Computer Science (A Level)

Assembly Language booklet
B Ebrahim

Please note that you may see slight differences between this paper and the original.

Candidates answer on the Question paper.

OCR supplied materials:
Additional resources may be supplied with this paper.

Other materials required:

- Pencil
- Ruler (cm/mm)

Duration: Not set

Candidate forename		Candidate surname	

Centre number						Candidate number				

INSTRUCTIONS TO CANDIDATES

- Write your name, centre number and candidate number in the boxes above. Please write clearly and in capital letters.
- Use black ink. HB pencil may be used for graphs and diagrams only.
- Answer **all** the questions, unless your teacher tells you otherwise.
- Read each question carefully. Make sure you know what you have to do before starting your answer.
- Where space is provided below the question, please write your answer there.
- You may use additional paper, or a specific Answer sheet if one is provided, but you must clearly show your candidate number, centre number and question number(s).

INFORMATION FOR CANDIDATES

- The quality of written communication is assessed in questions marked with either a pencil or an asterisk. In History and Geography a *Quality of extended response* question is marked with an asterisk, while a pencil is used for questions in which *Spelling, punctuation and grammar and the use of specialist terminology* is assessed.
- The number of marks is given in brackets **[ ]** at the end of each question or part question.
- The total number of marks for this paper is **60**.
- The total number of marks may take into account some 'either/or' question choices.

1. An assembler may be used to produce machine code from assembly language.

Describe machine code and assembly language, making clear the differences between them, and describe the tasks done by an assembler.

The quality of written communication will be assessed in your answer to this question.

[8]

2. A digital coffee making machine has a CPU that uses the Little Man Computer Instruction Set.

Part of the coffee making machine's code asks the user to press a button to select strength. The code outputs 1 which will switch on a green light to indicate a valid selection or outputs 0 to indicate an invalid selection.

The code is shown below:

```
                INP
                STA   entry
                LDA   max
                SUB   entry
                 BRP   accept
                LDA   redLight
                BRA   printAndEnd
accept          LDA   greenLight
printAndEnd     OUT
                HLT
greenLight      DAT   1
redLight        DAT   0
max             DAT   5
entry           DAT
```

Fig. 1

(i) Tick the appropriate boxes below to indicate which inputs will result in a green light (i.e. code outputs 1) and which with a red light.

Input	Green Light	Red Light
1		
2		
3		
4		
5		
6		
7		
8		
9		

[2]

(ii) Explain which registers and buses are used, and the values they store/carry, when the line `LDA redLight` is executed (after it has been fetched and decoded). You should assume the address `redLight` refers to memory location 11.

---**[6]**

(iii) Write code in a high-level language or pseudocode that has the same functionality as the code in Fig. 1.

____ [3]

(iv) * Discuss the differences between assembly code and high-level languages. You should refer to:

- the advantages and disadvantages of writing programs in assembly code rather than a high-level language
- when each approach might be used
- why the coffee machine was programmed in assembly code.

____

____

____

____

____

____

____

____

____

____

____

____

____

____

[9]

3. A Little Man Computer (LMC) assembly language program is stored in memory as shown in Fig. 3.1.

0	LDA &7
1	ADD #4
2	OUT
3	HLT
4	6
5	2
6	10
7	15
8	16
9	17

Fig. 3.1

In this variant of LMC the symbols & and # are used to denote different modes of addressing.

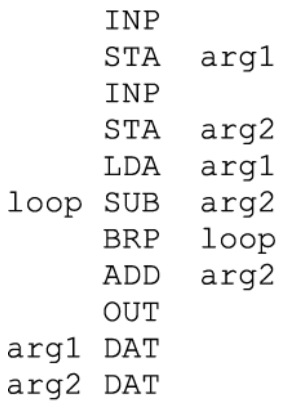
Given that the output is 17, state the addressing mode represented by each symbol.

(i) & ____ **[1]**

(ii) # ____ **[1]**

An assembler is used on the code.

4. The following assembly code in Fig. 1 is written for the Little Man Computer instruction set.



```
     INP
     STA  arg1
     INP
     STA  arg2
     LDA  arg1
loop SUB  arg2
     BRP  loop
     ADD  arg2
     OUT
arg1 DAT
arg2 DAT
```

Fig.1

(i) State what the program in Fig. 1 does.

[1]

(ii) Using pseudocode write a program for a procedural language that takes in two inputs and gives the same output as the program in Fig. 1.

[2]

5. Describe what is meant by the term ‘assembler‘.

[2]

6(a). A particular programming language uses facts and rules. For one problem in a college, some of the program statements about students and courses are listed.

student (ben)	{Ben is a student}
student (cindy)	
science (computing)	{Computing is a science}
science (mathematics)	
language (english)	{English is a language}
language (french)	

studies_science (A,B) if student (A) and science (B)
studies_language (C,D) if student (C) and language (D)

The examples you use in your answers to the questions must be based on the program statements listed.

Give the name for this type of programming language.

---[1]

(b). Give **one** example of a rule.

---[1]

(c). Explain the term goal by writing an example.

---[2]

(d). Explain the term instantiation, showing how it is used.

---[2]

(e). Explain how backtracking is used.

---[2]

7. The program, as shown in Fig.2 below, is written in assembly code using the Little Man Computer instruction set. It is *supposed* to take in two numbers and output the higher.

```
      INP
      STA   NUMA
      INP
      STA   NUMB
      SUB   NUMA
      BRP   NOTA
      LDA   NUMB
      BRA   QUIT
NOTA  LDA   NUMA
QUIT  OUT
      HLT

NUMA  DAT
NUMB  DAT
```

Fig.2

Programs can also be written in high level languages. In pseudocode write a procedural program that takes in two numbers and outputs the higher of them.

--- [4]

8(a). The following is a program written using the Little Man Computer instruction set.

```
start  LDA   one
       OUT
       LDA   zero
       OUT
       LDA   count
       SUB   one
       STA   count
       BRP   start
       HLT
one    DAT   1
zero   DAT   0
count  DAT   3
```

Describe the difference between the STA and LDA instructions.

[2]

(b). Identify the type of memory addressing the program uses.

[1]

(c). State the output this program generates.

[3]

9.

(i) A high-level language states what is required but not how to do it. The statements do not have to be in a specific order.

Identify the type of language described.

____

____ [1]

(ii) State **one** typical use for this type of language and give **one** reason for your choice.

____

____

____

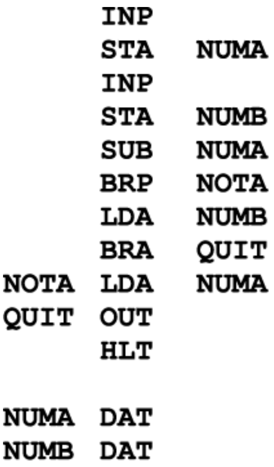
____ [2]

10(a) Explain how you would correct the program so it outputs the higher of the two numbers entered.

.

[2]

(b). The program, as shown in Fig.2 below, is written in assembly code using the Little Man Computer instruction set. It is *supposed* to take in two numbers and output the higher.



```
      INP
      STA   NUMA
      INP
      STA   NUMB
      SUB   NUMA
      BRP   NOTA
      LDA   NUMB
      BRA   QUIT
NOTA  LDA   NUMA
QUIT  OUT
      HLT

NUMA  DAT
NUMB  DAT
```

Fig.2

The program does not work correctly. Describe what the program actually does, using the numbers 4 and 9 being entered as an example.

[2]

END OF QUESTION PAPER

Question			Answer/Indicative content	Marks	Guidance
1			**Mark band 6-8. High level response.** Candidate has described all 3 terms highlighting the differences between machine code and assembly language. Candidate has used appropriate technical terminology throughout. There are few, if any, spelling errors or grammatical errors. **Mark band 3-5. Medium level response.** Candidate has described two terms adequately. Candidate has used some technical terminology in the response. There may be spelling errors or grammatical errors, but they are not obtrusive. **Mark band 0-2. Low level response.** Candidate has listed some relevant points but failed to give any detail. There is a lack of cohesion in the response. Candidate has failed to use correct technical terms in the response. Spelling and grammatical errors affect the readability of the response. *Machine code:* • Binary notation • Instructions operate on bytes of data • Dependent on architecture • Harder to program *Assembly language:* • Low level language… • …but higher level than machine code / mnemonics / hexadecimal • Uses mnemonics • Machine-specific / close to computer • Each instruction generally translated to 1 machine code instruction *Assembler tasks:* • Reserves storage for instructions & data • Replaces mnemonic opcodes by	8	Nothing about lexical analysis or code generation **Examiner's Comments** This question covered the full range of abilities for candidates and it was a good

Question			Answer/Indicative content	Marks	Guidance
			machine codes • Replaces symbolic addresses by numeric addresses • Creates symbol table • Checks syntax • Error diagnostics		discriminator of levels of ability. A small but significant amount of candidates answered about high level language compilers or interpreters rather than an assembler.
			Total	8	

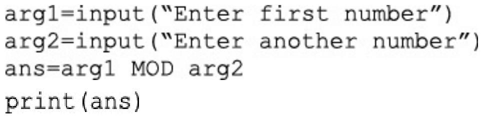
<table>
<tr><th colspan="3">Question</th><th>Answer/Indicative content</th><th>Marks</th><th>Guidance</th></tr>
<tr><td>2</td><td></td><td>i</td><td>
<table>
<tr><th>Input</th><th>Green Light</th><th>Red Light</th></tr>
<tr><td>1</td><td>✓</td><td></td></tr>
<tr><td>2</td><td>✓</td><td></td></tr>
<tr><td>3</td><td>✓</td><td></td></tr>
<tr><td>4</td><td>✓</td><td></td></tr>
<tr><td>5</td><td>✓</td><td></td></tr>
<tr><td>6</td><td></td><td>✓</td></tr>
<tr><td>7</td><td></td><td>✓</td></tr>
<tr><td>8</td><td></td><td>✓</td></tr>
<tr><td>9</td><td></td><td>✓</td></tr>
</table>
Rows 1-4 correct 1 Mark
Rows 5-9 correct 1 Mark</td><td>2 (AO3.3)</td><td>Accept T for a tick. Penalise if blank table elements have content.
<u>Examiner’s Comments</u>
Most candidates achieved the first mark on this question. The second mark was lost by those who ticked ‘red light’ for an input of 5.</td></tr>
<tr><td></td><td></td><td>ii</td><td>- The value 11 is stored in the MAR.
- 11 is sent down the address bus.
- A read signal is sent down the control bus.
- <u>0</u> is sent (back from memory) down the data bus.
- 0 is stored in the MDR…
- …and then copied to the ACC
(1 Mark per -, max 6)</td><td>6 (AO1.2)</td><td><u>Examiner’s Comments</u>
Many candidates gave a comprehensive description of the fetch execute cycle but did not apply their response to the assembly code instruction given in the question therefore losing marks.</td></tr>
</table>

Question			Answer/Indicative content	Marks	Guidance

		iii	- Takes in a value from user.
			- If value is 5 or less it shows green
			- Otherwise it shows Red
			
			(1 Mark per -, max 3)
				3 (AO 3.2)	Do not credit structured English
					Example
					
					`value = input("Enter a Value")`
					`if value <=5 then`
					`   print("GREEN")`
					`else`
					`   print(“RED”)`
					`endif`
					
					Accept equivalents to <=5 (e.g. < 6) For Green/Red (or 1/0) accept any pseudocode equivalent (GreenLightOn(), Output 1, print(1) Output Green etc.) as long as the logic is correct.
					
					Examiner’s Comments
					
					Most candidates gained credit for writing code to read the value from the user with some then losing marks for either specifying the incorrect condition e.g. IF value < 5 and/or not outputting the result.

		iv	**Mark Band 3-High Level (7-9 marks)1** The candidate demonstrates a thorough knowledge and understanding of assembly code and high level languages. The material is generally accurate and detailed. The candidate is able to apply their knowledge and understanding directly and consistently to the context provided. Evidence/examples will be explicitly relevant to the explanation. The candidate provides a thorough discussion which is well balanced. Evaluative comments are consistently relevant and well-considered. There is a well-developed line of reasoning which is clear and logically structured. The information presented is relevant and substantiated. **Mark Band 2-Mid Level (4-6 marks)** The candidate demonstrates reasonable knowledge and understanding assembly code and high level languages; the material is generally accurate but at times	9 AO1.1 (2) AO1.2 (2) AO2.1 (2) AO3.3 (3)	**AO1** Assembly code uses mnemonics to represent machine code instructions/opcodes. High level languages use more natural/mathematical notation. Assembly code consists of simple instructions As such many more lines of assembly code are required to perform the same task as a few lines of a high level language. Assembly code is specific to the instruction set of a given processor. High Level languages are not architecture specific. **AO2** Assembly code allows the programmer to choose the exact instructions so they can write code that is highly efficient. It also allows them to have direct control of how memory is used via addressing modes. Direct control of hardware. High level language compilers have optimisers that can also try and do this

Question			Answer/Indicative content	Marks	Guidance
			underdeveloped. The candidate is able to apply their knowledge and understanding directly to the context provided although one or two opportunities are missed. Evidence/examples are for the most part implicitly relevant to the explanation. The candidate provides a sound discussion, the majority of which is focused. Evaluative comments are for the most part appropriate, although one or two opportunities for development are missed. There is a line of reasoning presented with some structure. The information presented is in the most part relevant and supported by some evidence. **Mark Band 1-Low Level (1-3 marks)** The candidate demonstrates a basic knowledge assembly code and high level languages; the material is basic and contains some inaccuracies. The candidate makes a limited attempt to apply acquired knowledge and understanding to the context provided. The candidate provides a limited discussion which is narrow in focus. Judgments if made are weak and unsubstantiated. The information is basic and communicated in an unstructured way. The information is supported by limited evidence and the relationship to the evidence may not be clear. **0 marks** No attempt to answer the question or response is not worthy of credit.		(and in some cases may outperform a human writing in assembly code). As high level code is more intuitive and easier to read it is easier to follow, debug and build as part of a team. It can also be written in a much shorter time frame. The high level code can be recompiled for different architectures. High level languages come in a variety of paradigms so programmers can choose according to the problem/their preference. **AO3** Assembly language is best suited to situations such as: -compilers or interpreters don't exist for the target CPU i.e. embedded systems -highest possible performance is critical -memory is very limited. For larger projects which don't fall under the constraints above high level languages are likely to be preferable. Examiner's Comments Candidates were assessed on the quality of their extended response in this question. Most candidates could describe the basic differences between assembly code and high level languages, with many giving examples of where each would be best used. Some candidates gave clear and appropriate justification for the coffee machine being programmed in assembly code. In general, most candidates scored reasonably well on this question.
			Total	20	
3		i	& immediate addressing	1	
		ii	# indirect addressing	1	
			Total	2	

Question			Answer/Indicative content	Marks	Guidance
4		i	• Calculates the remainder of two numbers when the second is divided by the first (1).	1	For 1 mark. Accept finds modulo / modulus.
		ii	• Code takes in two values and provides an output (1). • The output is the modulus of the two inputs (1).	2	For 2 marks. Allow follow through for second mark if output matches answer to (i). Accept MOD, % or any existing alternative. Accept if candidate has calculated modulus using alternative method (e.g. using a loop). Example:  arg1=input("Enter first number") arg2=input("Enter another number") ans=arg1 MOD arg2 print(ans)
			Total	3	
5			A program that translates assembly code (1) into machine code/object code (1)	2	
			Total	2	

Question			Answer/Indicative content	Marks	Guidance
6	a		declarative	1	Throughout question, accept any appropriate example using the statements given in question **Examiner's Comments** Most could name the type of language as there was only one possible answer.
	b		e.g. studies_science (A,B) if student (A) and science (B)	1	**Examiner's Comments** Most candidates achieved this mark.
	c		A problem that needs to be solved e.g. student (X) ?	2	Accept e.g. “searching for a list of students” with either of answers here **Examiner's Comments** Those candidates that did not gain the full two marks here did so because their answers were either too vague or lacked an example.
	d		Setting an initial value to replace a variable e.g. find X=ben, set X=ben to test science(Y)	2	**Examiner's Comments** Another question that was set for the higher ability candidates and results on this were as expected with only a few candidates getting the full two marks on this.
	e		After finding a solution / failing to find a solution… …go back to an earlier step to test an alternative	2	Accept example that demonstrates this **Examiner's Comments** Mostly well answered by those who knew what the program was meant to achieve, a few wild guesses from candidates otherwise.
			Total	8	

Question			Answer/Indicative content	Marks	Guidance

7			• Takes in two numbers (1).
			• Compare the numbers (1).
			• If first number is biggest outputs first number (1).
			• If second number is biggest outputs the second number (1).
				4	For 4 marks – 1 mark for each correct step in process.
					Example:
					`INPUT "Please enter Number A" numA`
					`INPUT "Please enter Number A" numB`
					`IF numA>numB THEN`
					`      PRINT numA`
					`ELSE`
					`      PRINT numB`
					`ENDIF`

			Total	4	
8	a		– STA store the value in the accumulator into a given memory location – LDA loads the value in a memory location into the accumulator. (1 per –)	2	
	b		Direct addressing	1	Accept Symbolic Addressing
	c		– Answer contains at least 1 followed by 0 – Answer contains at least three 10s – Answer contains exactly four 10s (1 per –)	3	1 0 1 0 1 0 1 0 NB allow answers that are vertical or horizontal.
			Total	6	

Question			Answer/Indicative content	Marks	Guidance
9		i	• Declarative	1	**Examiner's Comments** Most candidates answered this correctly.
		ii	eg *Use:* • Medical diagnosis • Expert systems *Reason:* • Answer to one question affects the next question / Can find alternative solutions	2	Max one mark for use and max one mark for reason. Accept other example uses with reasons **Examiner's Comments** Nearly all candidates achieved at least one mark in this question.
			Total	3	
10	a		Award first mark: Changing LDA NUMB to LDA NUMA (1). Award second mark: Changing NOTA LDA NUMA to NOTA LDA NUMB (1).	2	For 2 marks. Accept changes annotated on provided code. Accept any other amendment that fixes program.
	b		• Program outputs smaller number (1) so in the case of 4 and 9 outputs 4 (1).	2	Up to 2 marks for a valid description.
			Total	4	