MCD4700 Introduction to Computer Systems, N[illegible]d Security – T1 2024

Assignment [illegible] and MARIE Programming_Instruction

Purpose	Processes and programs are what makes computers do what we want them to do. In the first part of this assignment, students will investigate the processes running on their computers. The second part is about programming in MARIE assembly language. This will allow students to demonstrate their comprehension of the fundamental way a processor works. The assignment relates to Unit Learning Outcomes 2, 3 and 4.
Your task	For part 1, you will write a short report describing the processes that are running on your computer. For part 2, you will implement a simple game in the MARIE assembly language.
Value	20% of your total marks for the unit The assignment is marked out of 100 marks.
Word Limit	See individual instructions
Due Date	**11:55 pm Friday 12 April 2024 (Week7)**
Submission	• Via Moodle Assignment Submission. • Turnitin will be used for similarity checking of all submissions. • This is an individual assignment (group work is not permitted). • Handwritten work is not accepted. docx for the written tasks. • MARIE files for the second part • DRAFT submission is not assessed. • You will need to **explain your code** in an interview.
Assessment Criteria	Part 1 is assessed based on correctness and completeness of the descriptions. Part 2 is assessed based on correctness of the code, documentation/comments, and test cases. See instructions for details.
Late Penalties	By submitting a Special Consideration Form or visit this link: https://lms.monashcollege.edu.au/course/view.php?id=1331 • Without special consideration, 10% deduction per calendar day or part thereof for up to one week

	• Assessment items will not be accepted after more than 7 calendar days unless a Special Consideration application has been approved. This 7-day time frame does not apply to assessments due in Week
Support Resources	sessment page
Feedback	e provided on student work via: ort performance • specific student feedback ten working days post submission
INSTRUCTIONS	• This assignment has two parts. Make sure you read the instructions carefully. • You need to submit one zip file includes five files through the Moodle Assignment activity:
Plagiarism	**Plagiarism:** It is an academic requirement that the work you submit be original. If there is any evidence of copying (including from online sources without proper attribution), collaboration, pasting from websites or textbooks, **zero marks** may be awarded for the whole assignment, the unit or you may be suspended or excluded from your course. Monash Colleges policies on plagiarism, collusion, and cheating are available here or see this link: https://www.monashcollege.edu.au/__data/assets/pdf_file/0010/17101/dip-assessment-policy.pdf Further Note: When you are asked to use Internet resources to answer a question, this **does not mean copy-pasting text** from websites. Write answers in your own words such that your understanding of the answer is evident. Acknowledge any sources by citing them. The generative AI is not allowed to be used to generate any solutions for this assessment.

1. Processes [15 marks]

Calculate the turnaround time for the following processes and subsequently calculate their average turnaround time

Process	
P1	
P2	
P3	
P4	
P5	
P6	4
P7	1

a- In **FCFS** first-come first-served

b- In **SJF** shortest job first

c- In **Round Robin** with slice time=2

2. MARIE (65 marks)

In this task you will develop a MARIE application that performs some manipulation of characters, strings and numbers. We will break it down into small steps for you. Most of the tasks require you to write code and[illegible]e must contain proper ***comments*** and well ***indented***. You submit it as .mas[illegible]e rest of your assignment. The test cases should also be working, self-cont[illegible]bly files (without requiring much input from the user).

In-Class interviews[illegible]ed to demonstrate your code to your tutor after the submission deadline[illegible]trate will lead to "**zero marks"** being awarded to the entire programming p[illegible]nt.

Background - Lists of data

This section introduces the concepts you need for the rest of the assignment. A string is a sequence of characters. It's the basic data structure for storing text in a computer. There are several different ways of representing a string in memory and how to deal with strings of arbitrary length.

For this assignment, we will use the following string representation:

- A string is represented in contiguous memory locations, with each address containing one character.
- The characters are encoded using the ASCII encoding.
- End of a string is marked by the ASCII character '.' (i.e. dot or full-stop).
- A string can be of any arbitrary length, and will be terminated by a '.', and it may contain any of the following: alphabets (A-Z, a-z), numbers (0-9), ASCII Space Character (Hex 020) and New Line (Hex 00A).

Here is an example. A string "Dong Satria." will be represented in memory (written as hexadecimal numbers):

044	06F	06E	067	020	053	061	074	072	069	061	02E
D	o	n	g		S	a	t	r	i	a	.

Note that, in the above example, for a string with 10 characters, we need (***10+2***) words of MARIE memory in order to store all the characters belonging to that string (including a space and a '.' characters).

In MARIE assembly language programming, we can make use of the ADR command, the HEX keyword and a label "myString" to put this string into memory:

```
myStringAddr,   ADR   myString
myString,       HEX 044     /'D'
                HEX 06F     /'o'
                HEX 06E     /'n'
                HEX 067     /'g'
                HEX 020     /Space
                HEX 053     /'S'
                HEX 061     /'a'
                HEX 074     /'t'
                HEX 072     /'r'
                HEX 069     /'i'
                HEX 061     /'a'
                HEX 00A     /NL(New Line)
                HEX 032     /'2'
                HEX 031     /'1'
                HEX 038     /'8'
                HEX 033     /'3'
                HEX 039     /'9'
                HEX 039     /'9'
                HEX 030     /'0'
                HEX 030     /'0'
                HEX 02E     /'.'
```

2.1. Your name as a MARIE string (5 marks)

The following example of a MARIE string "myString" encodes a name and an ID using ASCII characters. The "name" is separated from the ID by an ASCII character "Hex 00A" (New Line). Different parts of a name are separated by another ASCII character "Hex 020" (Space). And the entire string, consisting of a name and an ID, is terminated by a dot '.' character.

Please see the example below. The label "myStringAddr" holds the address of the first character of the string. You need to follow this MARIE string while solving the task given below.


```
myStringAddr,   ADR   myString
myString,       HEX 044     /'D'
                HEX 06F     /'o'
                HEX 06E     /'n'
                HEX 067     /'g'
                HEX 020     /Space
                HEX 053     /'S'
                HEX 061     /'a'
                HEX 074     /'t'
                HEX 072     /'r'
                HEX 069     /'i'
                HEX 061     /'a'
                HEX 00A     /NL(New Line)
                HEX 032     /'2'
                HEX 031     /'1'
                HEX 038     /'8'
                HEX 033     /'3'
                HEX 039     /'9'
                HEX 039     /'9'
                HEX 030     /'0'
                HEX 030     /'0'
                HEX 02E     /'.'
```

Prepare a MARIE program to encode a string that includes your full name (first name and last name) and your student ID using ASCII characters. Following the above example, you need to use two labels, one label (e.g. “myString”) to store the first character of the string, and another label (e.g. “myStringAddr”) to store the address of the first character of the same string.

You need to submit a program that contains codes, using the ADR command and HEX keywords (like the above example) that after assembling, your name, ID and the address (of the first character) will be stored in MARIE memory. The codes must be accompanied by appropriate comments (as a paragraph before any block of code or subroutine or as inline comments wherever appropriate).

2.2. Printing string (10 marks)

Prepare a MARIE program that can print the ASCII ‘.’ terminated string of your name and your student ID that you have implemented in task 2.1. You may use the “Output” instruction to print characters in the MARIE output space. The program should be able to print any string that terminated with ‘.’.

Hint: In your program, you need to use a label “myString” that holds the start address of the string (like, myStringAddr) that you want to print. Then, you should load a character from the address “myString”, print the character, then increment the address by one, and keep doing that up to the character loaded from the address is a ‘.’ (which signals the end of the string). The output may look similar to the output below. The codes must be accompanied by appropriate comments (as a paragraph before any block of code or subroutine or as inline comments wherever appropriate).

Inside the Memory

Dong Satria
21839900

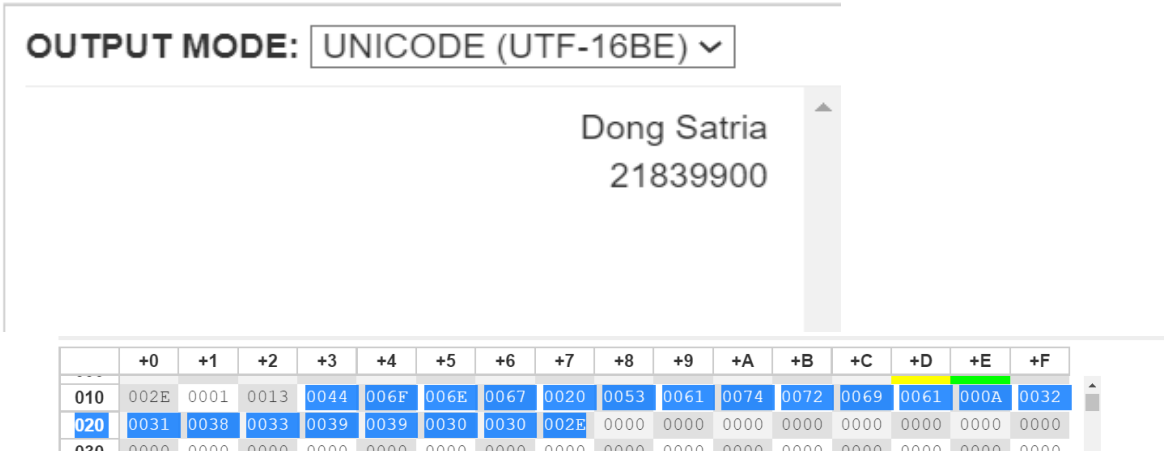

Figure 1: Print your name and ID

2.3 Subroutines to print a string and calculate its Numerology (Expression Number) (25 marks)

Numerology is a belief system that suggests the numerical value of a name can influence various aspects of personal development. The number associated with a name is often referred to as the 'expression number'. To calculate this number, each letter in the name is assigned a value from 1 to 26, following this key:

A=1, B=2, C=3, D=4, E=5, F=6, G=7, H=8, I=9, J=10,

K=11, L=12, M=13, N=14, O=15, P=16, Q=17, R=18,

S=19, T=20, U=21, V=22, W=23, X=24, Y=25, Z=26.

This method can be implemented with any name, assuming that all the letters are capital letters, and there are no special characters except spaces.

To find your Numerology or Expression number, follow these steps:

1. Write out the string (usually a name) for which you want to determine the Numerology or Expression number.
2. Match each letter in the string to its corresponding number using the key.
3. Add together all of the numbers associated with the letters in the string.
4. Reduce the sum of the string numbers:
   a. If the sum is two digits, repeatedly add together the two digits until you get a single digit.
   b. If the sum is greater than 2 digits, reduce the number to two digits by adding together the digits repeatedly until you get either one digit or two digits

Example1:

M	O	N	A	S	H
13	15	14	1	19	8

13 + 15 + 14 + 1 + 19 + 8 = 70

70 = 7 + 0 = 7

The Expression number = 7

Example2:

M	O	N	A	S	H	space	C	O	L	L	E	G	E
13	15	14	1	19	8		3	15	12	12	5	7	5

13 + 15 + 14 + 1 + 19 + 8 + 3 + 15 + 12 + 12 + 5 + 7 + 5 = 129

129 = 1 + 2 + 9 = 12

The Expression number = 12

Create two MARIE subroutines: one named "`subPrintString`" to print a string terminated with a period ('.'), and another named "`subCountNumerology`" to calculate the Expression Number for the given string. These subroutines should follow these guidelines:

The subroutines sho

1. Print the strin
2. On the secon of the (Expression Number) before reducing it.
3. On the third li sion Number after the reduction.

Figure 2.a: Using subroutines to Display a string and find an Expression number

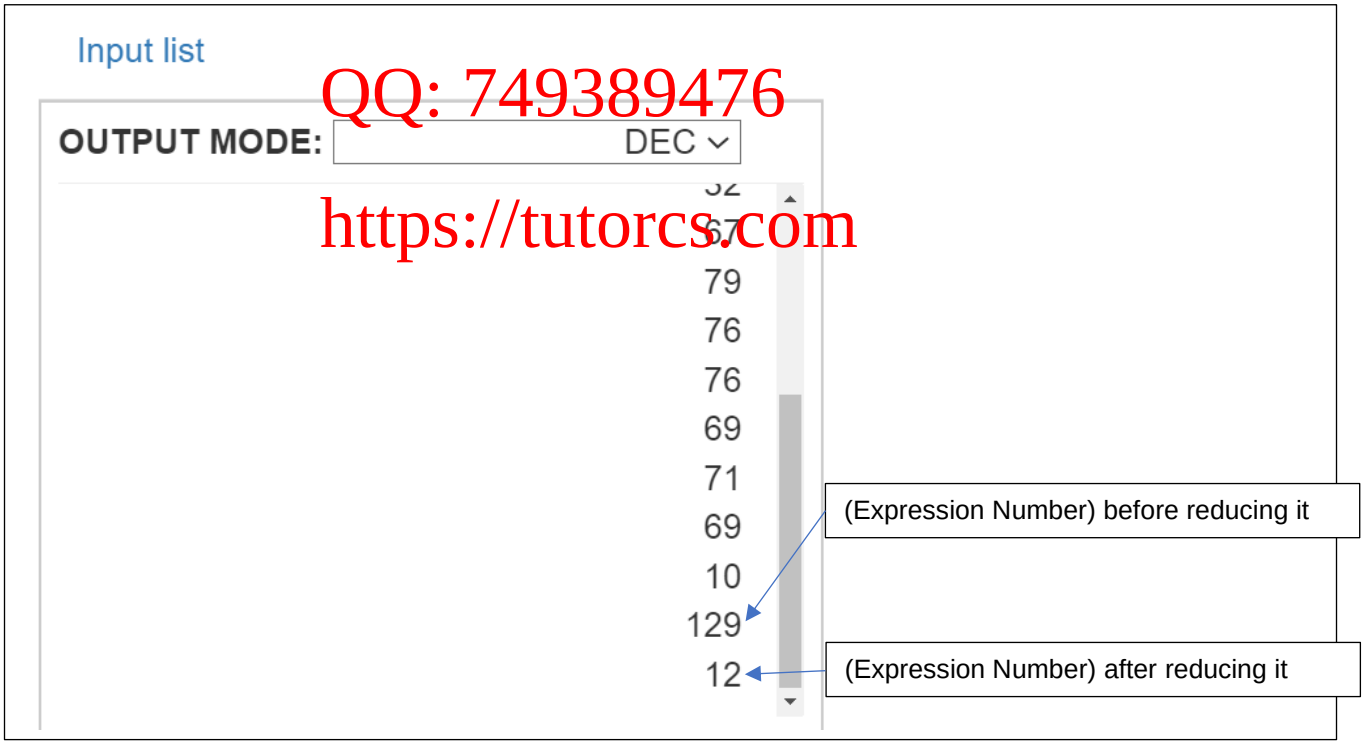

Figure 2.b: Using subroutines to Display a string and find an Expression number

Note: MONASH COLLEGE is just an example

2.4 Assembly Language and Machine Language (15 marks)

This task is for HD students

To get HD in this ass think again before ignoring this task

a. Write at least veen (at most 200 words):
   i. Assembly
   ii. Machine l
b. Prepare a MA itialised with three values (X, Y and Z) then find and print the result of (X+Y-Z) in:
   i. Assembly Language
   ii. Machine Language

Note: Assuming that the result is always between -9 and 9

The codes must be accompanied by appropriate comments (as a paragraph before any block of code or subroutine or as inline comments wherever appropriate). The code of the assemble and the machine should be in one MARIE file called (AssemblyANDMachine).

Example1:

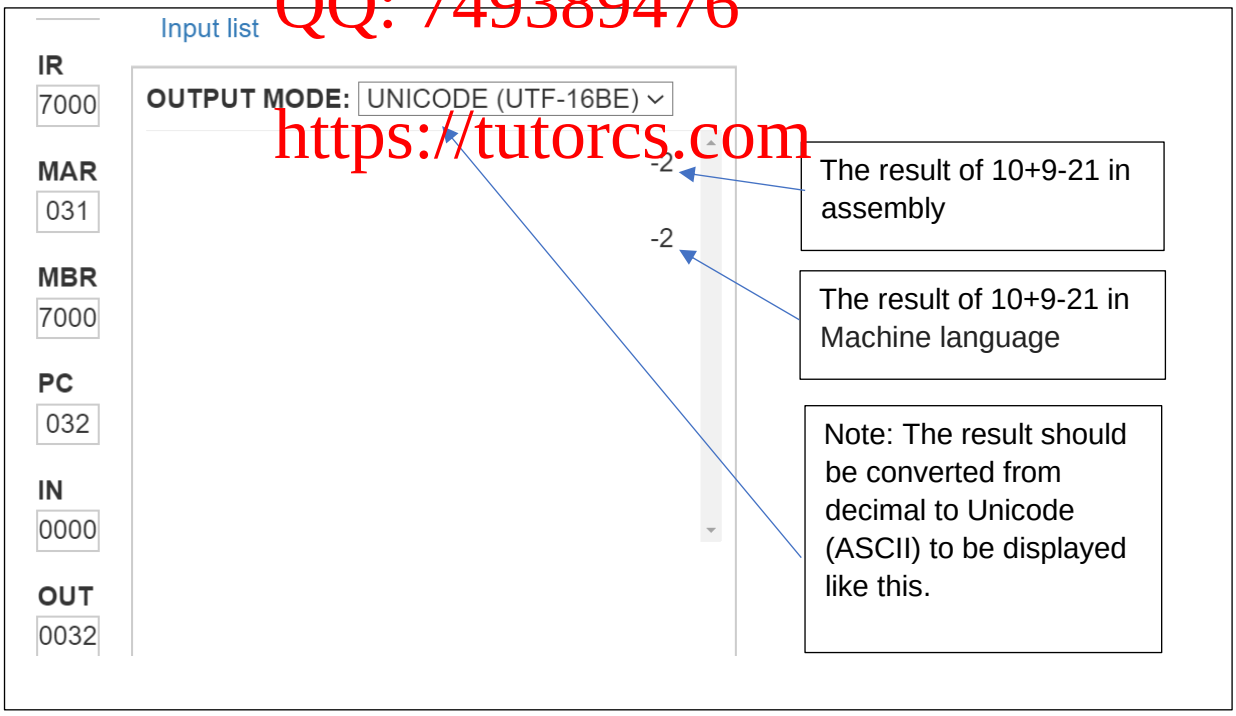

Figure 3.a: Assembly and Machine Language

Example2:

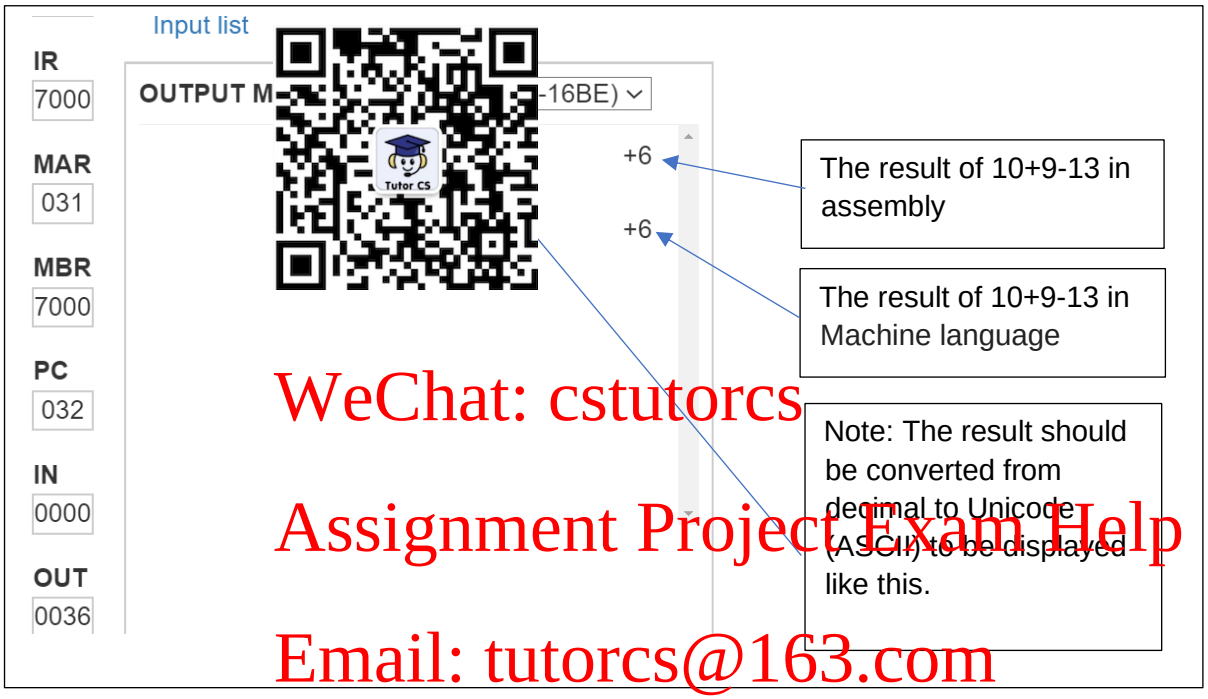

Figure 3.b: Assembly and Machine Language

Marks (for each representation):

- Correct explaining: 6 marks
- Correct code in assembly and machine: 6 marks
- Correctly convert the decimal to Unicode and display the result as above: 3 marks

Code Documentation and Development (5 marks)

All the variables/labels should have a meaningful naming convention. The code should include proper comments.

Code Readability (5 marks)

Before you submit, be sure your code is well organised and very easy to follow included code indentation, effective use of whitespace etc.

Report Structure and Correct files (5 marks)

Files to be submitted:

One folder named "YourFirstNameStudentID" containing the following files:

1. Report for the [illegible] Word file called YourFirstNameLastName StudentID.do[illegible] should include your Full name, your student ID, your class number[illegible]me.
2. MARIE files f[illegible]ame them as below:
   - 2.1_N[illegible]
   - 2.2_PrintNameID.mas
   - 2.3_SubroutinesToPrint&countNumerology.mas
   - 2.4_ AssemblyANDMachine.mas

Zip the folder under the same name and submit it to Moodle. You need to make sure there are no spaces in any of the filenames.

3. In-class oral/coding assessment (15 marks)

In addition, you will be asked some questions related to MARIE to assess your level of understanding. Your tutor will ask a couple of questions about the MARIE programming language and/or you are required to code a task using MARIE.

NOTE! *Your submitted files must be correctly identified (as described above). Any submission that does not comply will receive an automatic* ***10 marks penalty*** *(applied after marking).*