**Project Report**

**Project Report: Group 79**

Sunday, October 8, 2023

## Table of contents

[**1 Introduction** **2**](#_Toc3329)

[**2 JavaCraft’s Workflow** **2**](#_Toc3330)

[**3 Functionality Exploration** **2**](#_Toc3331)

[**4 Finite State Automata (FSA) Design** **3**](#_Toc3332)

[**5 Git Collaboration & Version Control** **3**](#_Toc3333)

[**6 Extending the Game Code (For Final Submission)** **3**](#_Toc3334)

[**7 Interacting with Flags API (For Final Submission)** **3**](#_Toc3335)

[**8 Conclusion (For Final Submission)** **3**](#_Toc3336)

[**9 Appendix** **3**](#_Toc3337)

[**10 References** **3**](#_Toc3338)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Attribute | | Details | | |
| Group Name | | Group79 | | |
| Group Number | | 79 | | |
| TA | | [] | | |
| Student Name | | Student ID |
| Hagenbeek Mika | | i6286542 |
| Fernandes Kennedy | | i6361324 |
| Vieversys Gustas | | i6358158 |
| Zerbi Liam | | i6368807 |
| Karaintros George | | i6356505 |

# Introduction

JavaCraft is a robust, text-based game project built entirely in Java that provides students with a holistic academic exercise in computer science, logical thinking, and collaboration skills. This project brings together over 35 Java functions to create a diverse gameplay experience that engages creativity, analytics, and technical expertise. We worked in a team of five to implement JavaCraft. Through this team-based approach, students develop key collaborative abilities such as communication, task delegation, and source control. The project has clearly defined learning outcomes covering core computer science concepts like system design, flow control, data structures, and object-oriented programming.

**Who did what Overview :**

|  |  |
| --- | --- |
| Section | Student(s) |
| Introduction | Liam |
| JavaCraft’s Workflow | Liam |
| Functionality Exploration | Mika (20%)/ Gustas (20%)/ Kennedy (20%)/ Liam (20%)/George (20%) |
| Secret Door (FSA) Design | Mika (70%)/ George (30%) |
| Git Collaboration & Version Control | Mika (20%)/ Gustas (20%)/ Kennedy (20%)/ Liam (20%)/George (20%) |
| Flowcharts/Pseudocodes of 15 functions | Mika (20%)/ Gustas (20%)/ Kennedy (20%)/ Liam (20%)/George (20%) |
| Report | Liam |

# JavaCraft’s Workflow

* Flowchart For Game A diagram of a diagram

  Description automatically generated

Pseudocode For Game:

Initialize Game(width, height)

Set worldWidth to width

Set worldHeight to height

Initialize world as a 2D array of size (width, height)

Set playerX to width / 2

Set playerY to height / 2

Initialize inventory as an empty list

Set unlockMode to false

Set secretDoorUnlocked to false

Set inSecretArea to false

Generate World()

For each cell (x, y) in world

Generate a random value

Assign a block type based on the random value

Display World()

Display the world map with symbols and colors

Highlight the player's position

Move Player(direction)

Update player's position based on the direction

Mine Block()

Get the block type at the player's position

If the block is not empty

Add the block to the inventory

Set the block at the player's position to empty

Place Block(blockType)

If the block type is valid

If the block is in the inventory

Remove the block from the inventory

Place the block at the player's position

Craft Item(recipe)

Check the recipe number

If it matches a crafting recipe

Craft the corresponding item if resources are available

Interact with World()

Get the block type at the player's position

Perform an interaction based on the block type

Save Game(fileName)

Serialize game state data and write to the file

Load Game(fileName)

Deserialize game state data from the file and load into the program

Unlock Secret Door()

Check if certain conditions are met (unlockMode, actions performed)

Set secretDoorUnlocked to true

Reset the world

Get Country and Quote from Server()

Send a request to a server to get country and quote data

Parse the response and display country and quote

Main()

Initialize the game

Generate the world

Display instructions

Start the game loop

Handle user input and game logic

End the game loop

Exit the game

# Functionality Exploration

List of key functionalities explored:

|  |  |  |
| --- | --- | --- |
| No. | Function Name | Description |
| 1 | Main | This is the entry point of the program. It initializes the game, generates the game world, and handles user input to start the game or exit. |
| 2 | initGame | this function initializes the game loading up any initial information such as the welcome page as well as basic instructions on how to start up the program |
| 3 | generateWorld | this function generates a world, filling up the 2D array with different blocks based on a random value generator. Based on the value, each ‘block’ is filled with a certain element such as leaves |
| 4 | displayWorld | this method goes through the array and displays a certain character based on what value is stored in that part of the array. This repeats for each value until the boundaries ‘worldHeight’ and ‘worldWidth’ are reached |
| 5 | getBlockSymbol | this method ‘gets’ the block symbol depending on what value is stored within that space in the array. |
| 6 | getBlockChar | Returns the ASCII character for a given block type used in crafting recipes. |
| 7 | startGame | this function starts the game, this means that after the user inputs ‘y’, the game begins and the function startGame calls upon functions such as displayWorld and others in order to show and display the game |
| 8 | movePlayer | Moves the player's position based on the input direction (WASD or arrow keys). |
| 9 | mineBlock | Makes the player mine and obtain the block placed in the position of the player, if there is indeed a block this function will notify that the action has been successful and the type of block mined. In the same way, if there is no block that can be mined the system will tell it to the player |
| 10 | displayInventory | The system will display a section which is the player’s inventory, containing every item and block the player has obtained, specifying the quantity of each |
| 11 | placeBlock | Allows the player to place a block from their inventory at their current position. |
| 12 | displayCraftingRecipes | The system will display a section which is the player’s inventory, containing every item and block the player has obtained, specifying the quantity of each |
| 13 | craftItem | Crafts an item based on the selected recipe. |
| 14 | interactWithWorld | Allows the player to interact with different block types in the game world, adding them to the inventory. |
| 15 | saveGame | Saves the current game state, including world data, player position, and inventory, to a file. |
| 16 | loadGame | Loads a saved game state from a file and restores it to continue the game. |
| 17 | lookAround | Displays a limited view of the nearby blocks from the player's current position. |
| 18 | getCountryAndQuoteFromServer | Makes an HTTP request to a server to retrieve and display a country name and a quote. |
| 19 | waitForEnter | Pauses the game and waits for the player to press Enter. |
| 20 | resetWorld | Resets the game world to its initial state for the secret door unlock sequence. |
| 21 | clearScreen | Clears the console screen, providing a visual update for the player. |
| 22 | fillInventory | this method first clears the player’s inventory, then it loops and ‘fills up’ the inventory of the player with the certain block type. This loop ends once inventory is full. |
| 23 | generateEmptyWorld | generates a new map with the a custom width and height ,as well as the default  red block, white block and blue block amount, it also divides the height into 3 equal parts |
| 24 | getBlockName | this function gets the block name depending on the block type, it returns wood if the block type is wood etc etc |
| 25 | getCraftedItemFromBlockType | Converts a block type to its corresponding crafted item |
| 26 | getCraftedItemName | Returns the name of a crafted item based on its type. |
| 27 | craftWoodenPlanks | this method creates the item WoodenPlanks and adds it to your inventory. To do this you need to have the blocks needed to build it, that being 2 wood blocks. It also removes 2 wood blocks from your inventory. |
| 28 | craftStick | this method creates the item stick and adds it to your inventory. To do this you need to have the blocks needed to build it, that being 1 wood block. It also removes 1 wood block from your inventory. |
| 29 | craftIronIngot | this method creates the item ironIngot and adds it to your inventory. To do this you need to have the blocks needed to build it, that being 3 iron ore. It also removes 3 iron ore from your inventory. |
| 30 | getBlockTypeFromCraftedItem | depending on what type of block the user has the code lets the user print either wooden\_planks, crafted\_sticks or crafted\_iron\_ingot |
| 31 | getCraftedItemFromBlockType | Converts a block type to its corresponding crafted item. |
| 32 | displayLegend | this function allows the player to understand which block is represented by what symbol on the map. It shows colored text and says what type of block it shows |
| 33 | removeItemsFromInventory | this method removes a certain item as well as the amount you remove from a player's inventory |
| 34 | inventoryContains | using the name of the item as a parameter, this function checks if the inventory contains that item. |
| 35 | getBlockColor | this method is used in order to show the color of each block. For example, air is blank, wood is red, etc etc. |
| 36 | addCraftedItem | Adds a crafted item to the player's list of crafted items. |

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

# Finite State Automata (FSA) Design

* Secret Door Logic Analysis:

The secret door and unlock logic allow the player to access a hidden "secret area" in the game world, but only if they perform the correct sequence of actions to unlock it.

How it works:

-There is a special "unlock mode" that tracks the player's actions

-The player enters "unlock" command to activate unlock mode

-In unlock mode, the following actions are tracked:

-Movement (with WASD or arrow keys)

-Mining a block (with M)

-Crafting an item (with C)

-Opening the door (with Open)

-To unlock the door, the player must:

-Enter unlock mode

-Perform movement

-Perform mining

-Perform crafting

-Then enter the Open command

-If the correct sequence is entered, the secretDoorUnlocked boolean is set to true

-This generates a new world and fills the player inventory to access the secret area

* FSA Illustration & Description: A diagram of a diagram

  Description automatically generated

# Git Collaboration & Version Control

* Repository Link: https://gitlab.maastrichtuniversity.nl/bcs1110/javacraft.git
* Branch Details: Branch Group79, Members: Mika Hagenbeek, Liam Zerbi, Kennedy Fernandes, Gustas Vieversys, George Karaintros

# Extending the Game Code (For Final Submission)

[Provide details on the new block types, craft recipes, and their integration into the game. Include code snippets where appropriate]

[Provide Java code here]

# Interacting with Flags API (For Final Submission)

[Details on Flags API exploration and flag rendering on the grid.]

# Conclusion (For Final Submission)

[Provide a summary of achievements, challenges, and learnings.]

# Appendix

# playerMove function:

Flowchart: A diagram of a game

Description automatically generated

Pseudocode:

Function playerMove(direction, playerX, playerY, worldWidth, worldHeight)

Switch direction:

Case "W" or "UP":

If playerY > 0:

Decrement playerY by 1

Case "S" or "DOWN":

If playerY < worldHeight - 1:

Increment playerY by 1

Case "A" or "LEFT":

If playerX > 0:

Decrement playerX by 1

Case "D" or "RIGHT":

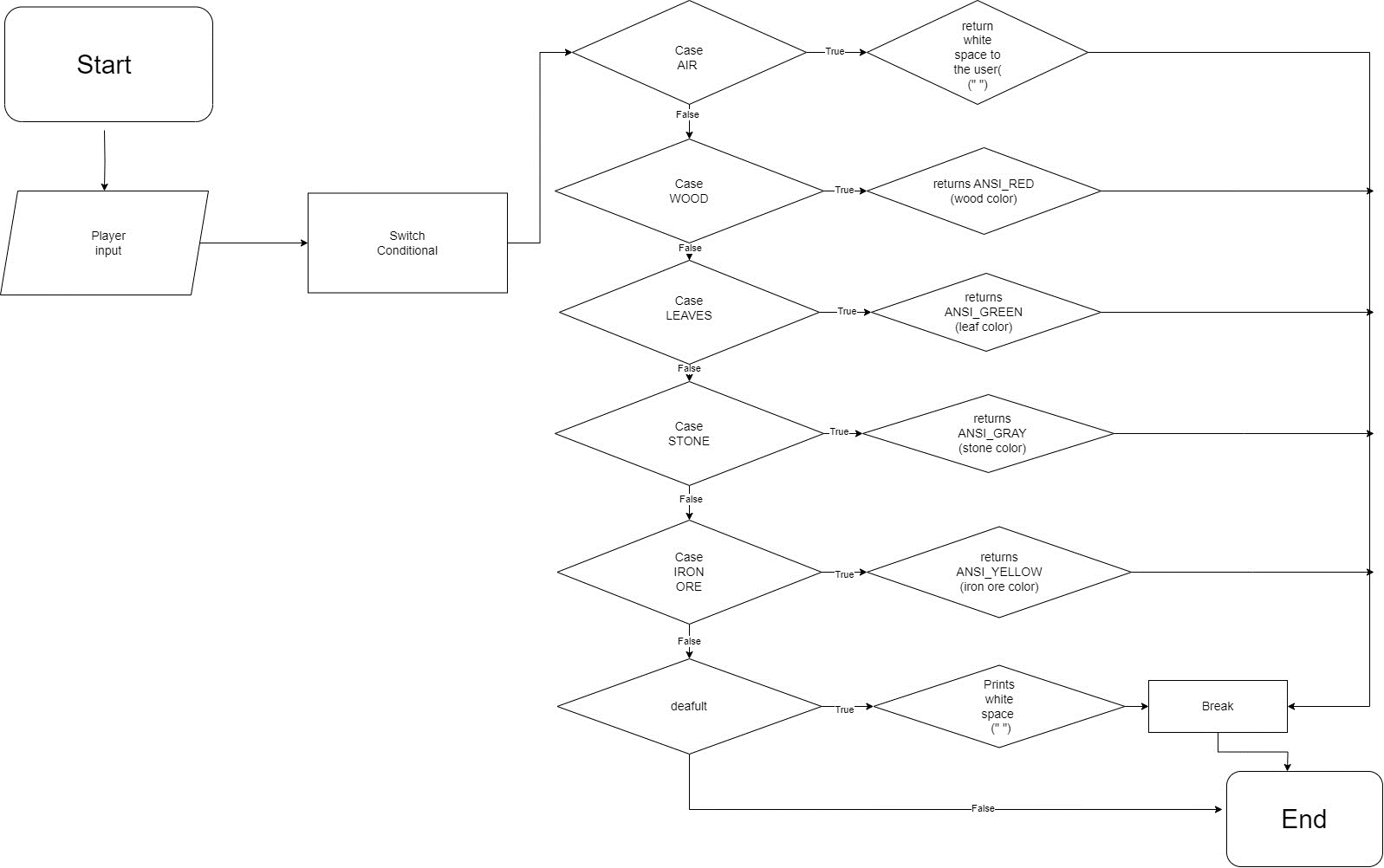
If playerX < worldWidth - 1:

Increment playerX by 1

Default:

Do nothing for invalid input.

getBlockColor function:

Flowchart: 

PseudoCode:

function getBlockColor(blockType)

if blockType is AIR

return "No Color"

else if blockType is WOOD

return "Red"

else if blockType is LEAVES

return "Green"

else if blockType is STONE

return "Gray"

else if blockType is IRON\_ORE

return "White"

else if blockType is SAND

return "Yellow"

else if blockType is OBSIDIAN

return "Purple"

else

return "Unknown Color"

displayLegend function :

Flowchart : A diagram of a flowchart

Description automatically generated

PseudoCode :

function displayLegend()

print "Legend:"

print "-- - Empty block"

print "?? - Wood block"

print "## - Leaves block"

print "[] - Stone block"

print "@@ - Iron ore block"

print "?? - Sand block"

print " ¥ ¥ - Obsidian block"

print "P - Player"

removeItem function :

Flowchart : A diagram of a process

Description automatically generated

PseudoCode :

function removeItem(item, count)

Initialize itemCount to 0

Initialize iterator to inventory.iterator()

while iterator has next

itemInInventory = iterator.next()

if itemInInventory is equal to item

Remove itemInInventory from inventory

Increment itemCount by 1

if itemCount is equal to count

Exit loop

addCraftedItem function :

Flowchart : A diagram of a work flow

Description automatically generated

PseudoCode :

function removeItem(item, count)

Initialize itemCount to 0

Initialize iterator to inventory.iterator()

while iterator has next

itemInInventory = iterator.next()

if itemInInventory is equal to item

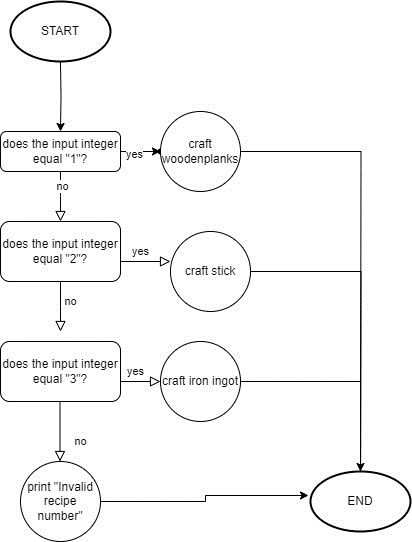
Remove itemInInventory from inventory

Increment itemCount by 1

if itemCount is equal to count

Exit loop

craftItem function :

Flowchart : 

PseudoCode :

function craftItem(recipe)

case recipe of

1:

craftWoodenPlanks()

2:

craftStick()

3:

craftIronIngot()

4:

craftSandStone()

default:

print "Invalid recipe number."

displayInventory function :

Flowchart : A diagram of a flowchart

Description automatically generated

PseudoCode :

function displayInventory()

print "Inventory:"

if inventory is empty

print "Empty"

else

Initialize blockCounts as an array of size 7, all elements set to 0

for item in inventory

Increment the corresponding element in blockCounts

for blockType from 1 to 6

occurrences = blockCounts[blockType]

if occurrences is greater than 0

print "Block: " + getBlockName(blockType) + " - Count: " + occurrences

print "Crafted Items:"

if craftedItems is null or empty

print "None"

else

for item in craftedItems

print "Crafted Item: " + getCraftedItemName(item)

getBlockName function :

Flowchart : A diagram of a block diagram

Description automatically generated

PseudoCode :

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"

case SAND:

return "Sand"

case OBSIDIAN:

return "Obsidian"

default:

return "Unknown"

placeBlock function :

Flowchart : A diagram of a diagram

Description automatically generated

PseudoCode :

Function placeBlock(blockType, inventory, world, playerX, playerY)

If blockType is in range [0, 10]:

If blockType <= 6:

If inventory contains blockType:

Remove blockType from inventory

Set world[playerX][playerY] to blockType

Output "Placed " + getBlockName(blockType) + " at your position."

Else:

Output "You don't have " + getBlockName(blockType) + " in your inventory."

Else:

Set craftedItem to getCraftedItemFromBlockType(blockType)

If craftedItems contains craftedItem:

Remove craftedItem from craftedItems

Set world[playerX][playerY] to blockType

Output "Placed " + getCraftedItemName(craftedItem) + " at your position."

Else:

Output "You don't have " + getCraftedItemName(craftedItem) + " in your crafted items."

Else:

Output "Invalid block number. Please enter a valid block number."

interactWithWorld function :

Flowchart : A diagram of a flowchart

Description automatically generated

PseudoCode :

Function interactWithWorld(blockType, inventory)

Switch blockType:

Case WOOD:

Output "You gather wood from the tree."

Add WOOD to inventory

Case LEAVES:

Output "You gather leaves from the tree."

Add LEAVES to inventory

Case STONE:

Output "You gather stones from the ground."

Add STONE to inventory

Case IRON\_ORE:

Output "You mine iron ore from the ground."

Add IRON\_ORE to inventory

Case SAND:

Output "You grabbed sand from the ground."

Add SAND to inventory

Case OBSIDIAN:

Output "You mined obsidian from the ground."

Add OBSIDIAN to inventory

Case AIR:

Output "Nothing to interact with here."

Default:

Output "Unrecognized block. Cannot interact."

craftWoodenPlanks function :

Flowchart : A diagram of a wood crafting process

Description automatically generated

PseudoCode :

Function craftWoodenPlanks(inventory, craftedItems)

If inventoryContains(WOOD, 2):

RemoveItemsFromInventory(WOOD, 2)

AddCraftedItem(CRAFTED\_WOODEN\_PLANKS)

Output "Crafted Wooden Planks."

Else:

Output "Insufficient resources to craft Wooden Planks."

displayCraftingRecipes function :

Flowchart : A diagram of a crafting process

Description automatically generated

PseudoCode :

Function displayCraftingRecipes()

Output "Crafting Recipes:"

Output "1. Craft Wooden Planks: 2 Wood"

Output "2. Craft Stick: 1 Wood"

Output "3. Craft Iron Ingot: 3 Iron Ore"

Output "4. Craft Sandstone: 1 Sand block, 1 Stone"

mineBlock function :

Flowchart : A diagram of a flowchart

Description automatically generated

PseudoCode :

Function mineBlock(blockType, inventory, world, playerX, playerY)

If blockType is not AIR:

Add blockType to inventory

Set world[playerX][playerY] to AIR

Output "Mined " + getBlockName(blockType) + "."

Else:

Output "No block to mine here."

saveGame function :

Flowchart : A diagram of a game

Description automatically generated

PseudoCode :

Function saveGame(filename)

open filename for binary writing

serialize the following game state:

world array

player position

inventory

crafted items

unlock mode

write the serialized data to the file

close the file

loadGame :

Flowchart : A diagram of a game

Description automatically generated

PseudoCode :

Function loadGame(filename)

open filename for binary reading

deserialize the game state including:

world array

player position

inventory

crafted items

unlock mode

load the deserialized state into the program:

set world grid

set player position

set inventory

set crafted items

set unlock mode

close file

# References

1. Source Name - Description
2. …