

Part 1:

1. From the left to right column in the picture below, the first column in the processors tab contains all the running programs and applications running on the computer.
 - a. The third column counting from the left in the processors tab shows how much memory is used. The 71% shown above the memory column is the total physical memory used by active processors. The yellow cells under the memory column show how much memory each application is using. Memory tab under the performance tab shows the memory usage which is related to Random Access Memory (RAM). RAM is the fastest way to store temporary data and to retrieve it if required. RAM is not permanent and will be lost as soon as the computer is shut down. RAM is part of memory and if it's full, the computer will turn to disk and will respond slower than usual as disk is located further away from CPU compared to RAM. Disk is hard drive memory.
 - b. The second column counting from the left in the processors tab shows the CPU usage. The 43% shown above the CPU column is the total processor utilisation across all the 8 cores this laptop has. The yellow cells under the CPU column show the percentage of processor activity falls under each application. The CPU tab under the performance tab shows CPU utilisation percentage. When an application is just starting to run, the CPU usage would be high, the graph shown in that tab will increase and peak then decrease as it will initially use more processors temporarily to start the application as shown in the sixth cell under the CPU column when I screenshotted the screen when I opened the application. Processors will retrieve and use certain information and data stored in RAM to execute specific programs so if more programs are opened, processor activity will rise and memory usage will increase due to more information and data being stored in RAM.
2. I did not expect to find Client Session Agent to be running as a background processor. Client Session Agent is part of Trend Micro Anti-Malware Solution Platform which is under an application I have installed called Trend Micro Maximum Security which is an antivirus software that also provides protection for my computer against evolving malware infections and threats. Trend Micro Maximum Security protects my computer from ransomware and potentially harmful websites by blocking access to some suspicious and unsecured websites. Of course if I really want to access these specific websites, I can as there is an option to allow me to do so. Trend Micro Maximum Security will not show any pop-ups notifications on my screen unless it detects suspicious activity from me. Trend Micro Maximum Security also performs regular security scanning on my computer and the files present in the computer to ensure that my computer is malware and threats free. The application will also inform me 5 minutes before their regular scanning to inform the user beforehand but the scanning will not affect the user from using their device. Hence, to perform what the application was designed to do, the application will always be running in the background as soon as the computer is on.
3. Operating systems can easily locate the files for the user in the computer system, so it's user-friendly and the files under the operating system would also need to obey certain rules and reach a certain security threshold for the computer to interact with the specific application. If it was the application itself managing their files themselves, malware attacks could happen anytime and infect the user's computer which will then expose the user's files and personal information like bank account information.

Task Manager									
File Options View									
Processes Performance App history Startup Users Details Services									
Name	Status	43% CPU	71% Memory	0% Disk	0% Network utilization	8% GPU utilization	GPU engine	Power usage	Power usage tr...
Apps (6)									
> Canon Quick Menu (32 bit) (3)		0%	28.2 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Canon Quick Menu Image Displ...		0%	11.6 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Google Chrome (4)		2.9%	204.8 MB	0 MB/s	0 Mbps	0.8%	GPU 0 - 3D	Low	Very low
> Microsoft Teams (9)		0%	442.1 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Spotify (6)		0%	122.0 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Task Manager		16.6%	25.5 MB	0 MB/s	0 Mbps	0%		Very high	Low
Background processes (119)									
> ACCStd		0%	8.2 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> ACCSvc		0%	0.6 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> aeg_launcher		0%	0.4 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> AggregatorHost		0%	0.7 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Application Frame Host		0%	9.2 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Brother MFC Windows Software...		0%	1.3 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> BrNCSvc (32 bit)		0%	2.7 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Canon IJ Network Scanner Selec...		0%	0.8 MB	0 MB/s	0 Mbps	0%		Very low	Very low
> Client Session Agent		0%	0.2 MB	0 MB/s	0 Mbps	0%		Very low	Very low
Fewer details									
End task									

Part 2:

Test Case for Task 2.2:

1. String = fOokAiYaN!

Assembly code:

```

25
26 myName, ADR Name
27 Name, HEX 066 /f
28 HEX 04F /o
29 HEX 06F /o
30 HEX 06B /k
31 HEX 041 /A
32 HEX 069 /i
33 HEX 059 /Y
34 HEX 061 /a
35 HEX 04E /N
36 HEX 021 /! as my 10th character
37 HEX 000 / null
38
39 PrintFrom, HEX 0 / Address to start printing from
40 PrintString, HEX 0 / Return address placeholder
41 PrintLoop, LoadI PrintFrom
42 Skipcond 800

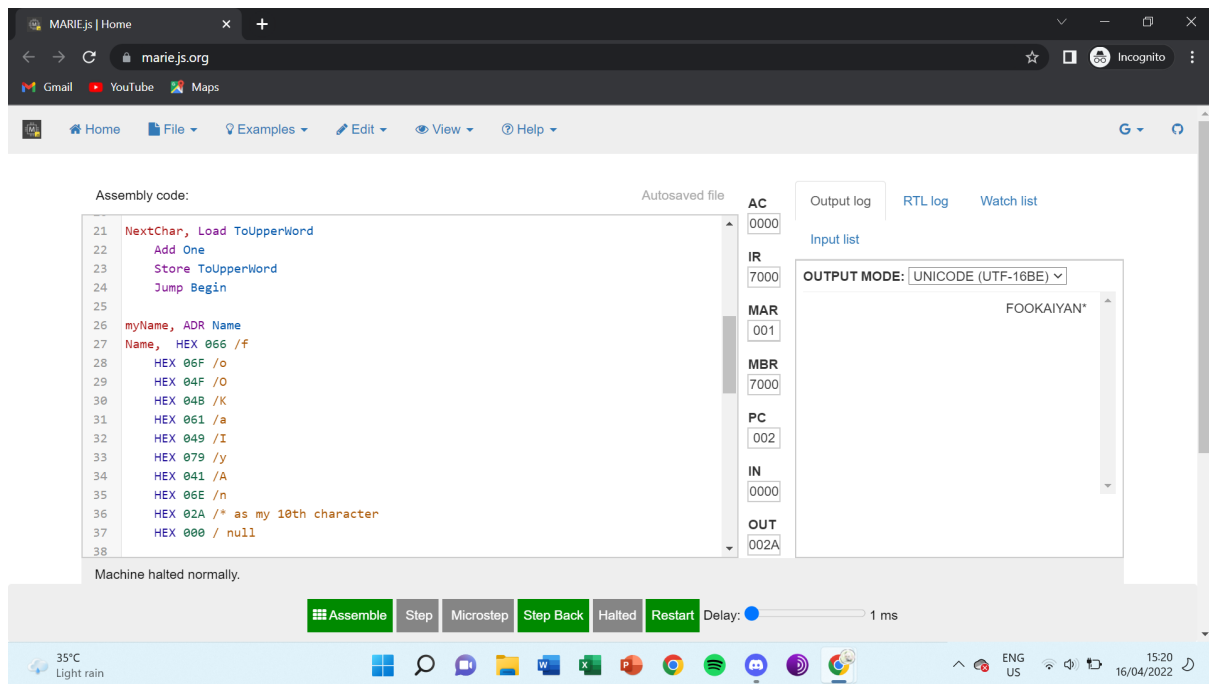
```

Machine halted normally.

Output log: FOOKAIYAN!

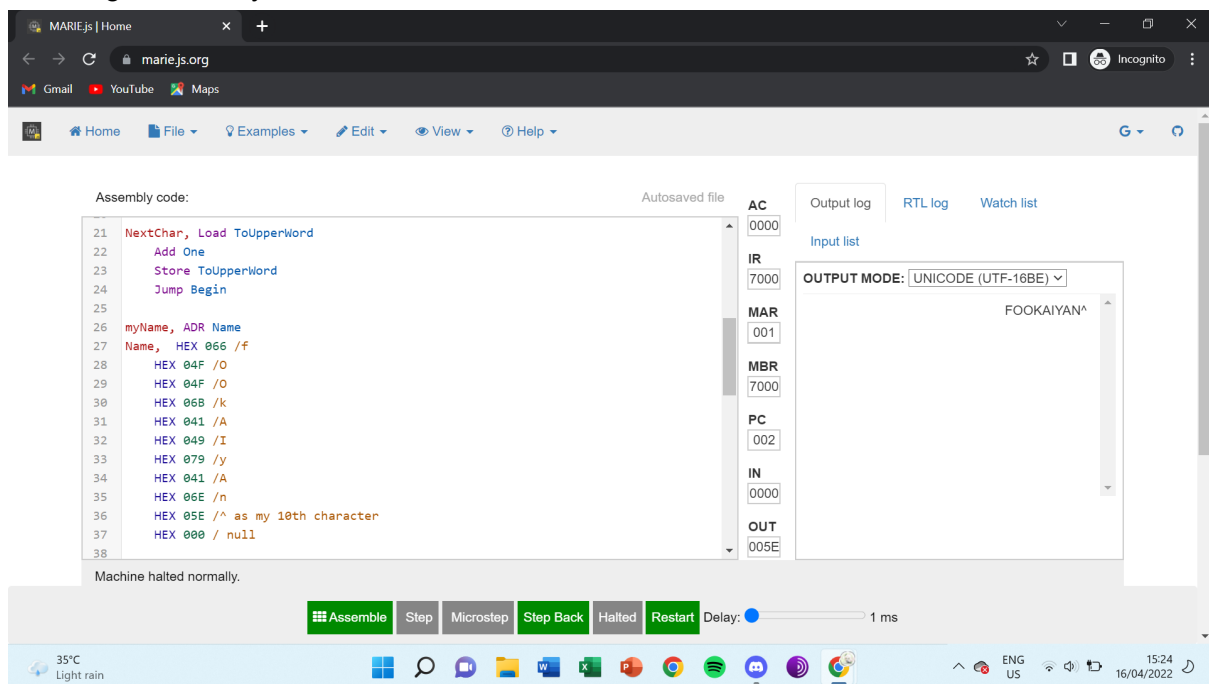
This string tests the Check subroutine on the position of the upper and lower case of characters.

2. String = foOKalyAn*



This string tests the Check subroutine on the position of the upper and lower case of characters.

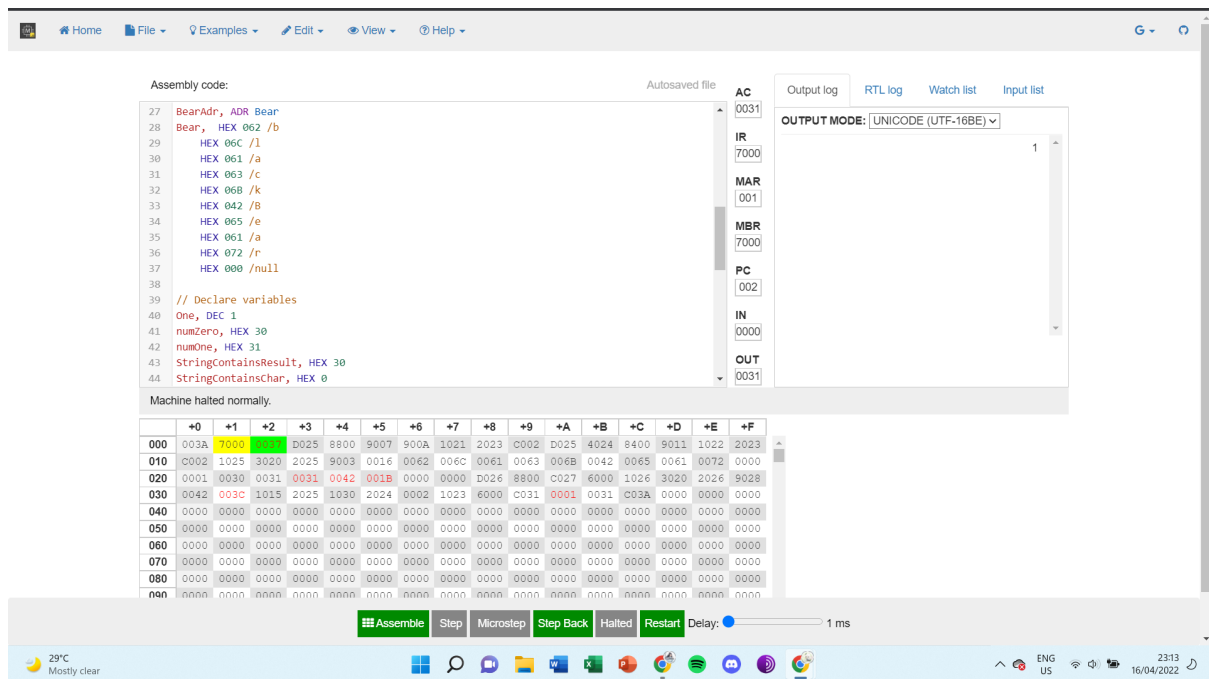
3. String = fOOkAIyAn^



This string tests the Check subroutine on the position of the upper and lower case of characters.

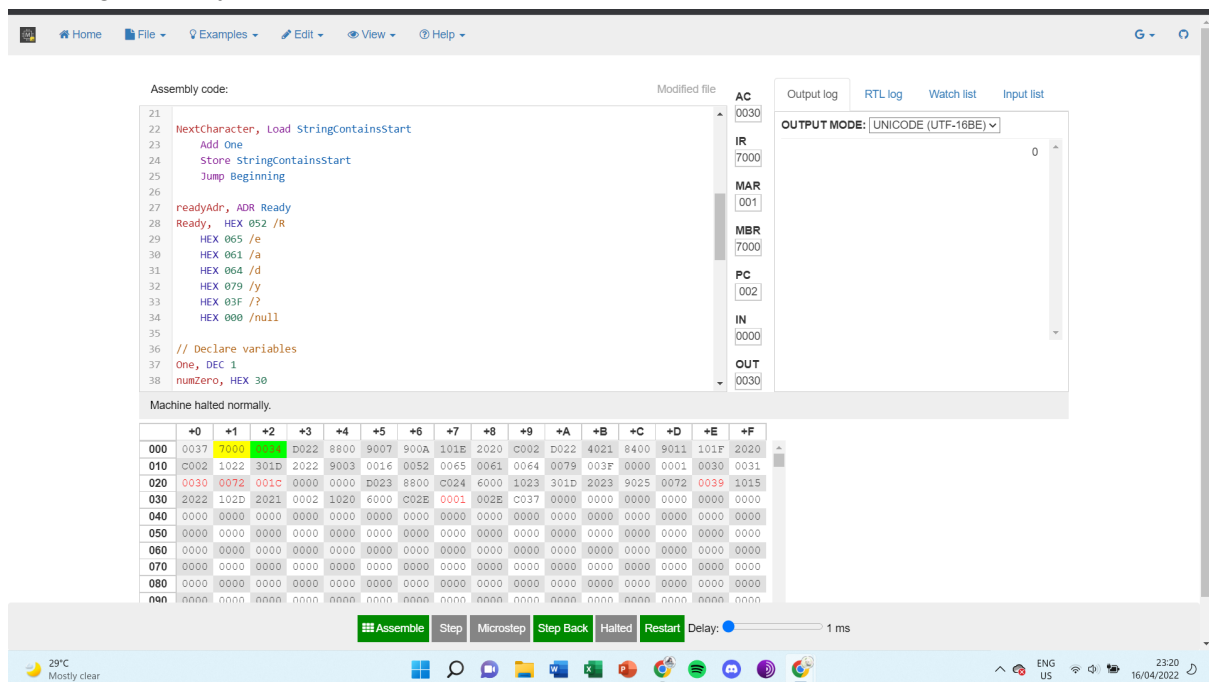
Test Case for Task 2.3:

1. String = blackBear ; Character to be found = B



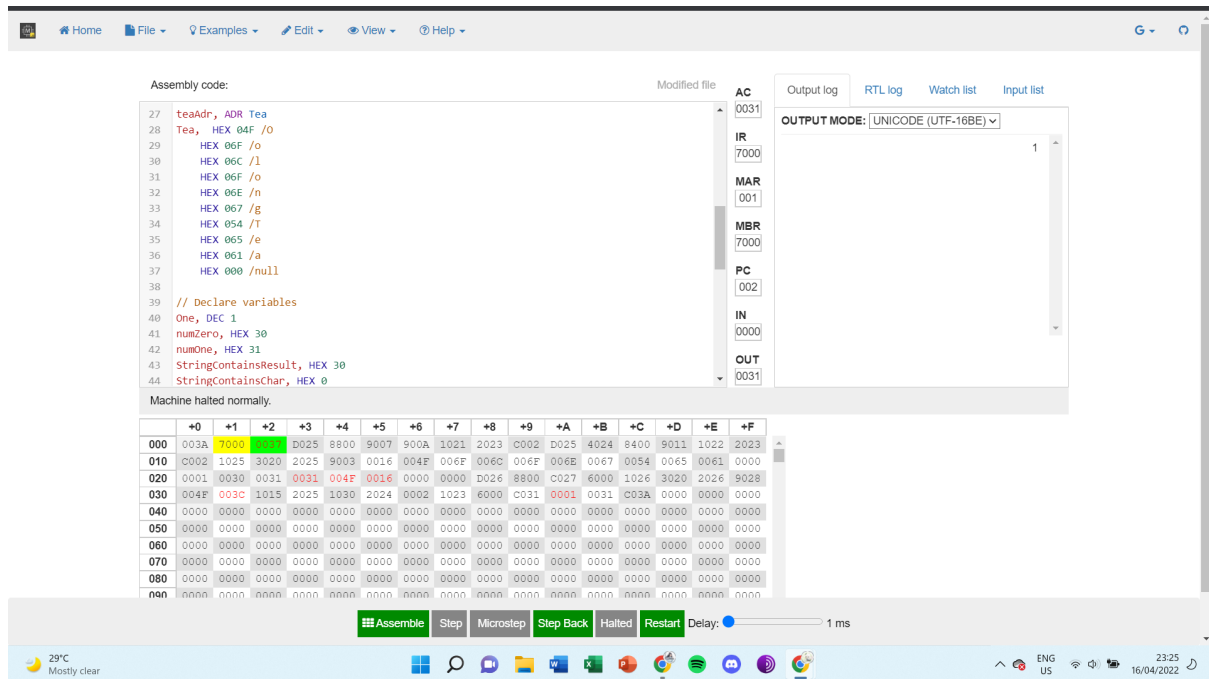
This string tests to see if the lowercase of the character is detected as present or absent and if it will continue on to loop through the characters present in the string to detect the uppercase B.

2. String = Ready? ; Character to be found = r



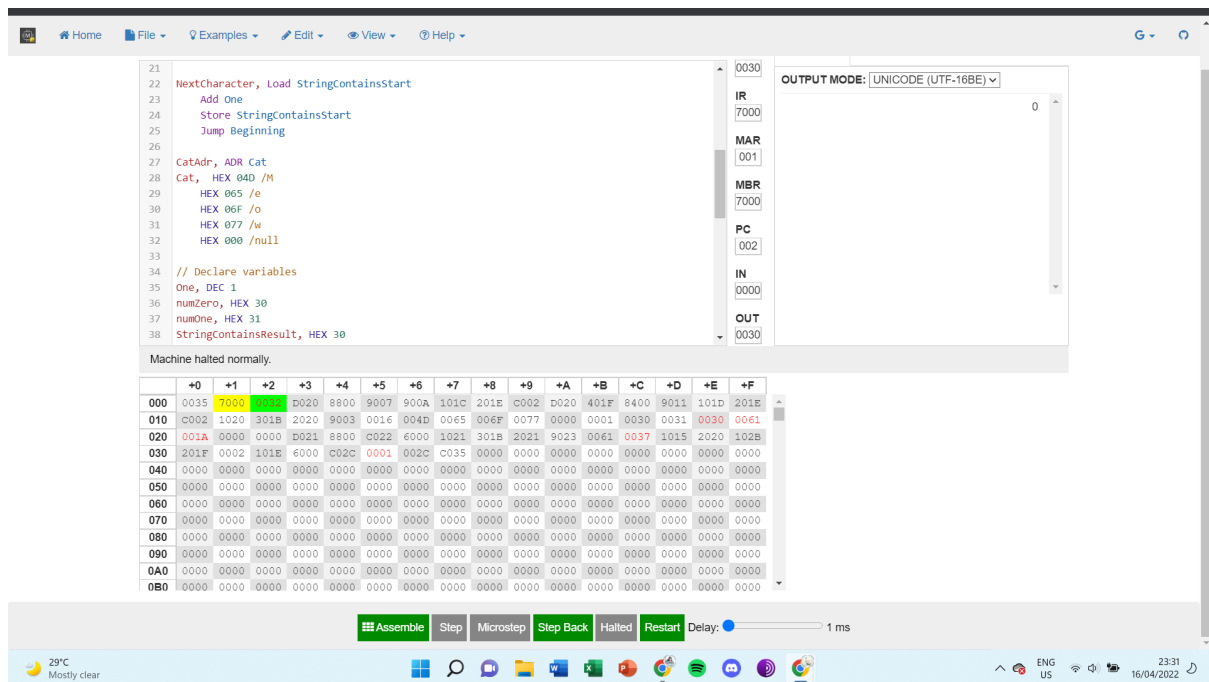
This string tests to see if the uppercase of the character is detected as present or absent.

3. String = OolongTea ; Character to be found = O



This string tests to see if the character is detected as present or absent and if it will continue on to loop through the characters present in the string.

4. String = Meow ; Character to be found = a



This string tests to see if the checking subroutine can detect a character not found in the string.

Test Case for Task 2.4:

1. Guessed_Word = Cat ; Selected_Word = Words

Assembly code:

```

29 // FOR ERROR MESSAGE
30 TooLong, LoadI ErrorLongAdr
31   Skipcond 800 /check if AC > 0
32   JumpI checkGuess
33   Output
34   Load ErrorLongAdr
35   Add One
36   Store ErrorLongAdr
37   Jump TooLong
38 // FOR ERROR MESSAGE
39 TooShort, LoadI ErrorShortAdr
40   Skipcond 800 /check if AC > 0
41   JumpI checkGuess
42   Output
43   Load ErrorShortAdr
44   Add One
45   Store ErrorShortAdr
46   Jump TooShort

```

Machine halted normally.

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	008B	7003	0000	D04A	8800	8400	9009	900D	9011	D049	8400	9019	9021	108E	303F	208E
010	9011	D063	8800	C002	6000	1063	303F	2063	9011	D04B	8800	C002	6000	104B	303F	204B
020	9019	D047	2045	8800	D048	2046	4045	8400	C02A	6000	0000	D042	8800	902F	9032	1040
030	2043	C02A	D042	4046	8400	903A	1046	3041	2043	C02A	1042	303F	2042	902B	0005	0001
040	005F	0020	0000	0000	0000	0000	0000	008F	0097	0097	008F	0062	0059	006F	0075	0072
050	0020	0077	006F	0072	0064	0020	0069	0073	0020	0074	006F	006F	0020	0073	0068	006F
060	0072	0074	0000	0064	0059	006F	0075	0072	0020	0077	006F	0072	0064	0020	0069	0073
070	0020	0074	006F	006F	0020	006C	006F	006E	0067	0000	0000	0000	D07A	8800	C07B	6000
080	107A	303F	207A	907C	008D	108E	2047	1096	2048	0002	C084	0001	0084	C08B	008F	0047
090	0059	0045	0041	0054	0000	0097	0097	0057	004F	0052	0044	0053	0000	0000	0000	0000

AC: 0000, IR: 7000, MAR: 001, MBR: 7000, PC: 002, IN: 0000, OUT: 0074

Buttons: Assemble, Step, Microstep, Step Back, Halted, Restart. Delay: 1 ms

If guessed word shorter than selected work then output “Your word is too short”

2. Guessed_Word = Kitten ; Selected_Word = Words

Assembly code:

```

26 TooLong, LoadI ErrorLongAdr
27   Skipcond 800 /check if AC > 0
28   JumpI checkGuess
29   Output
30   Load ErrorLongAdr
31   Add One
32   Store ErrorLongAdr
33   Jump TooLong
34 // FOR ERROR MESSAGE
35 TooShort, LoadI ErrorShortAdr
36   Skipcond 800 /check if AC > 0
37   JumpI checkGuess
38   Output
39   Load ErrorShortAdr
40   Add One
41   Store ErrorShortAdr
42   Jump TooShort
43

```

Machine halted normally.

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	008C	7000	0000	D04B	8800	9007	9012	8400	900E	D04A	8400	901A	9022	108F	3040	
010	208F	9012	D064	8800	C002	6000	1064	3040	2064	9012	D04C	8800	C002	6000	104C	3040
020	204C	901A	D048	2046	8800	D049	2047	4046	8400	C02B	6000	0000	D043	8800	9030	9033
030	1041	2044	C02B	D043	4047	8400	903B	1047	3042	2044	C02B	1043	3040	2043	902C	0005
040	0001	005F	0020	0000	0000	0000	0000	0000	0090	0098	0098	0090	004D	0059	006F	0075
050	0072	0020	0077	006F	0072	0064	0020	0069	0073	0020	0074	006F	006F	0020	0073	0068
060	006F	0072	0074	0000	007A	0059	006F	0075	0072	0020	0077	006F	0072	0064	0020	0069
070	0073	0020	0074	006F	006F	0020	006C	006F	006E	0067	0000	0000	0000	D07B	8800	C07C
080	6000	107B	3040	207B	907D	008E	108F	2048	1097	2049	0002	C085	0001	0085	C08C	0090
090	0047	0052	0045	0041	0054	0000	0098	0098	0097	004F	0052	0044	0053	0000	0000	0000

AC: 0000, IR: 7000, MAR: 001, MBR: 7000, PC: 002, IN: 0000, OUT: 0067

Buttons: Assemble, Step, Microstep, Step Back, Halted, Restart. Delay: 1 ms

If guessed word longer than selected work then output “Your word is too long”

3. Guessed_Word = Kitten ; Selected_Word = Cat

The screenshot shows the Marie.js emulator interface. The assembly code on the left includes a test case where the guessed word is 'cat' and the selected word is 'kitten'. The output window on the right displays 'Your word is too long'. The memory dump at the bottom shows the state of the machine's memory.

```

160      JnS TestCheckGuess1 / Run another test case
161      JumpI TestAll / Return
162
163      GreatAdr, Adr Great
164      Great, HEX 068
165      HEX 069
166      HEX 074
167      HEX 074
168      HEX 065
169      HEX 06E
170      HEX 000 /null - kitten
171
172      Selectedword, Adr words
173      wordsAdr, Adr words
174      words, HEX 063
175      HEX 061
176      HEX 074
177      HEX 000 /null - cat
  
```

Machine halted normally.

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	0086	7005	0000	D09B	8800	9007	9012	8400	900A	900E	D09A	8400	901A	9022	1089	303F
010	2089	9012	D05E	8800	C002	6000	105E	303F	205E	9012	D046	8800	C002	6000	1046	303F
020	2046	901A	D099	2045	8800	D099	2097	4045	8400	C02B	6000	0000	D042	8800	9030	9033
030	1040	2043	C02B	D042	4097	8400	903B	1097	3041	2043	C02B	1042	303F	2042	902C	0001
040	005F	0020	0000	0000	0000	0000	0047	0059	006F	0075	0072	0020	0077	006F	0072	0064
050	0020	0069	0073	0020	0074	006F	006F	0020	0073	0068	006F	0072	0074	0000	0074	0059
060	006F	0075	0072	0020	0077	006F	0072	0064	0020	0069	0073	0020	0074	006F	006F	0020
070	006C	006F	006E	0067	0000	0000	0000	D075	8800	C076	6000	1075	303F	2075	9077	0088
080	1089	2098	1092	2099	0002	C07F	0001	007F	C086	008A	006B	0069	0074	0074	0065	006E
090	0000	0093	0093	0063	0061	0074	0000	0000	008A	0093	0093	008A	0000	0000	0000	0000
0A0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
0B0	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

If guessed word longer than selected word then output “Your word is too long”

4. Guessed_Word = Cat ; Selected_Word = Kitten

The screenshot shows the Marie.js emulator interface. The assembly code on the left includes a test case where the guessed word is 'cat' and the selected word is 'kitten'. The output window on the right displays 'Your word is too short'. The memory dump at the bottom shows the state of the machine's memory.

```

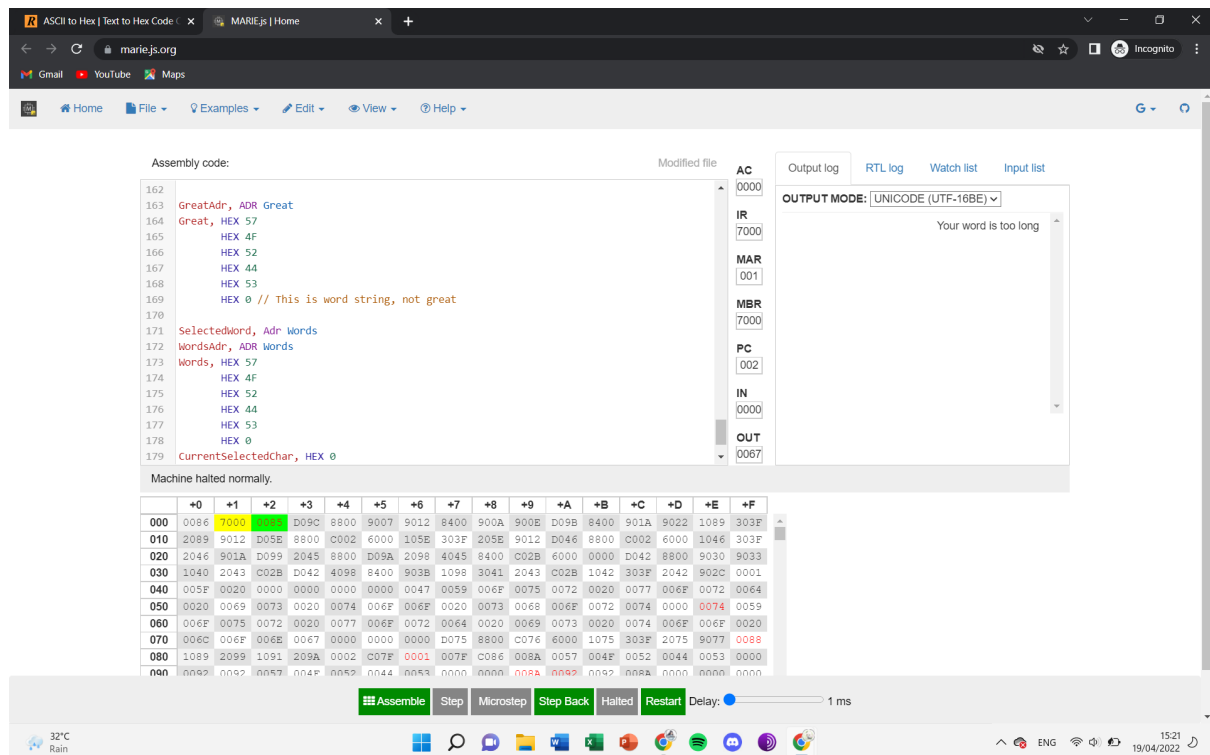
26      JumpI CheckGuess
27      Output
28      Load ErrorLongAdr
29      Add One
30      Store ErrorLongAdr
31      Jump TooLong
32
33      TooShort, LoadI ErrorShortAdr
34      skipcond 800 /check if AC > 0
35      JumpI CheckGuess
36      Output
37      Load ErrorShortAdr
38      Add One
39      Store ErrorShortAdr
40      Jump TooShort
41
42      // CHECKGUESS SUBROUTINE
43      CheckStringPosition, LoadI CheckGuessedword
  
```

Machine halted normally.

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	0086	7005	0000	D09C	8400	9007	9012	8400	900A	900E	D09B	8400	901A	9022	1089	303F
010	2089	9012	D05E	8800	C002	6000	105E	303F	205E	9012	D046	8800	C002	6000	1046	303F
020	2046	901A	D099	2045	8800	D09A	2098	4045	8400	C02B	6000	0000	D042	8800	9030	9033
030	1040	2043	C02B	D042	4098	8400	903B	1098	3041	2043	C02B	1042	303F	2042	902C	0001
040	005F	0020	0000	0000	0000	0000	005D	0059	006F	0075	0072	0020	0077	006F	0072	0064
050	0020	0069	0073	0020	0074	006F	006F	0020	0073	0068	006F	0072	0074	0000	005F	0059
060	006F	0075	0072	0020	0077	006F	0072	0064	0020	0069	0073	0020	0074	006F	006F	0020
070	006C	006F	006E	0067	0000	0000	0000	D075	8800	C076	6000	1075	303F	2075	9077	0088
080	1089	2098	1091	209A	0002	C07F	0001	007F	C086	008A	0047	0052	0045	0041	0054	0000
090	0000	0092	0092	0097	004F	0052	0044	0053	0000	008A	0092	0092	008A	0000	0000	0000

If guessed word shorter than selected word then output “Your word is too short”

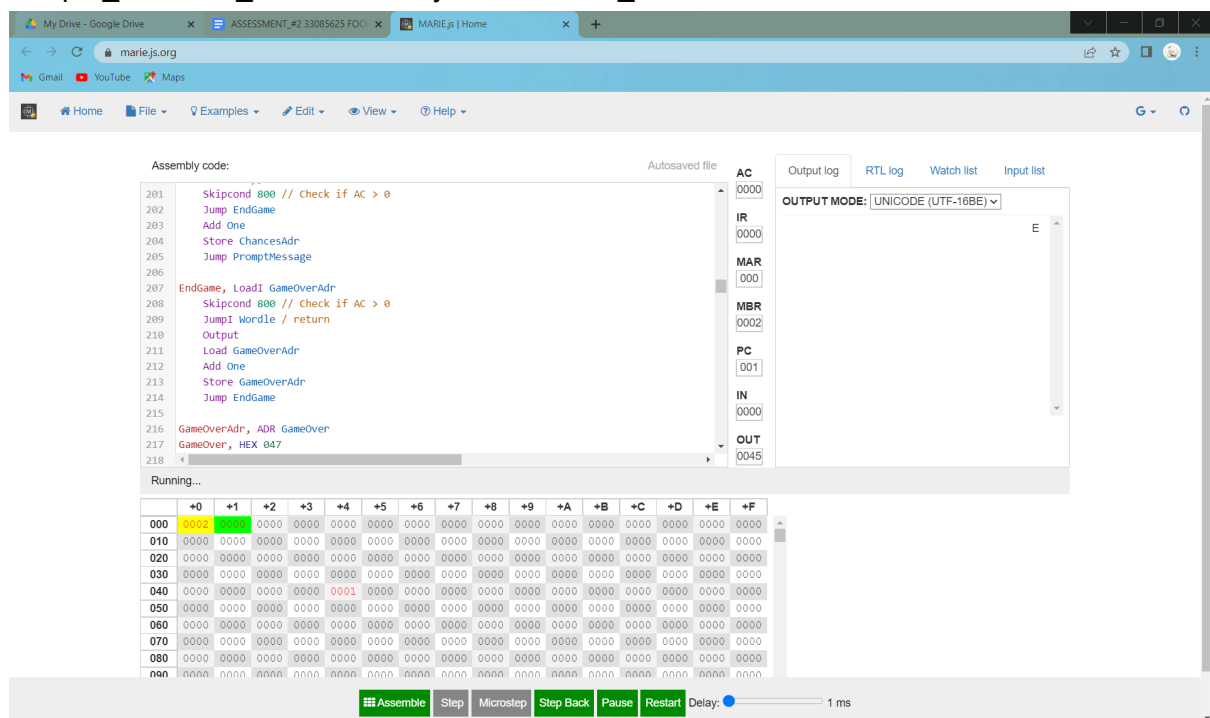
5. Guessed_Word = Words ; Selected_Word = Words



There is something wrong with my 2.4 code where it can only print too long error messages or too short error messages even though both selected-word and guessed-word are the same.

Test Case for Task 2.5:

1. Input_Guessed_Word = Samoyed ; Selected_Word = Words



The guessed word is longer than the selected word so, it is supposed to output "Your word is too long" but I did something wrong in 2.4 which only gives me the output of either "Your

word is too long” or “Your word is too short”. But this time in 2.5, no matter what I input, all my output is “E”.

2. Input_Guessed_Word = Pup ; Selected_Word = Words

Assembly code:

```

201 Skipcond 800 // Check if AC > 0
202 Jump EndGame
203 Add One
204 Store ChancesAdr
205 Jump PromptMessage
206
207 EndGame, LoadI GameOverAdr
208 Skipcond 800 // check if AC > 0
209 JumpI Wordle / return
210 Output
211 Load GameOverAdr
212 Add One
213 Store GameOverAdr
214 Jump EndGame
215
216 GameOverAdr, ADR GameOver
217 GameOver, HEX 047
218

```

Running...

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	0002	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
020	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
030	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
040	0000	0000	0000	0000	0001	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
050	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
060	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
070	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
080	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan

AC 0000
IR 0000
MAR 000
MBR 0002
PC 001
IN 0000
OUT 0045

Output log: OUTPUT MODE: UNICODE (UTF-16BE) E

Buttons: Assemble, Step, Microstep, Step Back, Pause, Restart, Delay: 1 ms

The guessed word is shorter than the selected word so, it is supposed to output “Your word is too short” but I did something wrong in 2.4 which only gives me the output of either “Your word is too long” or “Your word is too short”. But this time in 2.5, no matter what I input, all my output is “E”.

3. Input_Guessed_Word = Doggies ; Selected_Word = Words

Assembly code:

```

201 Skipcond 800 // Check if AC > 0
202 Jump EndGame
203 Add One
204 Store ChancesAdr
205 Jump PromptMessage
206
207 EndGame, LoadI GameOverAdr
208 Skipcond 800 // check if AC > 0
209 JumpI Wordle / return
210 Output
211 Load GameOverAdr
212 Add One
213 Store GameOverAdr
214 Jump EndGame
215
216 GameOverAdr, ADR GameOver
217 GameOver, HEX 047
218

```

Running...

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	0002	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
020	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
030	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
040	0000	0000	0000	0000	0001	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
050	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
060	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
070	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
080	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan	nan

AC 0000
IR 0000
MAR 000
MBR 0002
PC 001
IN 0000
OUT 0045

Output log: OUTPUT MODE: UNICODE (UTF-16BE) E

Buttons: Assemble, Step, Microstep, Step Back, Pause, Restart, Delay: 1 ms

The guessed word is longer than the selected word so, it is supposed to output “Your word is too long” but I did something wrong in 2.4 which only gives me the output of either “Your word is too long” or “Your word is too short”. But this time in 2.5, no matter what I input, all my output is “E”.

4. Input_Guessed_Word = Golden ; Selected_Word = Words

The screenshot shows the MarieJS IDE interface. The assembly code is as follows:

```

201 Skipcond 000 // Check if AC > 0
202 Jump EndGame
203 Add One
204 Store ChancesAdr
205 Jump PromptMessage
206
207 EndGame, LoadI GameOverAdr
208 Skipcond 000 // Check if AC > 0
209 JumpI Wordle / return
210 Output
211 Load GameOverAdr
212 Add One
213 Store GameOverAdr
214 Jump EndGame
215
216 GameOverAdr, ADR GameOver
217 GameOver, HEX 047
218

```

The right-hand side of the IDE shows the state of various registers and memory locations:

- AC: 0000
- IR: 0000
- MAR: 000
- MBR: 0002
- PC: 001
- IN: 0000
- OUT: 0045

The output log shows the character 'E'. The memory dump at the bottom shows a grid of memory locations (000 to 0FF) with their corresponding values. The value at address 000 is 0002, and the value at address 047 is 0001.

The guessed word is longer than the selected word so, it is supposed to output “Your word is too long” but I did something wrong in 2.4 which only gives me the output of either “Your word is too long” or “Your word is too short”. But this time in 2.5, no matter what I input, all my output is “E”.

5. Input_Guessed_Word = Words ; Selected_Word = Words

Assembly code:

```

201 Skipcond 800 // Check if AC > 0
202 Jump EndGame
203 Add One
204 Store ChancesAdr
205 Jump PromptMessage
206
207 EndGame, LoadI GameOverAdr
208 Skipcond 800 // Check if AC > 0
209 JumpI wordle / return
210 Output
211 Load GameOverAdr
212 Add One
213 Store GameOverAdr
214 Jump EndGame
215
216 GameOverAdr, ADR GameOver
217 GameOver, HEX 047
218

```

Autosaved file

Output log RTL log Watch list Input list

OUTPUT MODE: UNICODE (UTF-16BE)

AC 0000

IR 0000

MAR 000

MBR 0002

PC 001

IN 0000

OUT 0045

Running...

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	0002	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
020	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
030	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
040	0000	0000	0000	0000	0001	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
050	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
060	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
070	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
080	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
090	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Assemble Step Microstep Step Back Pause Restart Delay: 1 ms

There is something wrong with my 2.4 code where it can only print too long error messages or too short error messages even though both selected-word and guessed-word are the same. But I did something wrong in 2.4 which only gives me the output of either “Your word is too long” or “Your word is too short”. But this time in 2.5, no matter what I input, all my output is “E”.