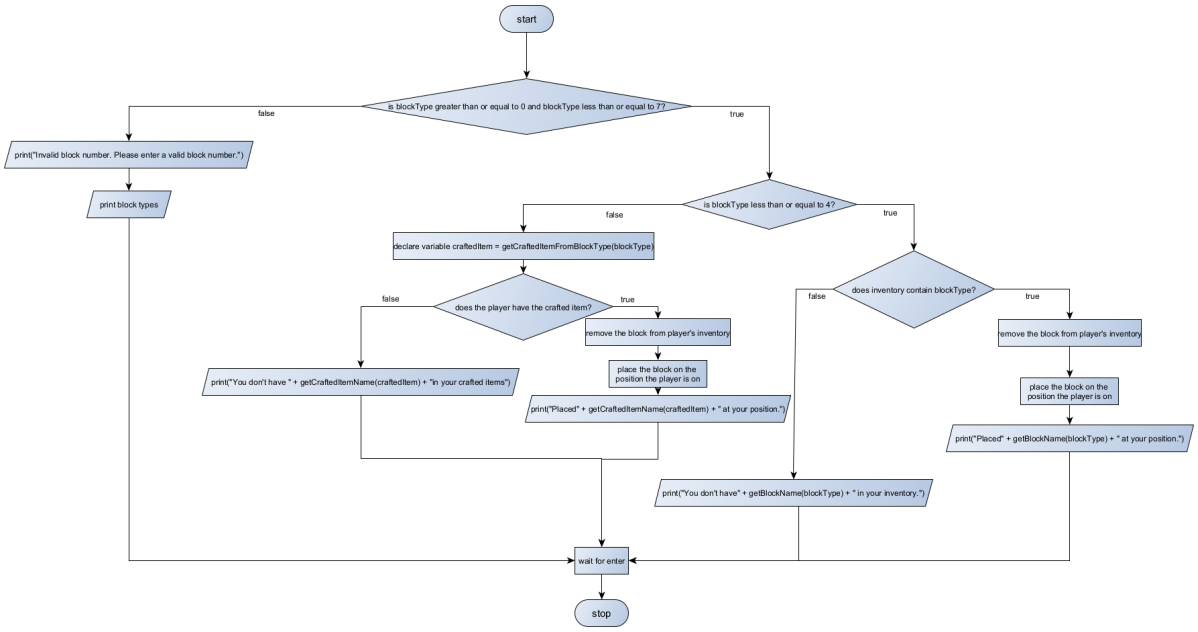
When we first met up with each other we were with the three of us. During the first week we got to know each other. Long proposed to be the document manager because he knows how to use LaTeX. He also created a (now deprecated) GitLab repository found at [URL]. During the second week Alexia joined the group. We met up with each other and we divided the roles. We had to create at least 16 flowcharts alongside with pseudocode. Long mentioned that most of the functions are vey easy to understand, and since he wanted a difficult one he proposed to do the function getCountryAndQuoteFromServer and he proposed the other three group members to do 5 flowcharts with their respective pseudocode. Everybody agreed and thinks it is a good idea. In week 3 Élisa and Chris decided to work on the FSA together. Everybody in the group was new to Git except Long, so he helped out everyone set-up the Git environment and he explained to everyone how it works. In the meantime, Alexia worked on the flowchart and the

pseudocode of the entire game JavaCraft. Later, Long found out he made a mistake and that we had to create a branch on the repository [LINK] so he informed everybody about the mistake and that we all had to migrate to the correct branch.

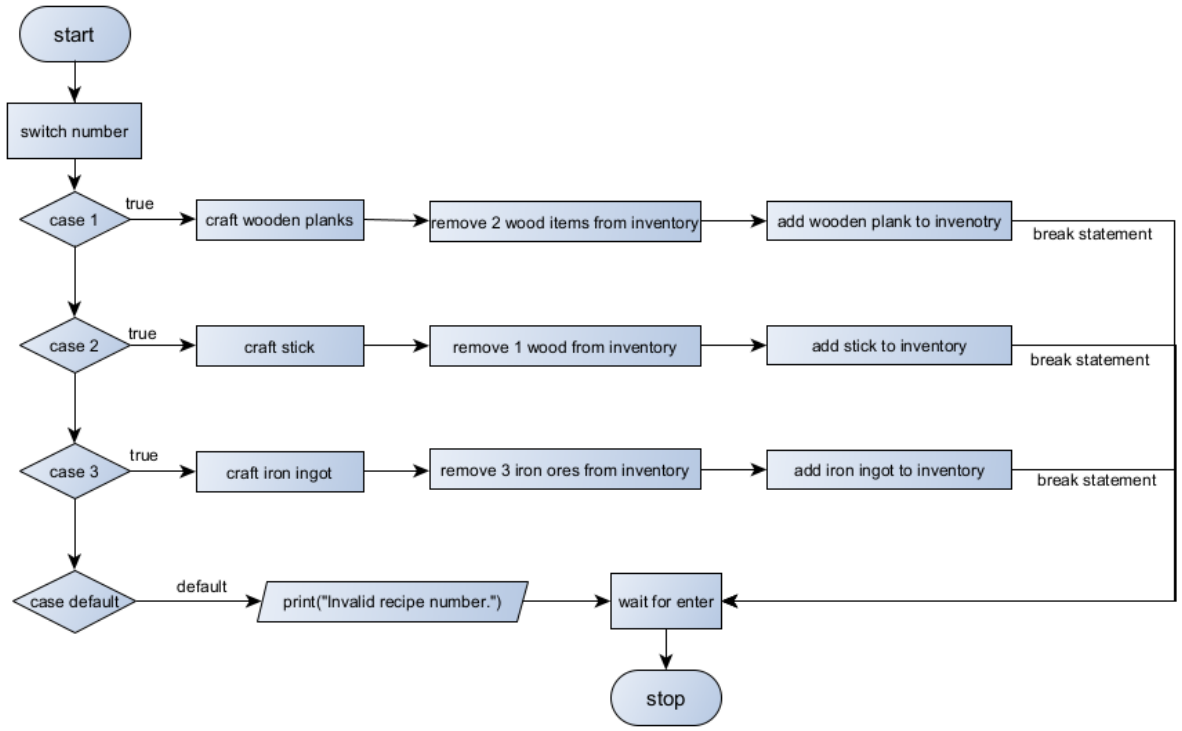
7 | References

A | Appendix: pseudocode and flowcharts

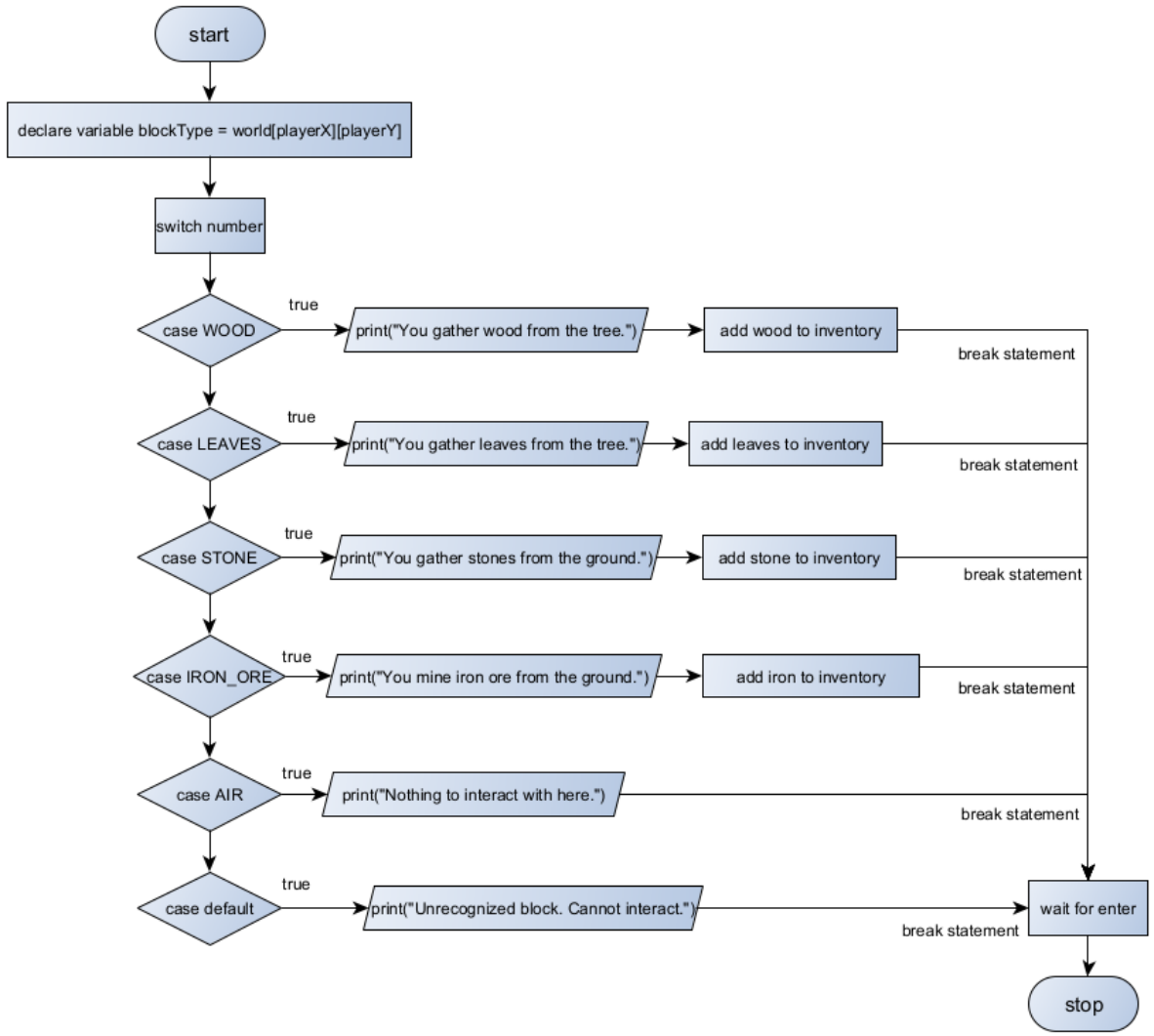
This appendix provides the full blocks of pseudocode and its flowcharts.




```
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
```

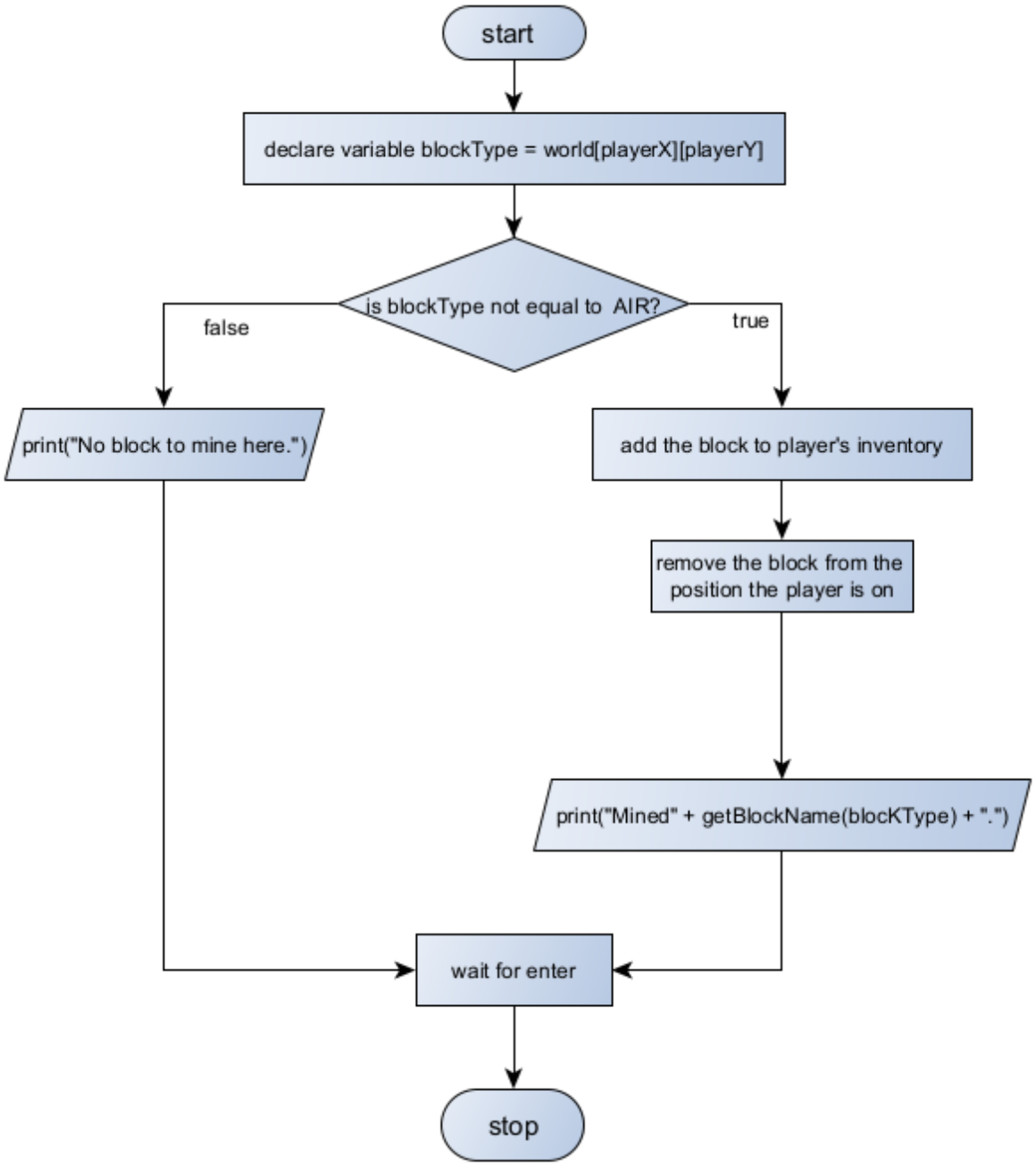



```
function craftItem(recipe)

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
```

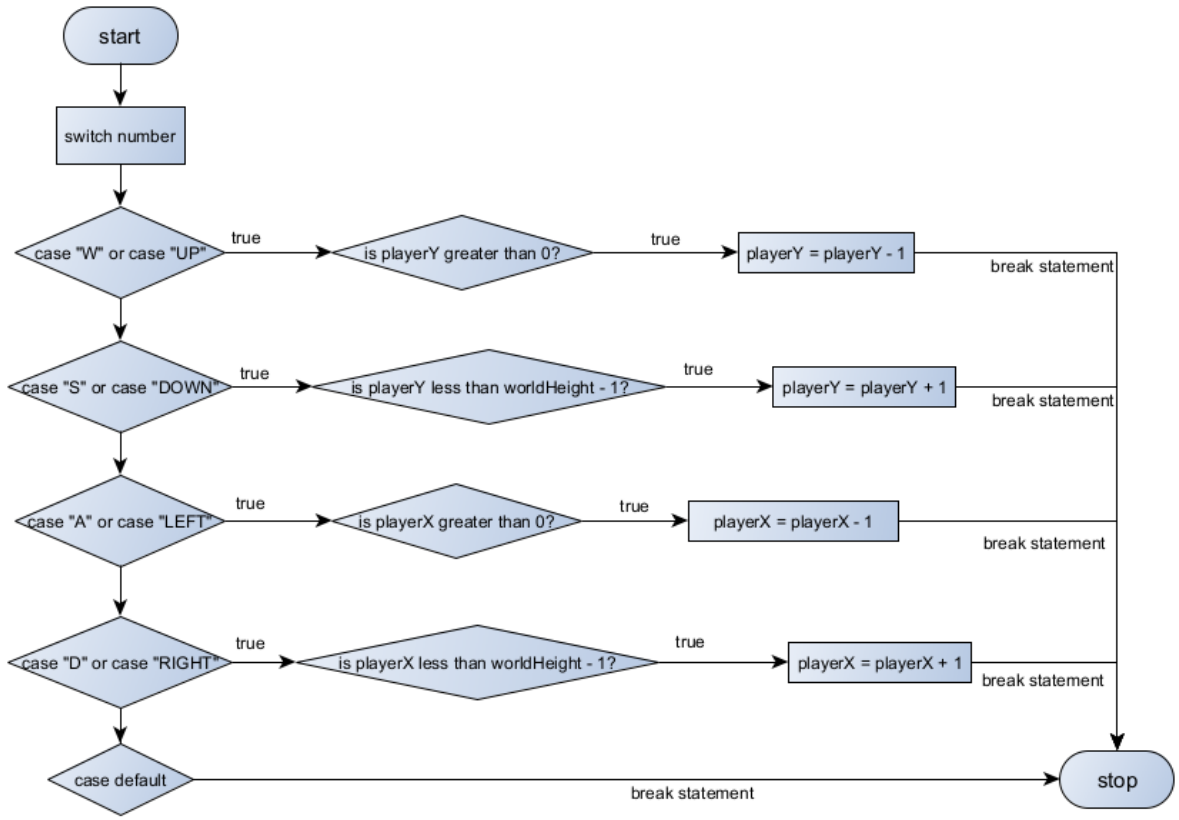



```
function interactWithWorld()

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
```

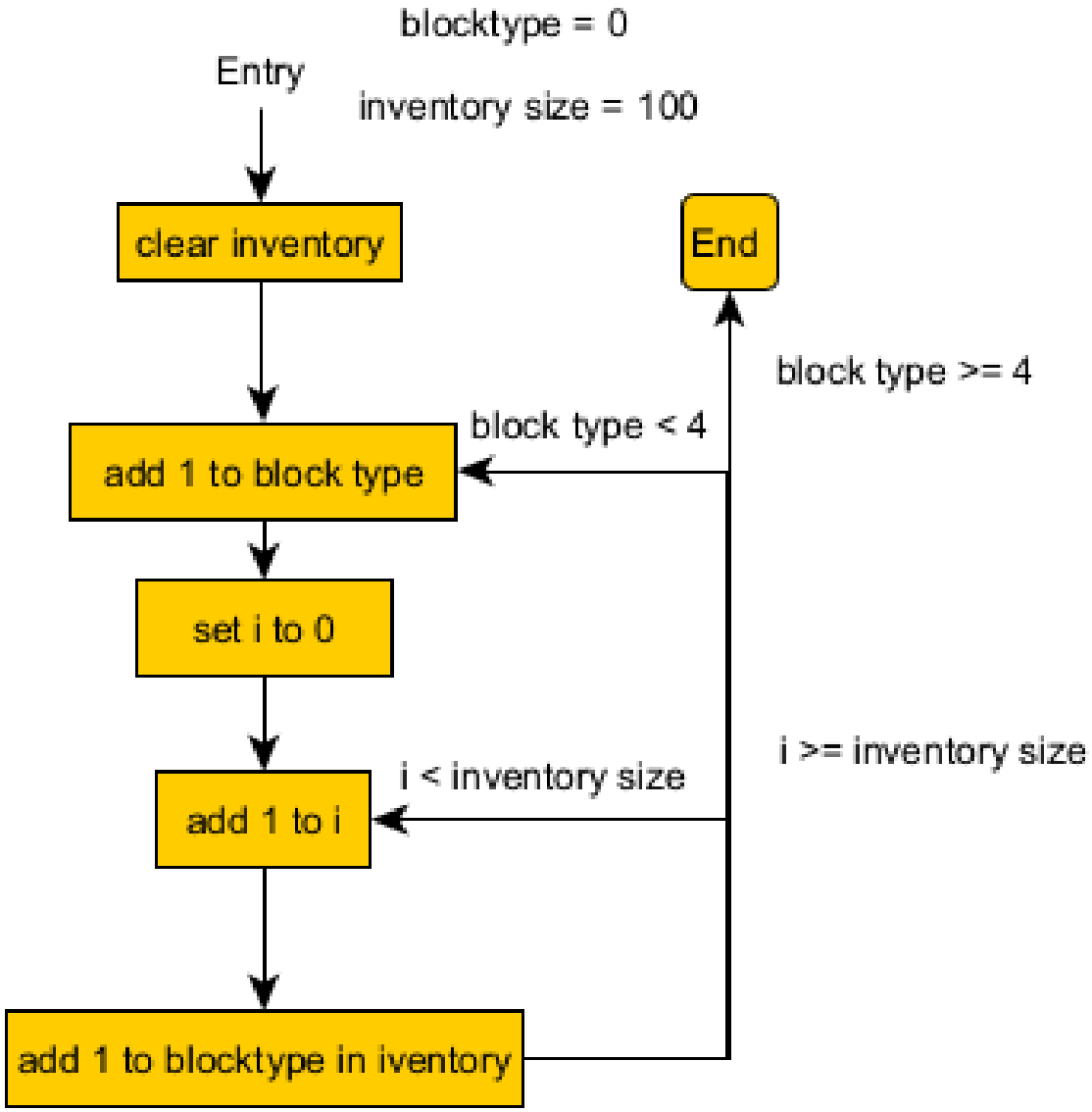
```
waitForEnter()
end function
```



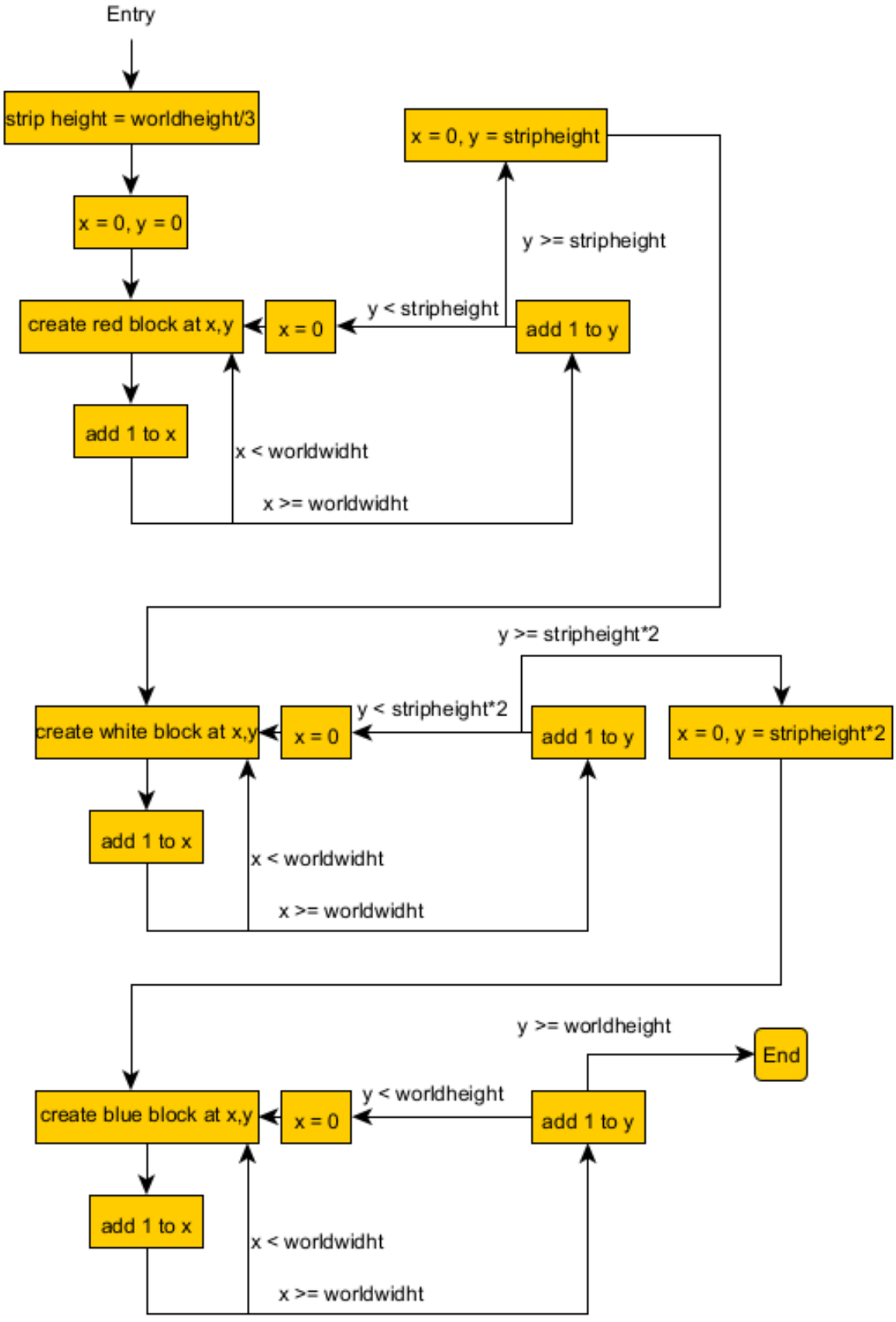

```
function mineBlock()

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
```

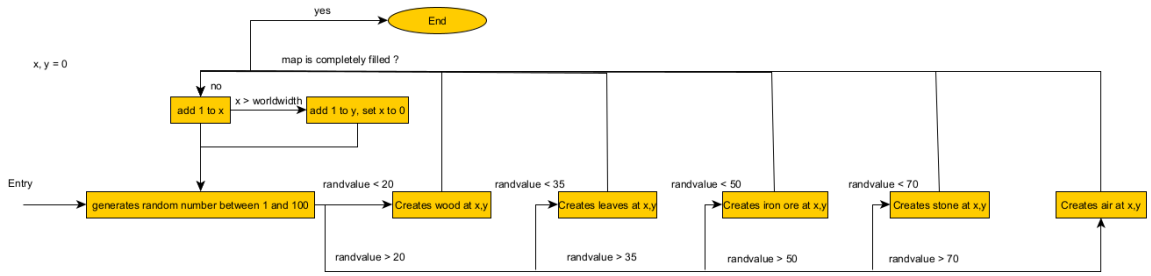



```
fillInventory()
Set INVENTORY_SIZE = 100

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

end function
```



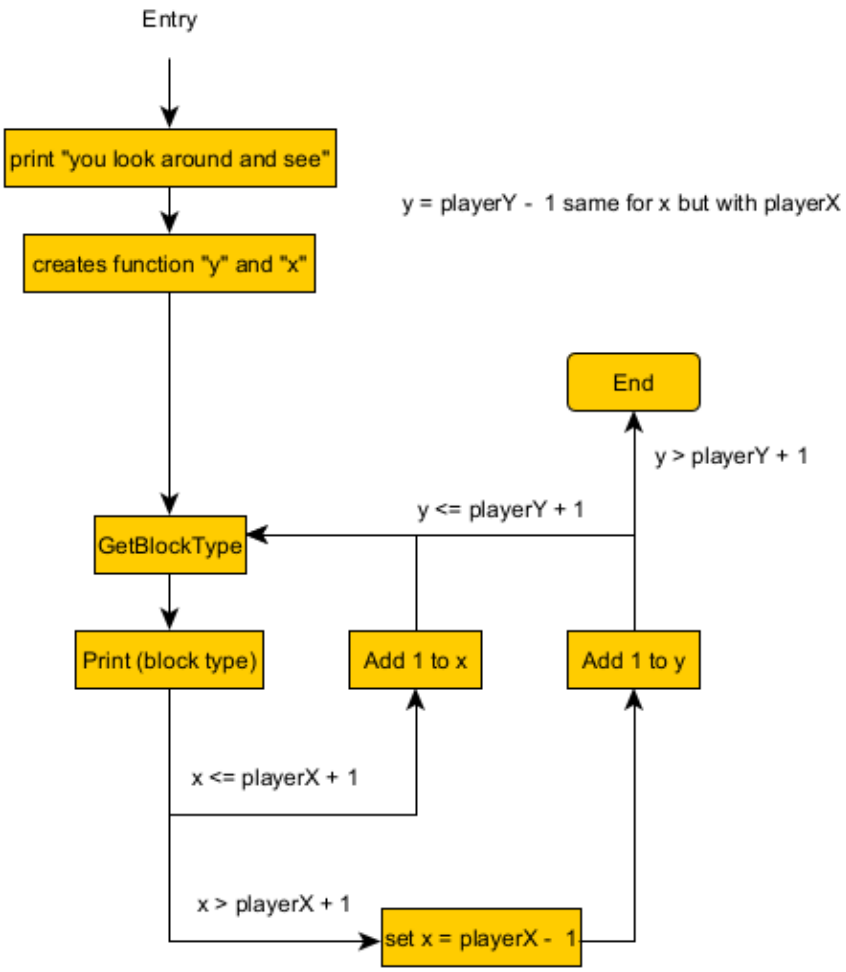

```
function generateEmptyWorld()
```

```
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
```

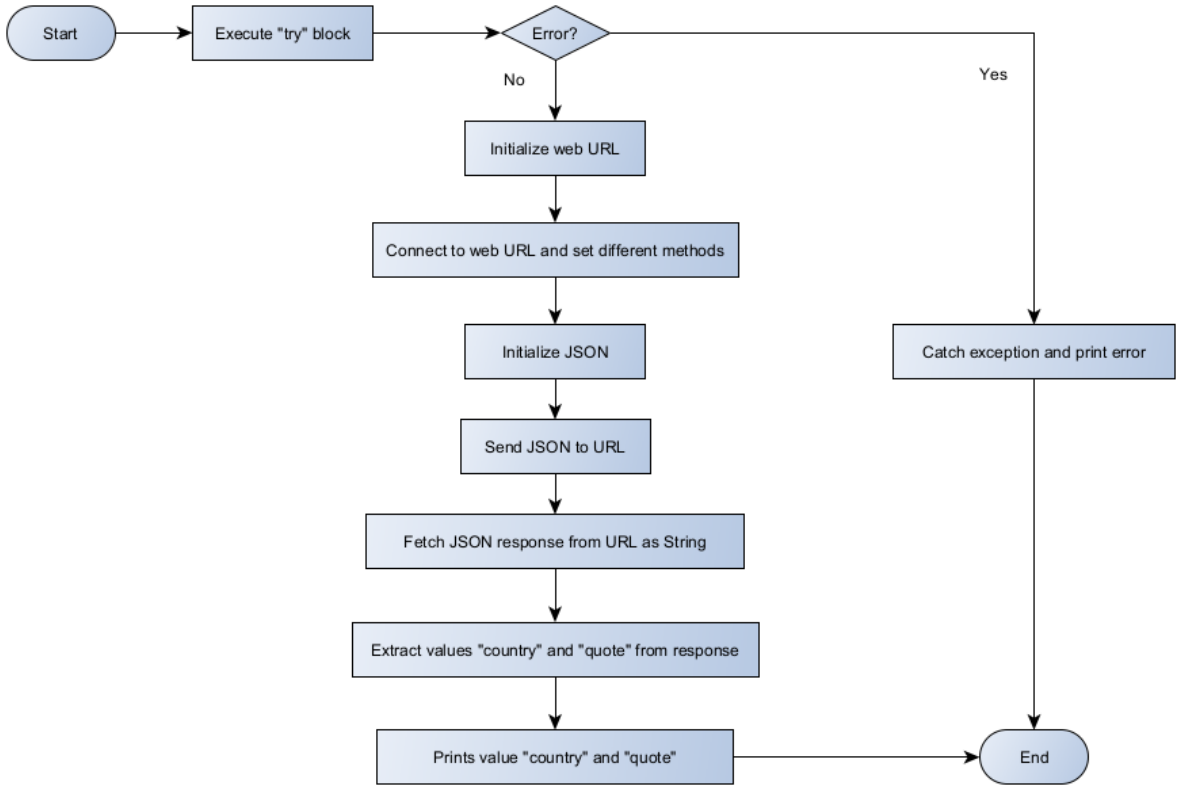



```
function generateWorld()

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
```