

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 newline 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:

Program

```

1 main:
2     MOV R4, #askMakerName // read maker's name
3     STR R4, .WriteString
4     MOV R4, #codemaker
5     STR R4, .ReadString
6     MOV R4, #askBreakerName // read breaker's name
7     STR R4, .WriteString
8     MOV R4, #codebreaker
9     STR R4, .ReadString
10    MOV R4, #askMaxQueries // read max queries
11    STR R4, .WriteString
12    LDR R5, .InputNum
13    MOV R4, #printMaker // print maker's name
14    STR R4, .WriteString
15    MOV R4, #codemaker
16    STR R4, .WriteString
17    BL newline
18    MOV R4, #printBreaker // print breaker's name
19    STR R4, .WriteString
20    MOV R4, #codebreaker
21    STR R4, .WriteString
22    BL newline
23    MOV R4, #printMaxQueries // print max queries
24    STR R4, .WriteString
25    STR R5, .WriteSignedNum
26    BL newline
27    HALT
28 // desc: print new line char
29 newline:
30     MOV R0, #0xA
31     STRB R0, .WriteChar
32     RET
33 codemaker: .BLOCK 128
34 codebreaker: .BLOCK 128
35 askMakerName: .ASCIIZ "Enter code maker name:\n"
36 askBreakerName: .ASCIIZ "Enter code breaker name:\n"
37 askMaxQueries: .ASCIIZ "Enter the maximum number of queries:\n"
38 printMaker: .ASCIIZ "Codemaker is: "
39 printBreaker: .ASCIIZ "Codebreaker is: "

```

Load
Save
Edit

Processor

PC	0x00000068
LR	0x00000064
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000000
R6	0x00000000
R5	0x0000000a
R4	0x000001ec
R3	0x00000000
R2	0x00000000
R1	0x00000000
R0	0x0000000a

Count 35

Current Instruction

Status bits NZCV 0000

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: 10

10

Stage 2:

Program

```

1 main:
2     MOV R4, #askMakerName // read maker's name
3     STR R4, .WriteString
4     MOV R4, #codemaker
5     STR R4, .ReadString
6     MOV R4, #askBreakerName // read breaker's name
7     STR R4, .WriteString
8     MOV R4, #codebreaker
9     STR R4, .ReadString
10    MOV R4, #askMaxQueries // read max queries
11    STR R4, .WriteString
12    LDR R5, .InputNum
13    MOV R4, #printMaker // print maker's name
14    STR R4, .WriteString
15    MOV R4, #codemaker
16    STR R4, .WriteString
17    BL newline
18    MOV R4, #printBreaker // print breaker's name
19    STR R4, .WriteString
20    MOV R4, #codebreaker
21    STR R4, .WriteString
22    BL newline
23    MOV R4, #printMaxQueries // print max queries
24    STR R4, .WriteString
25    STR R5, .WriteSignedNum
26    BL newline
27    MOV R0, #querycode
28    BL getcode
29    HALT
30 // desc: print new line char
31 newline:
32     MOV R0, #0xA
33     STRB R0, .WriteChar
34     RET
35 // desc: read code from user input and store it into arr
36 // params: R0 -> arr
37 // return: R0 -> arr with values
38 getcode:
39     PUSH {R3, R4, R5, R6, R7, R8, R9}

```

Load
Save
Edit

Processor

PC	0x00000070
LR	0x0000006c
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000000
R6	0x00000000
R5	0x0000000a
R4	0x000002e0
R3	0x00000000
R2	0x00000000
R1	0x000002fc
R0	0x000001e8

Count 130

Current Instruction

Status bits NZCV 0010

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: 10

Enter a code:

Program HALTED. STOP, LOAD or EDIT

rgby

Stage 3:

Program

```

6      MOV R4, #askBreakerName // read breaker's name
7      STR R4, .WriteString
8      MOV R4, #codebreaker
9      STR R4, .ReadString
10     MOV R4, #askMaxQueries // read max queries
11     STR R4, .WriteString
12     LDR R5, .InputNum
13     MOV R4, #printMaker // print maker's name
14     STR R4, .WriteString
15     MOV R4, #codemaker
16     STR R4, .WriteString
17     BL newline
18     MOV R4, #printBreaker // print breaker's name
19     STR R4, .WriteString
20     MOV R4, #codebreaker
21     STR R4, .WriteString
22     BL newline
23     MOV R4, #printMaxQueries // print max queries
24     STR R4, .WriteString
25     STR R5, .WriteSignedNum
26     BL newline
27     MOV R4, #codemaker // read secret code
28     STR R4, .WriteString
29     MOV R4, #askSecretCode
30     STR R4, .WriteString
31     MOV R0, #secretcode
32     BL getcode
33     HALT
34 // desc: print new line char
35 newline:
36     MOV R0, #0xA
37     STRB R0, .WriteChar
38     RET
39 // desc: read code from user input and store it into arr
40 // params: R0 -> arr
41 // return: R0 -> arr with values
42 getcode:
43     PUSH {R3, R4, R5, R6, R7, R8, R9}

```

Load
Save
Edit

Processor

PC	0x00000080
LR	0x0000007c
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000000
R6	0x00000000
R5	0x0000000a
R4	0x0000038c
R3	0x00000000
R2	0x00000000
R1	0x000003b6
R0	0x000001f8

Count 134

Current Instruction

Status bits NZCV
0010

Input/Output

Codebreaker is: mai

Maximum number of guesses: 10

quoc, please enter a 4-character secret code

Enter a code:

Program HALTED. STOP, LOAD or EDIT

rgby

Stage 4:

Program

```

32     BL getcode
33     CMP R5, #1
34     BLT end // skip game loop
35 loop:
36     MOV R4, #codebreaker // print number of guess
37     STR R4, .WriteString
38     MOV R4, #printGuessNumber
39     STR R4, .WriteString
40     STR R5, .WriteSignedNum
41     BL newline
42     MOV R4, #askQueryCode // read query code
43     STR R4, .WriteString
44     MOV R0, #querycode
45     BL getcode
46     SUB R5, R5, #1
47     CMP R5, #0
48     BGT loop
49 end:
50     HALT
51 // desc: print new line char
52 newline:
53     MOV R0, #0xA
54     STRB R0, .WriteChar
55     RET
56 // desc: read code from user input and store it into arr
57 // params: R0 -> arr
58 // return: R0 -> arr with values
59 getcode:
60     PUSH {R3, R4, R5, R6, R7, R8, R9}
61 getcodeMain:
62     MOV R1, #askCode
63     STR R1, .WriteString
64     STR R0, .ReadString
65     MOV R3, #0 // offset
66     MOV R4, #allowedChars
67     LDR R7, charSize
68     LDR R8, codeSize
69     LDR R9, allowedCharsSize
70 getcodeLoop: // for char in code

```

Load
Save
Edit

Processor

PC	0x000000bc
LR	0x000000ac
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000000
R6	0x00000000
R5	0x00000000
R4	0x0000040a
R3	0x00000000
R2	0x00000000
R1	0x0000042b
R0	0x000002b4

Count 378

Current Instruction

Status bits NZCV
0100

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:

mai, this is guess number: 1

Please enter a 4-character code

Enter a code:

Program HALTED. STOP, LOAD or EDIT

rgbc

Stage 5a:

Program

```

91 // desc: compare query to secret code and return feedback
92 // params: R0 -> secret array, R1 -> query array
93 // return: R0 -> number of exact matches, R1 -> number of colour matches
94 comparecodes:
95     PUSH {R3, R4, R5, R6, R7, R8, R9}
96     LDR R2, charSize
97     MOV R3, #0 // exact match
98     MOV R4, #0 // partial match
99     MOV R5, #0 // offset
100    LDR R9, codeSize
101 comparecodesLoop:
102    LDRB R6, [R0 + R5] // char in secret
103    LDRB R7, [R1 + R5] // char in query
104    CMP R6, R7 // exact match
105    BNE comparecodesElse
106    ADD R3, R3, #1
107    B comparecodesEndIf
108 comparecodesElse:
109    MOV R8, #0 // offset
110 comparecodesLoop2:
111    LDRB R6, [R0 + R8] // char in secret
112    CMP R6, R7 // partial match
113    BNE comparecodesLoop2Else
114    ADD R4, R4, #1
115    B comparecodesEndIf
116 comparecodesLoop2Else:
117    ADD R8, R8, R2
118    CMP R8, R9
119    BLT comparecodesLoop2
120 comparecodesEndIf:
121    ADD R5, R5, R2
122    CMP R5, R9
123    BLT comparecodesLoop
124    MOV R0, R3
125    MOV R1, R4
126    POP {R3, R4, R5, R6, R7, R8, R9}
127    RET
128 codemaker: .BLOCK 128

```

Load
Save
Edit

Processor

PC	0x000000bc
LR	0x000000ac
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000000
R6	0x00000000
R5	0x00000000
R4	0x0000047a
R3	0x00000000
R2	0x00000000
R1	0x0000049b
R0	0x00000324

Count 378

Current Instruction [Empty]

Status bits NZCV 0100

Input/Output

Enter code maker name: 0x00160

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:

mai, this is guess number: 1

Please enter a 4-character code

Enter a code:

Program HALTED. STOP, LOAD or EDIT

rgbc

Stage 5b:

Program

```

51 STR R4, .WriteString
52 STR R0, .WriteSignedNum
53 MOV R4, #printColourMatches
54 STR R4, .WriteString
55 STR R1, .WriteSignedNum
56 PUSH {R0, R1}
57 BL newline
58 POP {R0, R1}
59 CMP R0, R7 // check win
60 BEQ win
61 SUB R5, R5, #1
62 CMP R5, #0
63 BGT loop
64 lose:
65 MOV R4, #codebreaker
66 STR R4, .WriteString
67 MOV R4, #printLose
68 STR R4, .WriteString
69 HALT
70 win:
71 MOV R4, #codebreaker
72 STR R4, .WriteString
73 MOV R4, #printWin
74 STR R4, .WriteString
75 HALT
76 // desc: print new line char
77 newline:
78 MOV R0, #0xA
79 STRB R0, .WriteChar
80 RET
81 // desc: read code from user input and store it into arr
82 // params: R0 -> arr
83 // return: R0 -> arr with values
84 getcode:
85 PUSH {R3, R4, R5, R6, R7, R8, R9}
86 getcodeMain:
87 MOV R1, #askCode
88 STR R1, .WriteString
89 STR R0, .ReadString

```

Load
Save
Edit

Processor

PC	0x00000108
LR	0x000000dc
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000004
R6	0x00000000
R5	0x00000000
R4	0x00000551
R3	0x00000000
R2	0x00000001
R1	0x00000001
R0	0x00000002

Count 583

Current Instruction [Empty]

Status bits NZCV 0100

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: 2 , Colour matches: 1

mai, you LOSE!

Program HALTED. STOP, LOAD or EDIT

gcby

Code Editor

```

57     BL newline
58     POP {R0, R1}
59     CMP R0, R7          // check win
60     BEQ win
61     SUB R5, R5, #1
62     CMP R5, #0
63     BGT loop
64 lose:
65     MOV R4, #codebreaker
66     STR R4, .WriteString
67     MOV R4, #printLose
68     STR R4, .WriteString
69     HALT
70 win:
71     MOV R4, #codebreaker
72     STR R4, .WriteString
73     MOV R4, #printWin
74     STR R4, .WriteString
75     HALT
76 // desc: print new line char
77 newline:
78     MOV R0, #0xA
79     STRB R0, .WriteChar
80     RET
81 // desc: read code from user input and store it into arr
82 // params: R0 -> arr
83 // return: R0 -> arr with values
84 getCode:
85     PUSH {R3, R4, R5, R6, R7, R8, R9}
86 getCodeMain:
87     MOV R1, #askCode
88     STR R1, .WriteString
89     STR R0, .ReadString
90     MOV R3, #0           // offset
91     MOV R4, #allowedChars
92     LDR R7, charSize
93     LDR R8, codeSize
94     LDR R9, allowedCharsSize
95 getCodeLoop:             // for char in code

```

Processor

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	0x00000000
R2	0x00000001
R1	0x00000000
R0	0x00000004

Count: 517

Current Instruction:

Status bits: NZCV 0110

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

Load Save Edit

rgby

Program

```

113|      CMP R5, #0
114|      BGT loop
115|lose:
116|      MOV R4, #codebreaker
117|      STR R4, .WriteString
118|      MOV R4, #printLose
119|      STR R4, .WriteString
120|      B end
121|win:
122|      MOV R4, #codebreaker
123|      STR R4, .WriteString
124|      MOV R4, #printWin
125|      STR R4, .WriteString
126|end:
127|      PUSH {R0, R1}      // show the secret code
128|      MOV R0, #secretcode
129|      MOV R1, #0
130|      BL displayAnswer
131|      POP {R0, R1}
132|      HALT
133|// desc: print new line char
134|newline:
135|      MOV R0, #0xA
136|      STRB R0, .WriteChar
137|      RET
138|// desc: read code from user input and store it into arr
139|// params: R0 -> arr
140|// return: R0 -> arr with values
141|getcode:
142|      PUSH {R3, R4, R5, R6, R7, R8, R9}
143|getcodeMain:
144|      MOV R1, #askCode
145|      STR R1, .WriteString
146|      STR R0, .ReadString
147|      MOV R3, #0          // offset
148|      MOV R4, #allowedChars
149|      LDR R7, charSize
150|      LDR R8, codeSize
151|      LDR R9, allowedCharsSize

```

Load
Save
Edit

Processor

PC	0x000001f0
LR	0x000001e8
SP	0x00100000
R12	0x00000000
R11	0x00000000
R10	0x00000000
R9	0x00000000
R8	0x00000000
R7	0x00000004
R6	0x00000002
R5	0x00000001
R4	0x00000896
R3	0xffff3494
R2	0x0000073c
R1	0x00000000
R0	0x00000004

Count 18637

Current Instruction

Status bits NZCV
0110

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

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.