COS10004 Computer System

Assignment 2 Report

Student name: Kien Quoc Mai
Student Id: 103532920
Lab 17 (Friday 8:30 am EN409)

1. Outline

The program is an implementation of the game Mastermind in ARM assembly.

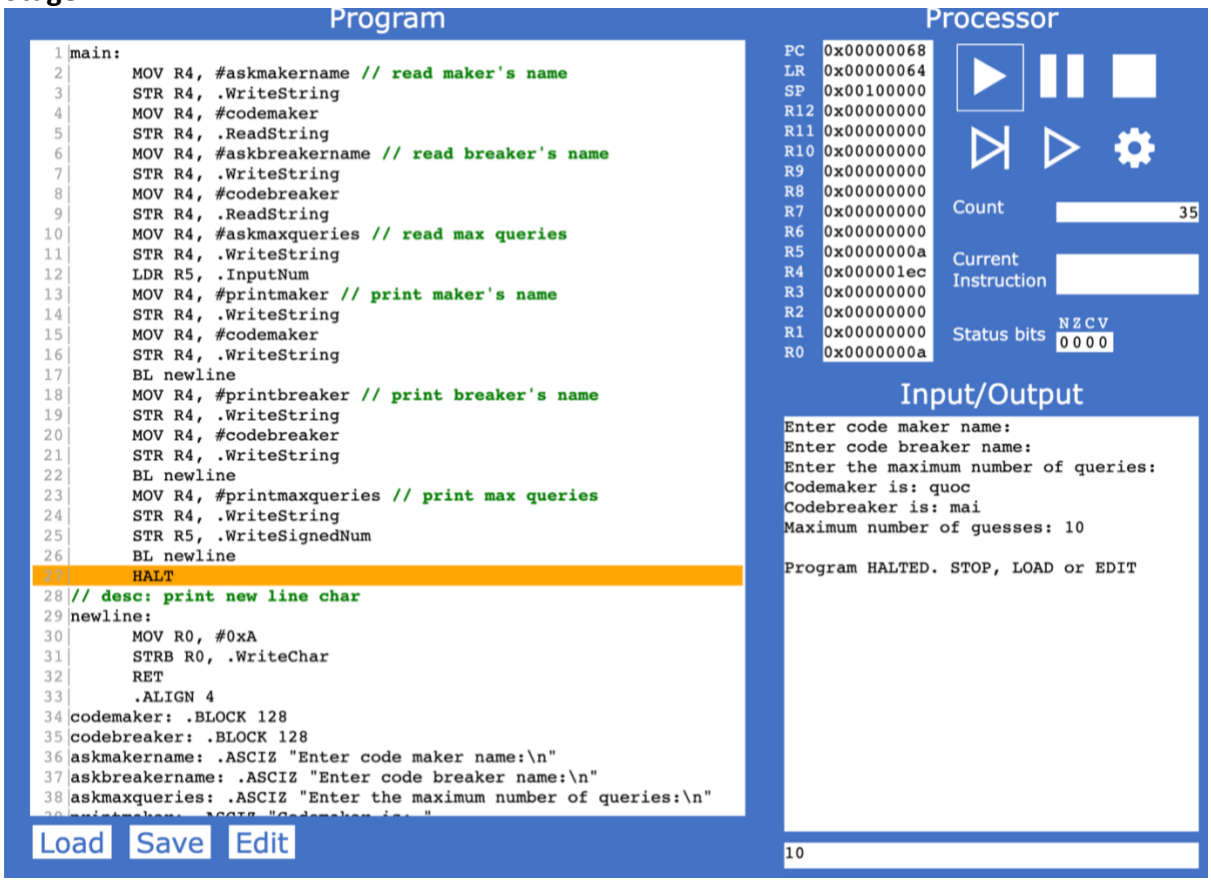
This program consists of multiple functions:

- newline:
  - Description: prints endline character
  - Note: This function reduces code duplication as printing newline characters is used throughout the program
- getcode:
  - Description: reads a code array, validates it and reads again if validation fails
  - Parameters: R0 -> array to store the code
  - Return values: R0 -> array with values from user input
  - Note: There are 2 validations, the code's length must be 4 and contains only "r", "g", "b", "y", "p", and "c". To get the length of a code, a while loop is used to iterate over the string, while counting the number of characters, until a value of 0 is found, which signals the end of the string. To check the second condition, each character is compared with every character stored in the "allowedchars" string.
- comparecodes:
  - Description: compare a query code to the secret code and return feedback
  - Parameters: R0 -> secret array, R1 -> query array
  - Return values: R0 -> number of exact matches, R1 -> number of colour matches
  - Note: Each character in the query code is compared with the corresponding character in the secret code. If they are not exactly matched, a nested loop is used to check for partial matches.
- getcolour:
  - Description: convert a code array to an array of colours
  - Parameters: R0 -> array to store colours, R1 -> code array
  - Return values: R0 -> array containing colours
  - Note: this function converts a code to a colour value using multiple if statements. Ex: "g" character is mapped to "#.green" value
- getresponsecode
  - Description: convert the number of exact and partial matches to an array of colour codes
  - Parameters: R0 -> array to store the code, R1 -> number of exact matches, R2 -> number of partial matches
  - Return values: R0 -> array containing colour codes
  - Ex: 2 exact matches and 1 partial match => "kkwo" ("k" is black, "w" is white, and "o" is the background colour)
- drawline:
  - Description: draw a 4-pixel line at a given coordinate
  - Parameters: R0 -> x coordinate, R1 -> y coordinate, R2 -> array of colours
- displayguess:
  - Description: draw a line representing a query code and another line representing feedback
  - Parameters: R0 -> the current number of guesses, R1 -> array containing query code, R2 -> array containing feedback code

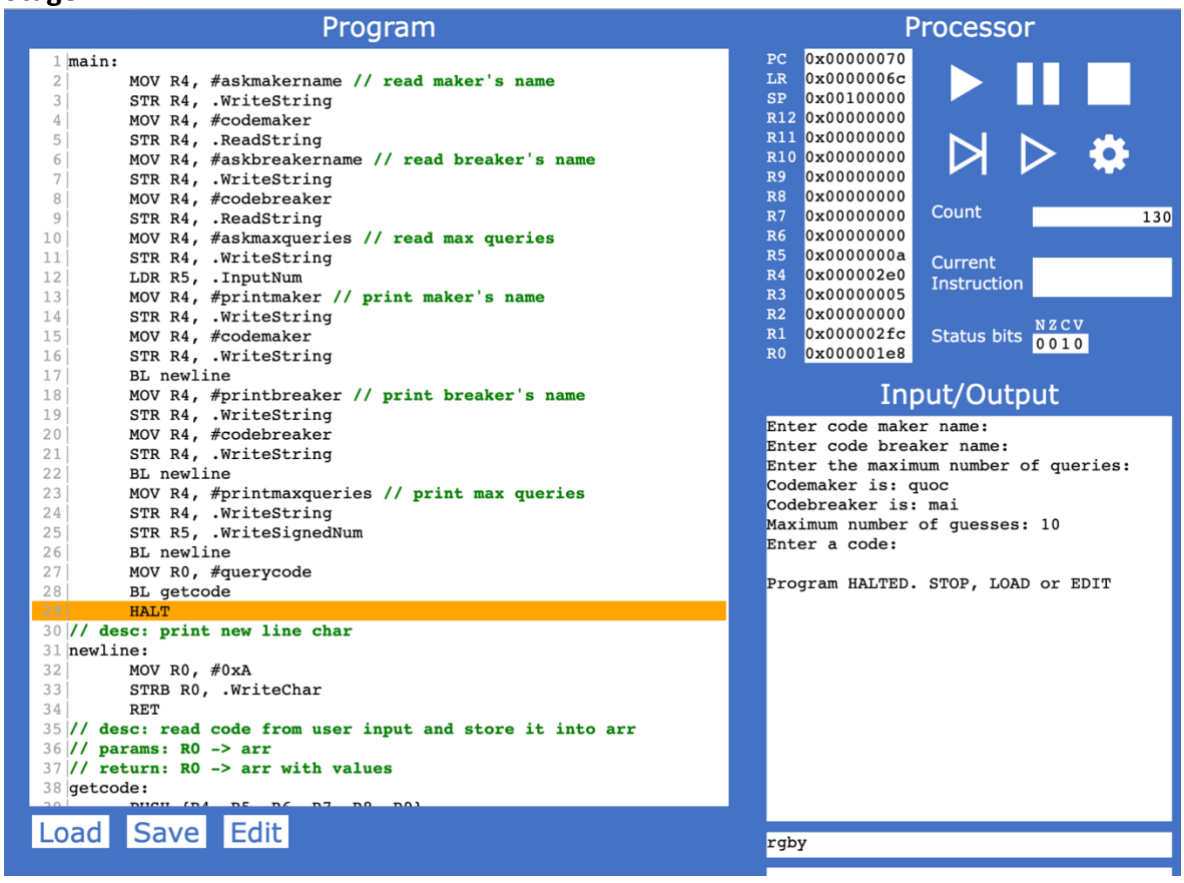
    - Note: this calls the getcolour and the drawline functions
- displayanswer
    - Description: draw a line representing the secret code, which is going to be revealed each time a pin is guessed correctly
    - Parameters: R0 -> array containing the secret code, R1 -> should hide the secret code or not
    - Note: if R1 is 0, the secret code is showed. Otherwise, a black line is displayed.

2. Screenshots

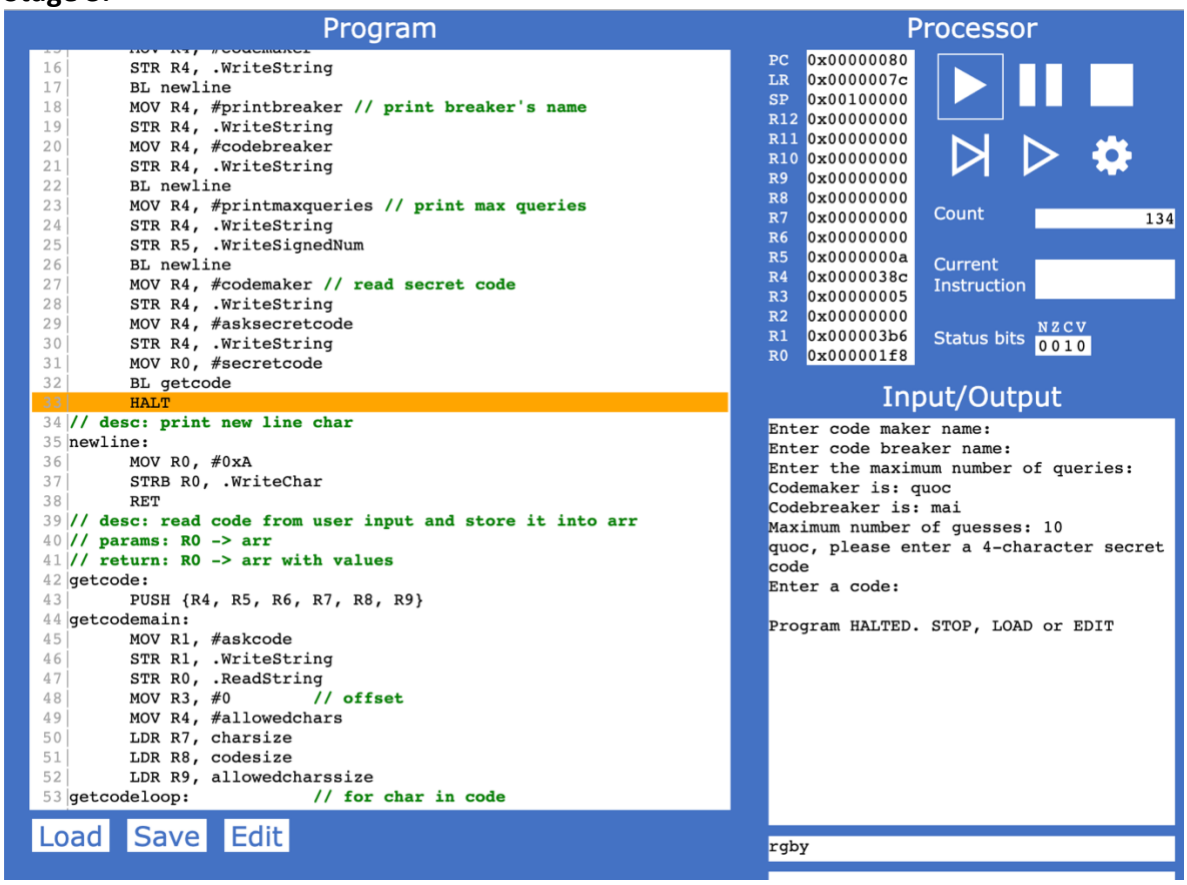
Stage 1:



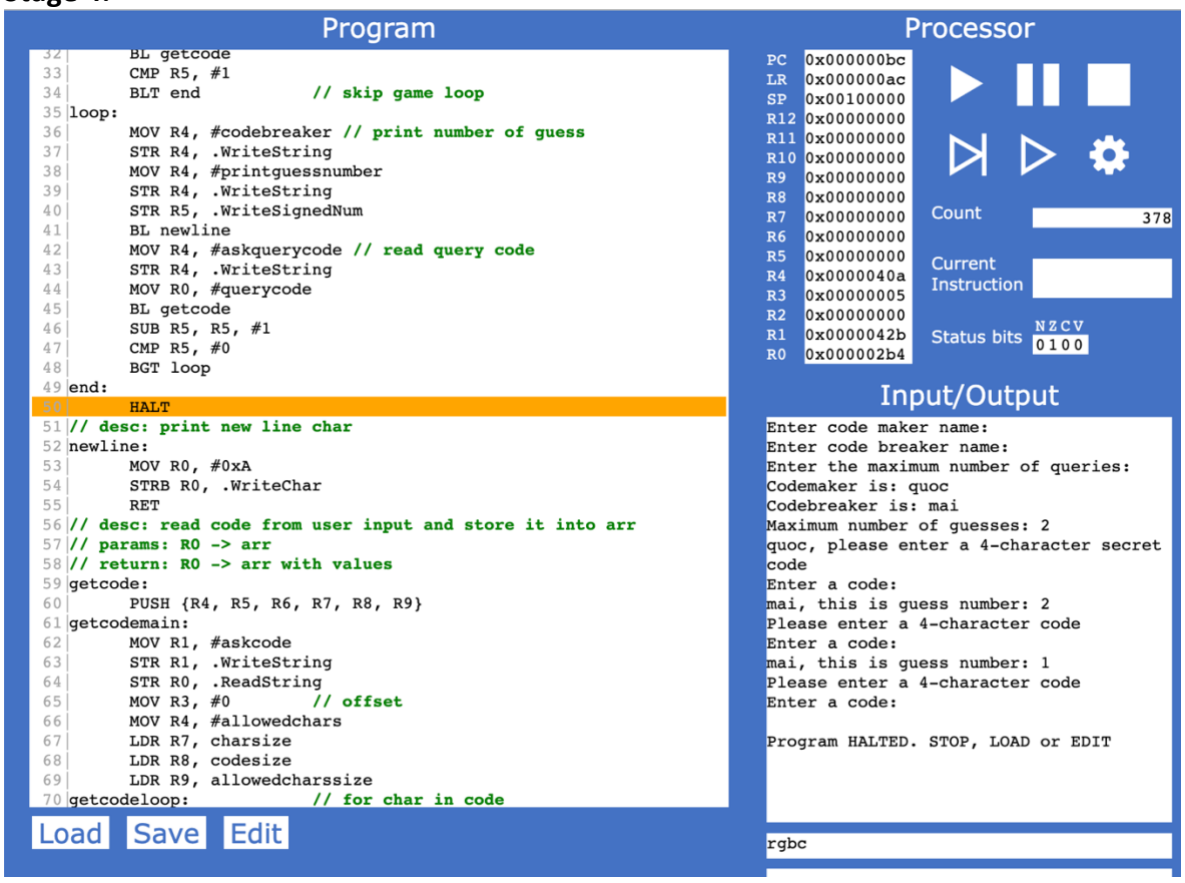
Stage 2:



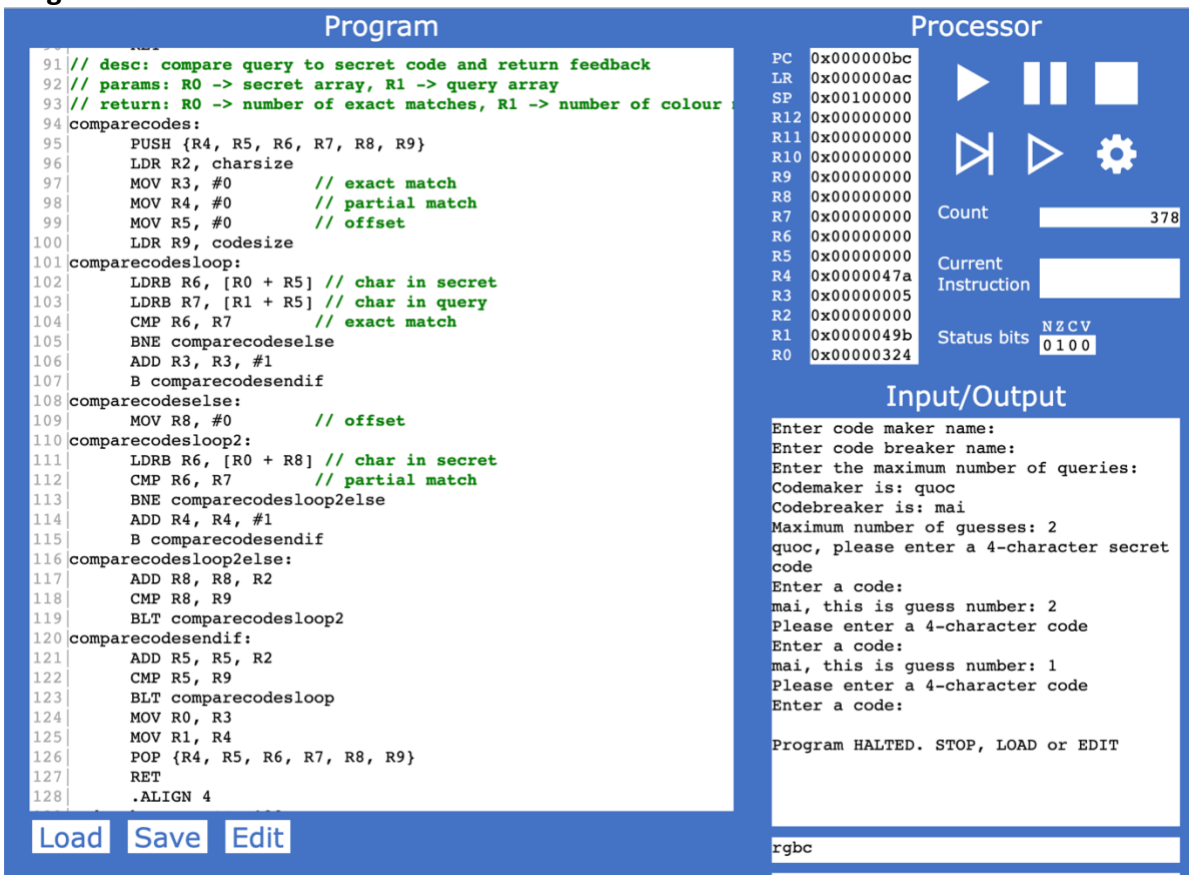
Stage 3:



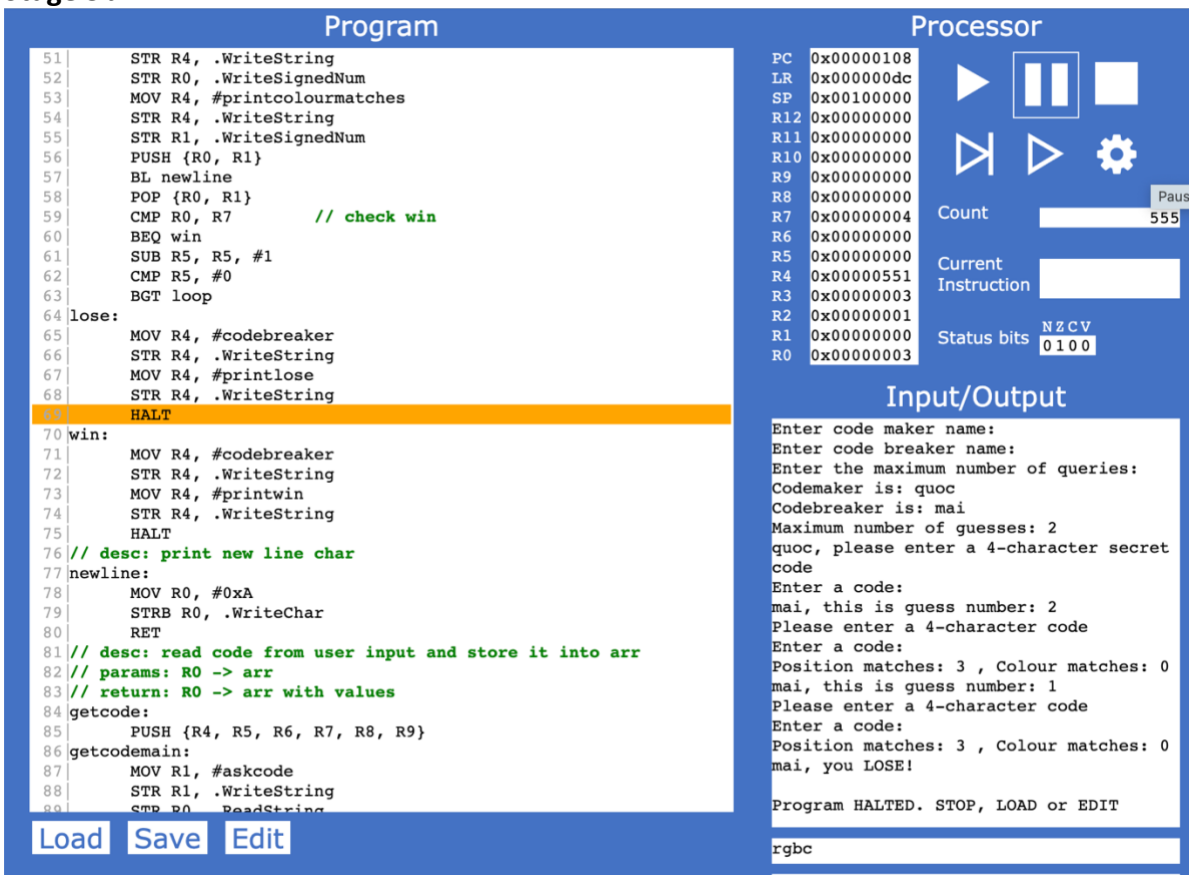
Stage 4:

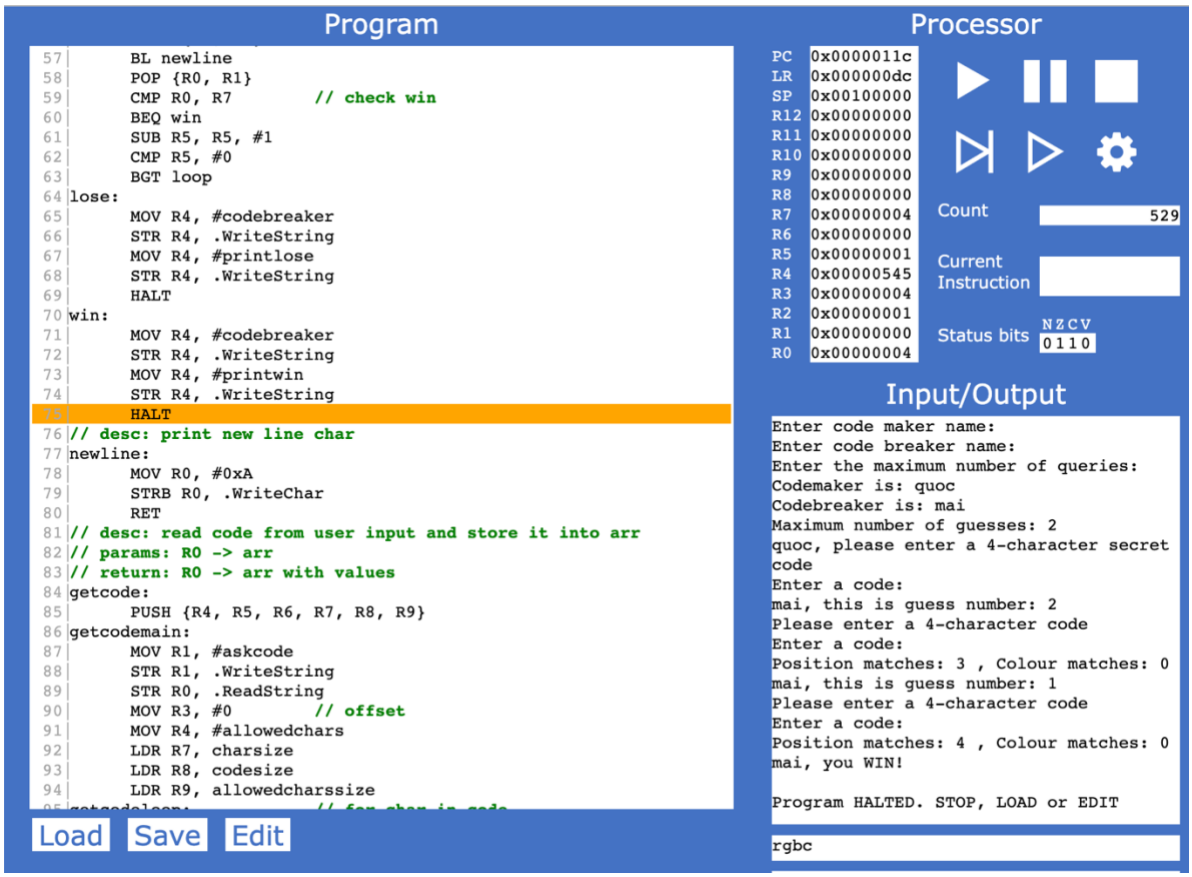


Stage 5a:

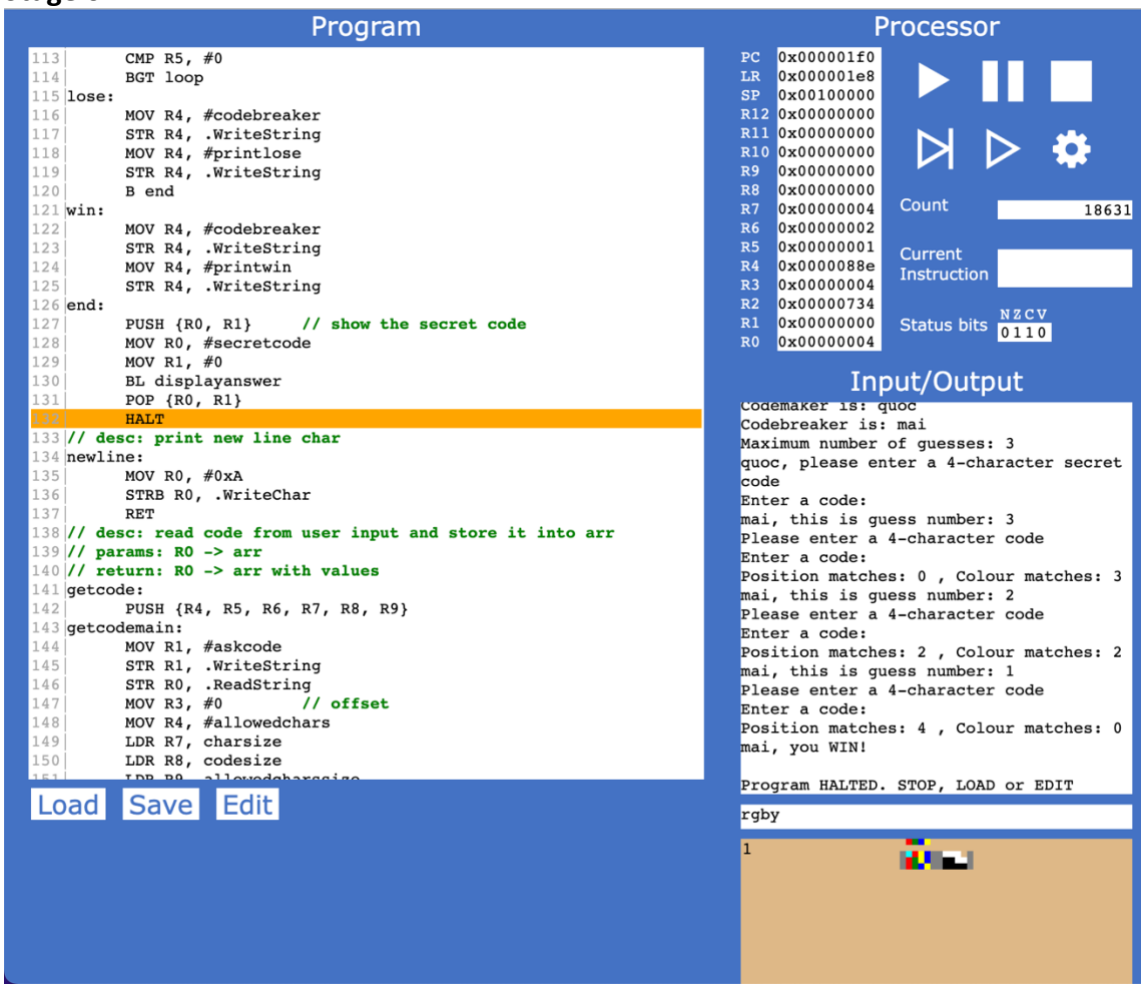


Stage 5b:




Program
Processor
BL newline
POP {R0, R1}
CMP R0, R7 // check win
BEQ win
SUB R5, R5, #1
CMP R5, #0
BGT loop
lose:
MOV R4, #codebreaker
STR R4, .WriteString
MOV R4, #printlose
STR R4, .WriteString
HALT
win:
MOV R4, #codebreaker
STR R4, .WriteString
MOV R4, #printwin
STR R4, .WriteString
HALT
// desc: print new line char
newline:
MOV R0, #0xA
STRB R0, .WriteChar
RET
// desc: read code from user input and store it into arr
// params: R0 -> arr
// return: R0 -> arr with values
getcode:
PUSH {R4, R5, R6, R7, R8, R9}
getcodemain:
MOV R1, #askcode
STR R1, .WriteString
STR R0, .ReadString
MOV R3, #0 // offset
MOV R4, #allowedchars
LDR R7, charsize
LDR R8, codesize
LDR R9, allowedcharssize
PC 0x0000011c
LR 0x000000dc
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000004
R6 0x00000000
R5 0x00000001
R4 0x00000545
R3 0x00000004
R2 0x00000001
R1 0x00000000
R0 0x00000004
Count 529
Current Instruction
Status bits N Z C V 0 1 1 0
Input/Output
Enter code maker name:
Enter code breaker name:
Enter the maximum number of queries:
Codemaker is: quoc
Codebreaker is: mai
Maximum number of guesses: 2
quoc, please enter a 4-character secret code
Enter a code:
mai, this is guess number: 2
Please enter a 4-character code
Enter a code:
Position matches: 3 , Colour matches: 0
mai, this is guess number: 1
Please enter a 4-character code
Enter a code:
Position matches: 4 , Colour matches: 0
mai, you WIN!
Program HALTED. STOP, LOAD or EDIT
rgbc
Load
Save
Edit

Stage 6:


Program
Processor
CMP R5, #0
BGT loop
lose:
MOV R4, #codebreaker
STR R4, .WriteString
MOV R4, #printlose
STR R4, .WriteString
B end
win:
MOV R4, #codebreaker
STR R4, .WriteString
MOV R4, #printwin
STR R4, .WriteString
end:
PUSH {R0, R1} // show the secret code
MOV R0, #secretcode
MOV R1, #0
BL displayanswer
POP {R0, R1}
HALT
// desc: print new line char
newline:
MOV R0, #0xA
STRB R0, .WriteChar
RET
// desc: read code from user input and store it into arr
// params: R0 -> arr
// return: R0 -> arr with values
getcode:
PUSH {R4, R5, R6, R7, R8, R9}
getcodemain:
MOV R1, #askcode
STR R1, .WriteString
STR R0, .ReadString
MOV R3, #0 // offset
MOV R4, #allowedchars
LDR R7, charsize
LDR R8, codesize
PC 0x000001f0
LR 0x000001e8
SP 0x00100000
R12 0x00000000
R11 0x00000000
R10 0x00000000
R9 0x00000000
R8 0x00000000
R7 0x00000004
R6 0x00000002
R5 0x00000001
R4 0x0000088e
R3 0x00000004
R2 0x00000734
R1 0x00000000
R0 0x00000004
Count 18631
Current Instruction
Status bits N Z C V 0 1 1 0
Input/Output
Codemaker is: quoc
Codebreaker is: mai
Maximum number of guesses: 3
quoc, please enter a 4-character secret code
Enter a code:
mai, this is guess number: 3
Please enter a 4-character code
Enter a code:
Position matches: 0 , Colour matches: 3
mai, this is guess number: 2
Please enter a 4-character code
Enter a code:
Position matches: 2 , Colour matches: 2
mai, this is guess number: 1
Please enter a 4-character code
Enter a code:
Position matches: 4 , Colour matches: 0
mai, you WIN!
Program HALTED. STOP, LOAD or EDIT
rgby
1
Load
Save
Edit

3. Assumptions

There is an assumption that all colours in user input's codes are unique. This assumption is also stated in the requirements of this assignment.

4. Unresolved Problems

There is not any unresolved problem, and the program works as expected.