Introduction to Computer Systems 2019

Assignment 2

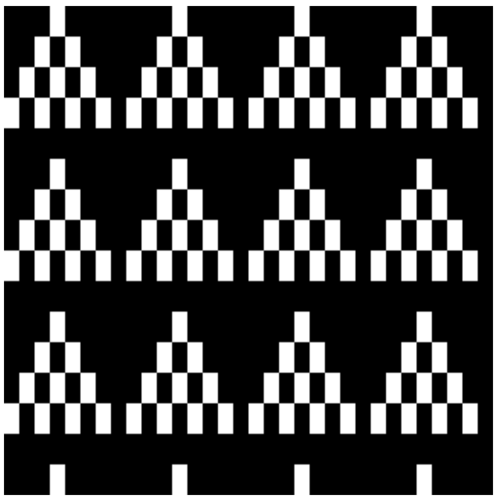
This assignment is due in by 16:00 on Thursday 9 May. It should be submitted via the Assignments section of the module Canvas site, **as a single ZIP file**. The ZIP file should contain: (1) your report, as a PDF; and (2) your program, as a plain text file.

Note that this assignment is **individual work**. Do not discuss it with your peers, or copy other people's work.

1. Choose **ONE** of the following two problems. For your chosen problem, answer **all** of the parts (a) to (d) that follow.

EITHER **Display a pattern.** The Brookshear machine emulator provided on the Teaching drive and on Canvas (Units > Resources) has a bitmapped display, which shows data in memory locations 80–FF as a black and white image, with one pixel per bit.

The image below can be put on the emulator display by writing a program that stores bit patterns in memory. **Inspect the image carefully**, and write a program to display it and then halt. The program must produce the pattern regardless of what was previously stored in memory, so it must write to every memory location from 80 to FF. The program itself may only be stored in locations 00 to 7F.



OR **Switch case and compress spaces in a character sequence.** Write a program that takes as input an arbitrary sequence of ASCII characters and switches the case of all the letters (lower case to upper case and upper case to lower case) and changes sequences of two or more space characters to a single space. For example, if the input is

```
AAAbbb123,   cccDDD
```

then the output should be

```
aaaBBB123, CCCddd
```

Note that characters other than letters and spaces must not change. Details of ASCII character codes can be found in textbooks or on the Web, and the BM emulator shows the character representation of each byte in memory.

Your program should assume that the input character sequence is stored in memory with one character per byte starting at location 80 (hex), and that the end of the sequence is indicated by a byte containing 00 immediately after the last character of the sequence. Your program should store the output starting at location C0, and should put a 00 byte immediately after the last character.

This means that the input cannot be longer than 63 (decimal) characters, since otherwise the output would overwrite the end of the input. Your program may assume that this requirement is satisfied; it does not need to check it. The program itself may only be stored in memory locations 00 to 7F.

Implement your solution to **ONE** of the problems above in a Brookshear Machine (BM) **assembly language** program. For the problem you have chosen, submit answers to **all** of parts (a) to (d) below:

(a) An algorithm to solve the problem. This must describe the steps needed to solve the problem *without* using any assembly language code or concepts. The control structure (loops, conditionals etc.) should be clearly shown. You are strongly recommended to use pseudocode with a brief supporting explanation, and to do this part before starting to program. [20 marks]

(b) A BM assembly language program that solves the problem. The program should use DATA pseudo-instructions as a way of naming important items of data. Marks will be awarded according to the criteria listed below, and also for how near your program comes to carrying out the required task, and how closely it follows your algorithm in part (a). [35 marks]

(c) A short description of what happens when your program is run — that is, what the display shows or what changes in memory as the code executes. Feel free to include screenshots of the emulator in your report to support your statements. If your program carries out the task exactly as specified, say so and give the number of instructions executed (as reported in the “Messages” pane). If your program does not carry out exactly the required task, state what it actually does (e.g. by showing the image it produces if you chose the pattern display option; you can save this with the “Save Image” button in the emulator). Mention any error messages that appear.

There are no marks for this part, but if you omit it or it turns out that your program does not do what you claim, *no marks will be awarded for the entire question 1*.

(d) A brief technical description of your program, which explains the role of each register, the ways in which the program addresses data in memory, and how any jump statements relate to your algorithm in part (a). Marks will be awarded according to the criteria listed below, but will be reduced if the program is very simple or does not complete at least part of the task. [15 marks]

2. This question is about computer programming tools. Answer both (a) and (b) below.

(a) Explain what an *interpreter* does. Some integrated development environments (IDEs) provide both a compiler and an interpreter for the same high-level language. Why?

(b) Explain what a *disassembler* does. A cybersecurity expert might use one to investigate a computer virus. Explain how, and discuss the limitations of a disassembler for this task.

You may need to do some independent research. You need not write more than 800 words in total for this question. [30 marks]

Marking Criteria

The Informatics marking criteria document at
http://www.sussex.ac.uk/ei/internal/documents/ugmarkingcriteria.pdf
states the level of achievement expected in each marks band.

Criteria specific to this assignment are listed below.

Text

- communicates concepts accurately
- written concisely, including only relevant information
- cites references for information that is not common knowledge
- shows understanding by not directly quoting from sources
- grammatical, punctuated correctly, with no spelling mistakes

Algorithms

- clearly, accurately, and completely described, with no unnecessary or duplicated steps

Programs

- readable and concise, with comments relating code to algorithm

Referencing

- referencing style is consistent, using just one of Harvard system, Numeric style, or footnotes
- sources contain full identifying information (author, year of publication, title, URL, etc.)
- sources are cited at appropriate points in the text

Overall submission

- report is word-processed, with nothing drawn by hand or handwritten
- submitted as a single ZIP file, containing a PDF and a plain text file