MONASH University

FACULTY OF INFORMATION TECHNOLOGY

程序代写代做 CS编程辅导

FIT1047 Introduction to computer systems, networks and security - S1 2024

A... esses and MARIE Programming

WeChat: cstutorcs

Purpose	Proc... re 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	Part 1: Write a short report describing the processes running on your computer. Part 2: Disassemble and add comments to a MARIE program. Part 3: Submit your reflections. Part 4: Write a MARIE program that can display bitmap numbers. Part 5: In-class in-person interview (week 8 applied session)
Value	25% of your total marks for the unit The assignment is marked out of 60 marks.
Word Limit	See individual instructions
Due Date	**Part 1-4: 9:30 am Monday 15 April 2024** **Part 5: Interview conducted during Week 8 your official allocated Applied Session**
Submission	Overall, 3 files are required via Moodle Assignment Submission: • Part 1: one pdf file (containing answers to the questions) • Part 3: one pdf file (containing reflection from Week 5 and 6) • Part 2 and 4: one .zip file, containing one .mas file for Part 2 and one .mas file for Part 4 Turnitin and MOSS will be used for similarity checking of all submissions. This is an individual assignment (group work is not permitted). In this assessment, you must not use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task. You will need to **explain and extend your code** in an interview. (Part 5)
Assessment Criteria	Part 1 is assessed based on correctness and completeness of the descriptions. Part 2 is assessed based on correctness of the code and the labels/comments. Part 3 is assessed based on relevance of the submission to the unit. Part 4 is assessed based on correctness of the code, as well as the documentation/comments.

Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

程序代写代做 CS编程辅导

	Part 5 is assessed based on the understanding of the code you have written. See instructions for details.
Late Penalties	calendar day or part thereof for up to one week than 7 calendar days after the due date will receive a d no assessment feedback will be provided.
Support Resources	nt page
Feedback	Feedback will be provided on student work via: general cohort performance specific student feedback ten working days post submission

WeChat: cstutorcs

INSTRUCTIONS

This assignment has five parts. Make sure you read the instructions carefully.

Assignment Project Exam Help

Part 1 and 2 are required to achieve a Pass or higher mark for the assignment.

Part 3 is a reflection activity. You do not receive marks for this task, but it is a hurdle requirement (i.e., you will not get a mark for this assignment if you don't submit it).

Email: tutorcs@163.com

Part 4 and 5 are MARIE programming tasks, which you need to complete in order to get an overall mark of 60 or higher in this assignment.

Failure to attend the interview (Part 5) will result in 0 points for the entire Part 4 and 5, regardless of your submission in Moodle.

QQ: 749389476

How are marks and grades determined?

https://tutorcs.com

Grade level	Requirements	exact mark
Pass	• submission includes responses addressing reflective questions • achieves between 60% and 79% in part 1 and 2	between 50 and 59 depending on your score in **parts 1 and 2**
Credit	• submission includes responses addressing reflective questions • achieves between 80% and 100% in parts 1 and 2 • achieves between 10% and 49% in parts 4 and 5	between 60 and 69 depending on exact scores in all parts
Distinction	• meets requirements for Credit • achieves between 50% and 79% in part 4 and 5	between 70 and 79 depending on exact score in **part 4 and 5**
High Distinction	• meets requirements for Credit • achieves between 80% and 100% in part 4 and 5	between 80 and 100 depending on exact score in **part 4 and 5**

程序代写代做 CS编程辅导

Part 1: Processes (10 marks)

For this task, write a brief report about processes that you observe running on your computer. You can u ng tools (depending on your operating system):

- On Windows ger
- On macOS, tor
- On Linux, us ol like htop, top, or the ps command

Answer the following

1. Briefly describe the columns displayed by the tool you use that relate to 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 graphs or charts for the comparison. (5 marks)
2. 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. (5 marks)

WeChat: cstutores

Assignment Project Exam Help

Include a screenshot of your processes in the report along with CPU/memory usage graphs and/or charts. The screenshot should show between 5 and 10 processes.
The word limit for this part (both questions together) is 500 words (about 1 page, not including images and tables).

Email: tutorcs@163.com

QQ: 749389476

Submit your report for this part (Part 1) as a PDF file (independent of the other parts) in Moodle.

https://tutorcs.com

程序代写代做 CS编程辅导

Part 2: MARIE D... ...marks)

Follow the link on M... ...personalised MARIE memory screenshot for this task.

Important: Your m... ...s different from the one other students are working on. Only c... ...hile you are correctly logged into Moodle with your own student a...

Task 2.1: Disassemble the memory (10 marks)

Based on the memory contents, recreate the MARIE program that corresponds to your personalised memory screenshot. This is called "disassembling" the machine code, since it is the opposite operation of "assembling" the MARIE code into the binary memory contents.

For each memory cell, decode the instruction and (if applicable) the address that the memory cell is encoding. You can make the following assumptions:

- There is exactly one Halt instruction in the code
- Every memory location after the Halt instruction contains data
- Any memory location that contains the value 0 is data (even before the Halt instruction)

WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

Here is an example of a memory screenshot and the corresponding decoded MARIE program:

	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	5000	3005	6000	9000	7000	000A	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Disassembled program:

```
Input
Add 005
Output
Jump 000
Halt
DEC 10
```

Note: You need to decode the actual instructions. E.g. for the first memory location, HEX 5000 would not be a valid answer. The contents of all memory that follows the Halt instruction is considered to be data. Therefore, DEC 10 is the correct decoding of location 5 (instead of JnS 00A), and HEX 00A would also be correct. You don't need to list all the locations containing zeros starting from address 006 (these will be filled with zeros by the assembler anyway).

Tip: You can verify that your disassembled code is correct by entering it into the MARIE simulator, assembling it and comparing the memory contents to the screenshot you started from.

Task 2.2: Add labels (5 marks)

Now update the program you decoded in Task 2.1. Removing all hard-coded memory addresses by adding labels to replace all memory locations that are used as addresses in

程序代写代做 CS编程辅导

the program instructions. Labels should have meaningful names in the context of what the program does (i.e., not just A, B, C).
For the example above, this could result in the following program:

```
MainLoop, Input
Add Ten
Output
Jump MainLoop
Halt
Ten, DEC 10
```

Task 2.3: Add comments (5 marks)

WeChat: cstutorcs

Comment the code based on your understanding of what it does. Comments should describe the function of the different parts. E.g., if you identify a subroutine in the code, add a comment at the start of the subroutine that describes what it does, and whether it takes any arguments.

Assignment Project Exam Help

For this part (Part 2), you need to submit one .mas file containing your final code. Do not submit one .mas file *per each subtask*! Your .mas file must be added to a .zip archive, together with the (separate) .mas file for Part 1.

Email: tutorcs@163.com

QQ: 749389476

Part 3: Reflections (hurdle requirement, no marks)

Copy/paste your reflections for weeks 5 and 6 from the Ed Lessons into a PDF document. This part is a hurdle requirement, i.e., we won't mark the other parts if you do not submit this part. The reflections can be just a few sentences per week, but need to genuinely relate to your learnings for the week.

https://tutorcs.com

Submit your reflection for this part (Part 3) as a PDF file (independent of the other parts) in Moodle.

程序代写代做 CS编程辅导

Part 4: MARIE Programming (22 marks)

In this task you will c[illegible]plication that draws numbers on the screen. We will break it down into si[illegible]

Note: This part is fo[illegible]to achieve a Distinction or High Distinction mark in this assignment. In o[illegible]marks for this part, you must reach at least a Credit grade for Parts 1, 2

Each task requires you to write **code** and **documentation**. On Moodle, you will find a **template** for the code. **Your submission must be based on this template**, i.e., you must add implementations of your own subroutines into the template. The template already contains the main program that calls the subroutines.

Your code must contain readable comments and meaningful labels for your tutor / marker to understand the logic flow of your program (e.g. the purpose of a subroutine, `jump` / `skipcond` statement etc.).

In-class interview (Part 5): You will be required to join an interview to demonstrate your code to your tutor during your applied session in week 8 (after the submission deadline). **Failure to demonstrate will lead to zero marks being awarded for the entire Part 4, regardless of your submission in Moodle.** In addition, during the interview (Part 5), you will also need to answer further questions about your submitted code (see below for details).

Code similarity: We use tools such as MOSS and Turnitin to check for collaboration and copying between students. If you copy parts of your code from other students, or you let them copy parts of your code, this will result in a report to the Academic Integrity team. As a result, **you may receive a penalty such as 0 marks for the entire assignment, 0 marks for the whole unit, or in severe cases (such as contract cheating), suspension or expulsion from Monash University**.

Rubric: The marking rubric on Moodle provides details for the marking. A correctly working MARIE program that covers all tasks and is well documented will receive full marks. Missing/incomplete documentation will result in a loss of up to ¼ of the task's marks.

Introduction: Bit-mapped displays

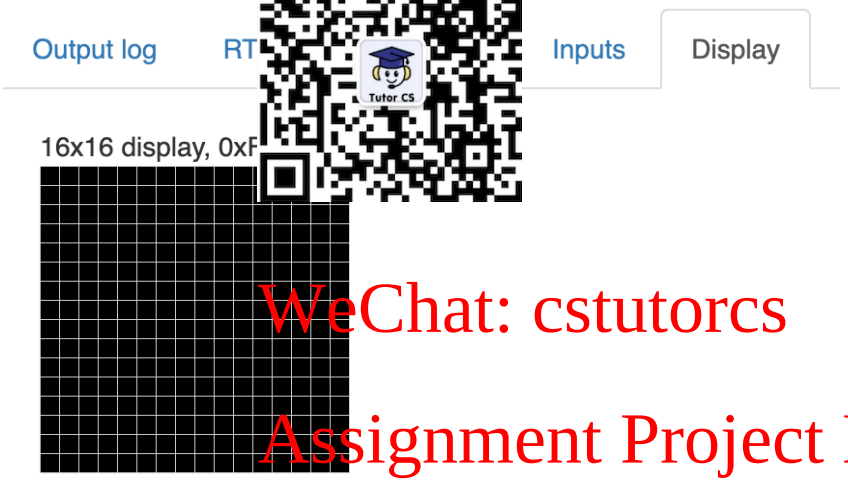
So far, the only output capability we have seen in the MARIE system is using the `Output` instruction, which will print a single 16-bit value. Many computers of course are capable of displaying arbitrary graphics, often in high resolution and great colour depth.
In the lectures on input/output systems, we have seen that one way to implement this is to ***map*** a certain location of the memory to an output device. I.e., writing to that memory location (using e.g. a `Store` instruction) causes the output to happen.
In the simplest form of graphics hardware, we can dedicate part of the RAM to be ***graphics memory***. Each memory cell corresponds to a ***pixel*** on screen, and the value in the memory cell encodes the ***colour*** of the pixel. That way, we can create arbitrary graphics by simply

WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

程序代写代做 CS编程辅导

writing values into the memory.

The MARIE simulator has a feature called ***Display***, which you access from the list of tabs that also shows the etc:

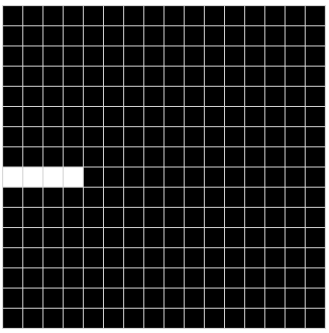


WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

The display shows the memory from address F00 to address FFF as a 16x16 pixel screen. The value in the memory locations represents the colour of the pixels. We will only use the colours black, represented as 0, and white, represented as FFFF. When you start the MARIE simulator and assemble your code, the memory starting from location F00 is (usually) filled with zeroes, which means that the display is black. Let's now change the contents of the memory using some Store instructions:

```
Load White
Store 0F80
Store 0F81
Store 0F82
Store 0F83
Halt
White, HEX FFFF
```

After running this program, the display will look like this:



You can see that the first four pixels in the 9th row have now turned white.

程序代写代做 CS编程辅导

Task 4.1 Clearing the display (4 points)

Write a subroutine SubClearDisplay that turns all pixels in the graphics memory white. Remember that the [illegible]nges from address 0F00 to address 0FFF, and that white pixels are repr[illegible]e FFFF. Document your subroutine with comments.

Task 4.2 Painting a [illegible]s)

The template for this [illegible]for **bitmaps** of the digits 0-9, stored at the label Font. Each digit consists [illegible] The first 3 words are the first row of pixels, the next 3 words are the sec[illegible] For example, the digit 2 is represented as

```
0     0     FFFF
FFFF FFFF 0
FFFF 0     FFFF
0     FFFF FFFF
0     0     0
```

WeChat: cstutorcs
Assignment Project Exam Help
Email: tutorcs@163.com
QQ: 749389476
https://tutorcs.com

You can see the pattern here, the zeros "paint" the shape of the character 2 in black, with the background in white (FFFF).

Your task is to write a subroutine called SubPaintDigit that paints a digit into the graphics memory. The start of the subroutine needs to look like this:

```
PaintDigitCharacter, HEX 0
PaintDigitDisplay, HEX 0
SubPaintDigit, HEX 0
```

In the PaintDigitCharacter argument, we pass the address of the first pixel data in the font for the digit we want to paint. In the PaintDigitDisplay argument, we pass the address of the top-left corner where we want to start painting in the graphics memory. For example, to paint the digit 0, starting from the second pixel in the second row, we could use the following code:

```
Load FontAddr
Store PaintDigitCharacter
Load Display22
Store PaintDigitDisplay
JnS SubPaintDigit
Halt
Display22, HEX 0F11
```

Note that the address 0F11 (label Display22) lies exactly 17 words after the start of the graphics memory. This means we're skipping the first row (16 words) and the first pixel in the second row (1 word).
Here we simply use FontAddr to refer to the first character (for the digit 0). For the other characters, we would have to add a corresponding offset into the font memory.

程序代写代做 CS编程辅导

In order to paint a digit in your subroutine, you can follow this "recipe":

- Your subroutine should contain two nested loops.
- Each digit contains 15 pixels, so you need to loop through those 15 pixels, load each one from the font definition and store it into the graphics memory. This is the outer loop of your subroutine.
- After each scanline, you need to start in the next row of the graphics display. This means if you are currently writing into graphics memory at address *X*, you now need to continue at address *X* plus the width of the display minus the width of a character. This is the inner loop of your subroutine.
- Once you have "copied" all 15 pixels from the font definition into the graphics memory, you can exit the subroutine.

WeChat: cstutorcs

Your subroutine needs to contain sufficient comments to enable someone else (like the person marking your assignment) to understand the purpose of each line of your code.

Assignment Project Exam Help

Task 4.3 Counting down (8 points)

Your final task is to implement a subroutine `SubCountDown` that clears the screen and then counts down from 9 to 0, drawing those digits on the bit-mapped display using the subroutines developed in the previous tasks.

Email: tutorcs@163.com

In order to get full marks, your code needs to use a loop that decrements a counter and calls `SubPaintDigit` based on the value of the counter, rather than a sequence of instructions that calls `SubPaintDigit` with each digit's address. Use additional subroutines to structure your code nicely.

QQ: 749389476

https://tutorcs.com

You will notice that it would be nice for the countdown to wait for a fraction of a second between digits. Think of a way you can achieve this, so that the countdown takes (more or less) exactly 10 seconds on your computer to execute. Document how you achieved this in the code comments.

For this part (Part 4), you need to submit one .mas file, based on the template, containing the code for all subroutines. Do not submit one .mas file *per each subtask*! Your .mas file must be added to the .zip archive that also contains your (separate) .mas file for Part 2.

Part 5: In-class interview (8 points)

You need to demonstrate the code you submitted for Task 4.1–4.3 to your tutor in an in-class in-person interview (**to be conducted during your official allocated Applied session in Week 8**) after the submission deadline. Failure to explain how your code works will result in 0 points for the individual tasks that you cannot demonstrate.

In addition, you will be asked to modify the code you submitted in certain ways and explain how the MARIE concepts work that you were required to use for the individual tasks. These additional questions add up to 8 points for this task (Task 4.4).

Failure to attend the interview will result in 0 points for the entire Part 4 and 5, regardless of your submission in Moodle.