MCD4700 Introduction to Computer Systems, Networks and Security – T1 2023

Assignment 2 – Processes 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 Thursday 6 April 2023 (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 or pdf file are not accepted. Should be docx file 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 and you will be asked to code some tasks in MARIE.
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, 5% deduction per calendar day or part thereof for up to one week ● Assessment items will not be accepted after more than 14 calendar days unless a Special Consideration application has been approved. This 14-day time frame does not apply to assessments due in Week 12.
Support Resources	See Moodle Assessment page
Feedback	Feedback will be provided on student work via: ● general cohort 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.

1. Processes (30 marks)

For this task, write a brief report about processes that you observe running on your computer. You can use one of the following tools (depending on your operating system):

On Windows, use the Task Manager

On macOS, use the Activity Monitor

On Linux, use a command line tool like htop, top, or the ps command

1.1. Briefly describe the columns displayed by the tool you use that relate to: (8 marks)

a. Memory usage and
b. CPU usage of a process.

What can you say about the overall memory usage of all processes, compared to the RAM installed in your computer?

Include screenshots for the performance of the memory and the CPU.

1.2. Pick a process (7 marks)

Pick a process you perhaps don't know much about, or which you did not expect to find running on your computer. Try to find out and describe briefly what it does.

Include a screenshot of your processes in the report along with utilization graphs. The word limit for this part (all three questions together) is 600 words (about 1 page, not including images and tables).

1.3. Calculate the average turnaround time (ATT) (15 marks)

Calculate the average turnaround time (ATT) for the following processes, show all the works

Process	Processing Time
P1	3
P2	5
P3	8
P4	2
P5	1
P6	4
P7	6

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

b- In **SJF** shortest job first

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs

c- In **Round Robin** with slice time=3

2. MARIE (55 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 test cases. The code must contain proper **comments** and well **indented**. You submit it as .mas files together with the rest of your assignment. The test cases should also be working, self-contained MARIE assembly files (without requiring much input from the user).

In-Class interviews: You will be required to demonstrate your code to your tutor after the submission deadline and you will be asked to code some tasks in MARIE. Failure to demonstrate will lead to "**zero marks"** being awarded to the entire programming part of this assignment.

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 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 MARIE file that contains codes, using the ADR command and HEX keywords (like the above example), so that after assembling, your name, ID and the address (of the first character of the string) is 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 your name and ID (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.

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

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
010	002E	0001	0013	0044	006F	006E	0067	0020	0053	0061	0074	0072	0069	0061	000A	0032
020	0031	0038	0033	0039	0039	0030	0030	002E	0000	0000	0000	0000	0000	0000	0000	0000
030	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Figure 1: Print your name and ID

2.3. A subroutine for entering name from keyboard and print it (15 marks)

In this section, we will focus on getting our names as input from keyboard, and storing them in a particular location in MARIE memory. For this, prepare a MARIE subroutine called **subInputNames** to input a name from keyboard as ASCII '.' terminated strings, two of them, consecutively, to be treated as First and Last names. The name is to be stored beginning at memory address 200$_H$ one name (First Name and Last Name) in one row of MARIE memory. At the end of the "Last Name" entry, aa ASCII '.' will be stored to mark the end of the current name in the memory. To separate between the first and last name the ASCII "0" should be sued. After entering the entire string print it.

To receive full marks, your code needs to be in the form of a subroutine that can be called using the JnS instruction. You need to write a MARIE main program to call this subroutine.

Figure 2. Input and Print any String

2.4. A subroutine to find suburb name base on postcode (25 marks)

Prepare a MARIE **subroutine** called *suburbName* which takes one out of five numbers which are represent the post code of specific suburbs in Melbourne. Then display the name of the suburb depending on the post code that you have entered.

You need to display the following message related to the input:

Input (Decimal)	Output (Unicode)
3000	**Melbourne (3000)**
3008	**Dockland (3008)**
3162	**Caulfield (3162)**
3168	**Clayton (3168)**
3922	**Phillip Island (3922)**
Anything else	**Wrong Post Code**

Write a main program where the user will be:

    a. Asked to enter (input) a post code.
    b. Find and display (output) the appropriate message.

The codes must be accompanied by appropriate comments (as a paragraph before any block of code or subroutine or as inline comments wherever appropriate).

To receive full marks, your code needs to be in the form of a subroutine that can be called using the *JnS* instruction. You need to write a MARIE main program to call this subroutine.
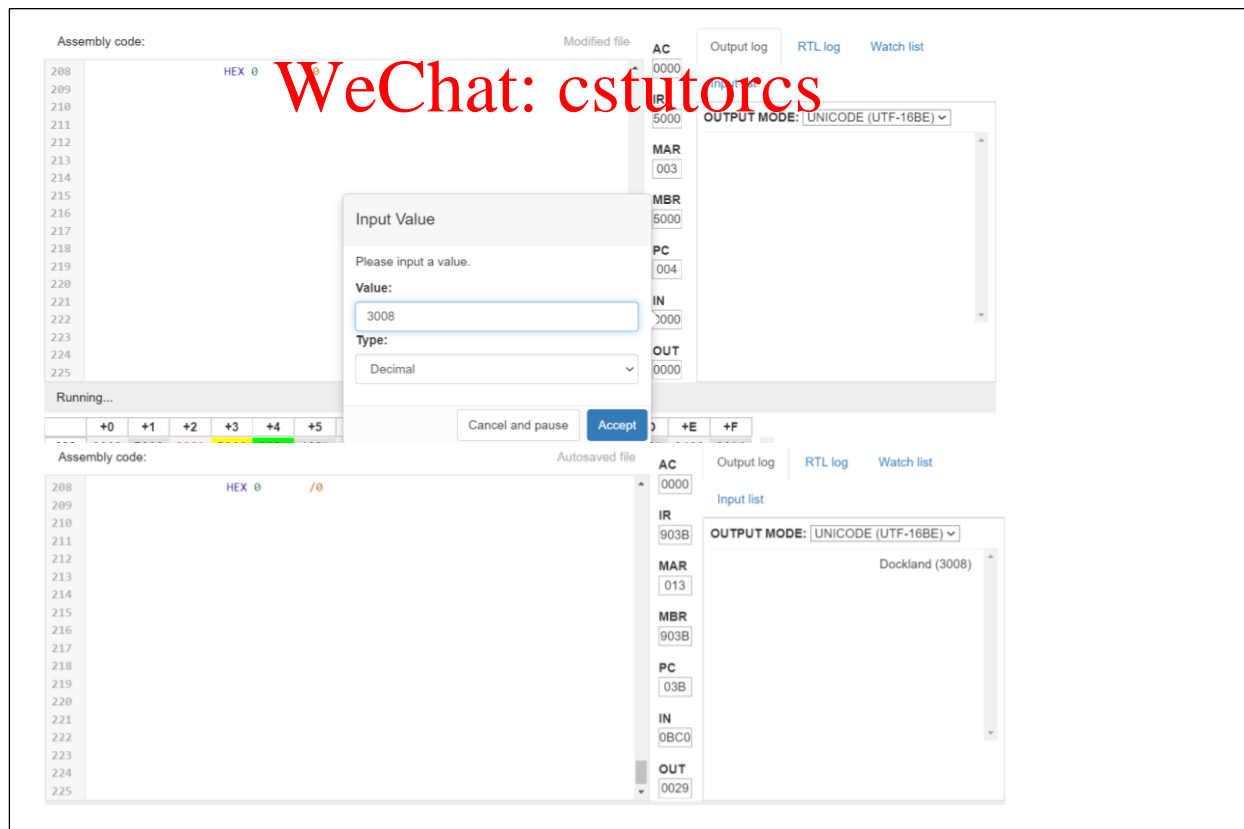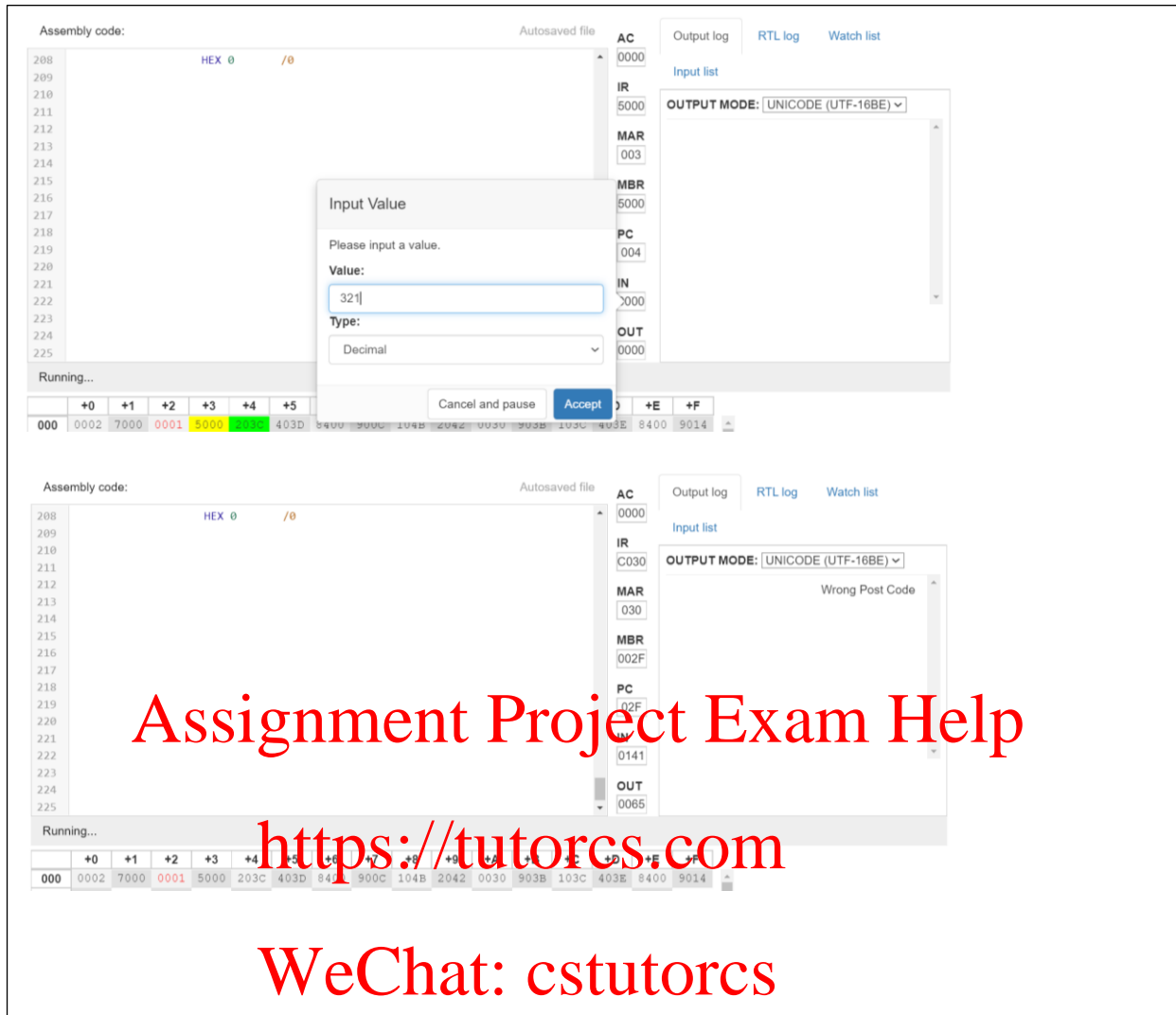

Figure 3: Reading a Post Code and Display a Message

Figure 4: Reading Wrong Post Code

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 "YourFirstNameLastName_StudentID" containing the following files:

1. Report for the written tasks (One Word file called YourFirstNameLastName_StudentID.doc / docx). The report should include your Full name, your student ID, your class number and your tutor's name.
2. MARIE files for tasks 2.1 to 2.4 name them as below:
    -   2.1_NameID.mas
    -   2.2_PrintNameID.mas
    -   2.3_InputandPrintName.mas
    -   2.4_PrintPostCode.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.

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).*

Assignment Project Exam Help

https://tutorcs.com

WeChat: cstutorcs