Past Papers May/June 2015 to 2018:

Q# 1
(a)

LDI 103

ACC:								

Main memory

100	0000 0010
101	1001 0011
102	0111 0011
103	0110 1011
104	0111 1110
105	1011 0001
106	0110 1000
107	0100 1011
...	
200	1001 1110

(b)

LDX 800

Index Register:	0	0	0	0	1	0	0	1
Accumulator:								

9608/11/M/J/16

Q.9 The table shows assembly language instructions for a processor which has one general purpose register, the Accumulator (ACC) and an index register (IX).

Instruction		Explanation
Op code	**Operand**	
`LDD`	`<address>`	Direct addressing. Load the contents of the given address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
`STO`	`<address>`	Store contents of ACC at the given address.
`ADD`	`<address>`	Add the contents of the given address to ACC.
`INC`	`<register>`	Add 1 to the contents of the register (ACC or IX).
`DEC`	`<register>`	Subtract 1 from the contents of the register (ACC or IX).
`CMP`	`<address>`	Compare contents of ACC with contents of <address>.
`JPE`	`<address>`	Following a compare instruction, jump to <address> if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to <address> if the compare was False.
`JMP`	`<address>`	Jump to the given address.
`OUT`		Output to screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

(a) The diagram shows the current contents of a section of main memory and the index register:

60	0011 0010
61	0101 1101
62	0000 0100
63	1111 1001
64	0101 0101
65	1101 1111
66	0000 1101
67	0100 1101
68	0100 0101
69	0100 0011
...	
1000	0110 1001

Index register:

0	0	0	0	1	0	0	0

(i) Show the contents of the Accumulator after the execution of the instruction:

```
LDX 60
```

Accumulator:								

Show how you obtained your answer.

....

....

....

....[2]

(ii) Show the contents of the index register after the execution of the instruction:

```
DEC IX
```

Index register:								

[1]

(b) Complete the trace table on the opposite page for the following assembly language program.

IX (Index Register)	1

Selected values from the ASCII character set:

ASCII Code	118	119	120	121	122	123	124	125
Character	v	w	x	y	z	{	|	}

50	LDD 100
51	ADD 102
52	STO 103
53	LDX 100
54	ADD 100
55	CMP 101
56	JPE 58
57	JPN 59
58	OUT
59	INC IX
60	LDX 98
61	ADD 101
62	OUT
63	END
...	
100	20
101	100
102	1
103	0

Trace table:

Instruction address	Working space	ACC	Memory address				IX	OUTPUT
			100	101	102	103		
			20	100	1	0	1	
50								
51								
52								
53								
54								
55								

[7]

Exam-style Questions

Q.2 Complete the trace table below for the following assembly language program.

800	LDD 810
801	INC ACC
802	STO 812
803	LDD 811
804	ADD 812
805	STO 813
806	END
...	
810	28
811	41
812	0
813	0

Trace table:

ACC	Memory address			
	810	811	812	813
	28	41	0	0

9608/13/M/J/16

Q4 The table shows assembly language instructions for a processor which has one general purpose register, the Accumulator (ACC) and an index register (IX).

Instruction		Explanation
Op code	**Operand**	
LDD	<address>	Direct addressing. Load the contents of the given address to ACC.
LDX	<address>	Indexed addressing. Form the address from <address> + the contents of the index register. Copy the contents of this calculated address to ACC.
STO	<address>	Store contents of ACC at the given address.
ADD	<address>	Add the contents of the given address to ACC.
INC	<register>	Add 1 to the contents of the register (ACC or IX).
DEC	<register>	Subtract 1 from the contents of the register (ACC or IX).
CMP	<address>	Compare contents of ACC with contents of <address>.
JPE	<address>	Following a compare instruction, jump to <address> if the compare was True.
JPN	<address>	Following a compare instruction, jump to <address> if the compare was False.
JMP	<address>	Jump to the given address.
OUT		Output to screen the character whose ASCII value is stored in ACC.
END		Return control to the operating system.

The diagram shows the contents of the index register:

Index register:	1	1	0	0	1	1	0	1

(a) Show the contents of the index register after the execution of the instruction:

`INC IX`

Index register:								

[1]

(b) Complete the trace table on the opposite page for the following assembly language program.

Selected values from the ASCII character set:

ASCII Code	65	66	67	68	69	70	71	72
Character	A	B	C	D	E	F	G	H

Trace table:

Instruction	Working space	ACC	Memory address				IX	OUTPUT
			90	91	92	93		
			2	90	55	34	2	
20								
21								
22								
23								
24								
25								
26								

20	`LDX 90`
21	`DEC ACC`
22	`STO 90`
23	`INC IX`
24	`LDX 90`
25	`DEC ACC`
26	`CMP 90`
27	`JPE 29`
28	`JPN 31`
29	`ADD 90`
30	`OUT`
31	`ADD 93`
32	`STO 93`
33	`OUT`
34	`END`
⋮	
90	`2`
91	`90`
92	`55`
93	`34`

[7]

9608/11/M/J/17

Q.4/- The following table shows part of the instruction set for a processor. The processor has one general purpose register, the Accumulator (ACC) and an Index Register (IX).

Instruction			Explanation
Op code (mnemonic)	**Operand**	**Op code (binary)**	
`LDM`	`#n`	0000 0001	Immediate addressing. Load the denary number `n` to ACC.
`LDD`	`<address>`	0000 0010	Direct addressing. Load the contents of the location at the given address to ACC.
`LDI`	`<address>`	0000 0101	Indirect addressing. At the given address is the address to be used. Load the contents of this second address to ACC.
`LDX`	`<address>`	0000 0110	Indexed addressing. Form the address from `<address>` + the contents of the Index Register (IX). Copy the contents of this calculated address to ACC.
`LDR`	`#n`	0000 0111	Immediate addressing. Load number `n` to IX.
`STO`	`<address>`	0000 1111	Store the contents of ACC at the given address.

The following diagram shows the contents of a section of main memory and the Index Register (IX).

(a) Show the contents of the Accumulator (ACC) after each instruction is executed.

IX								
	0	0	0	0	0	0	1	1

Address	Main Memory contents
495	13
496	86
497	92
498	486
499	489
500	496
501	497
502	499
503	502

(i) `LDM #500`

ACC ....[1]

(ii) `LDD 500`

ACC ....[1]

(iii) `LDX 500`

ACC ....[1]

(iv) `LDI 500`

ACC ....[1]

(b) Each machine code instruction is encoded as 16-bits (8-bit op code followed by an 8-bit operand). Write the machine code for the following instructions:

`LDM #17`

`LDX #97`

[3]

(c) Using an 8-bit operand, state the maximum number of memory locations, in denary, that can be directly addressed.

....[1]

(d) Computer scientists often write binary representations in hexadecimal.

(i) Write the hexadecimal representation for this instruction:

0	0	0	0	0	1	1	1	1	1	0	0	0	0	1	0

....[2]

(ii) A second instruction has been written in hexadecimal as:

05 3F

Write the equivalent assembly language instruction, with the operand in denary.

....[2]

9608/12/M/J/17

Q5 The following table shows part of the instruction set for a processor. The processor has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

Instruction			
Op code (mnemonic)	**Operand**	**Op code (binary)**	**Explanation**
`LDD`	`<address>`	`0001 0011`	Direct addressing. Load the contents of the location at the given address to the Accumulator (ACC).
`LDI`	`<address>`	`0001 0100`	Indirect addressing. The address to be used is at the given address. Load the contents of this second address to ACC.
`LDX`	`<address>`	`0001 0101`	Indexed addressing. Form the address from `<address>` + the contents of the Index Register. Copy the contents of this calculated address to ACC.
`LDM`	`#n`	`0001 0010`	Immediate addressing. Load the denary number `n` to ACC.
`LDR`	`#n`	`0001 0110`	Immediate addressing. Load denary number `n` to the Index Register (IX).
`STO`	`<address>`	`0000 0111`	Store the contents of ACC at the given address.

The following diagram shows the contents of a section of main memory and the Index Register (IX).

(a) Show the contents of the Accumulator (ACC) after each instruction is executed.

IX	0	0	0	0	0	1	1	0

(i) `LDD 355`

ACC .... [1]

(ii) `LDM #355`

ACC .... [1]

(iii) `LDX 351`

ACC .... [1]

(iv) `LDI 355`

ACC .... [1]

Address	Main memory contents
350	
351	86
352	
353	
354	
355	351
356	
357	22
358	

(b) Each machine code instruction is encoded as 16 bits (8-bit op code followed by an 8-bit operand). Write the machine code for these instructions:

`LDM #67`

`LDX #7`

[3]

(c) Computer scientists often write binary representations in hexadecimal.
(i) Write the hexadecimal representation for the following instruction.

0	0	0	1	0	1	0	0	0	1	0	1	1	1	1	0

....[2]

(ii) A second instruction has been written in hexadecimal as:

`16 4D`

Write the assembly language for this instruction with the operand in denary.

....[2]

9608/11/O/N/16
Q.8/- The table shows assembly language instructions for a processor which has one general purpose register, the Accumulator (ACC) and an Index Register (IX).

Instruction		Explanation
Op code	**Operand**	
`LDD`	`<address>`	Direct addressing. Load the contents of the given address to ACC.
`LDX`	`<address>`	Indexed addressing. Form the address from `<address>` + the contents of the index register. Copy the contents of this calculated address to ACC.
`STO`	`<address>`	Store contents of ACC at the given address.
`ADD`	`<address>`	Add the contents of the given address to ACC.
`CMP`	`<address>`	Compare contents of ACC with contents of `<address>`
`JPE`	`<address>`	Following a compare instruction, jump to `<address>` if the compare was True.
`JPN`	`<address>`	Following a compare instruction, jump to `<address>` if the compare was False.
`JMP`	`<address>`	Jump to the given address.
`OUT`		Output to the screen the character whose ASCII value is stored in ACC.
`END`		Return control to the operating system.

The diagram shows the contents of the main memory:

Main memory

800	0110 0100
801	0111 1100
802	1001 0111
803	0111 0011
804	1001 0000
805	0011 1111
806	0000 1110
807	1110 1000
808	1000 1110
809	1100 0010
⋮	
2000	1011 0101

(a) (i) Show the contents of the Accumulator after execution of the instruction:

```
LDD  802
```

Accumulator:

[1]

(ii) Show the contents of the Accumulator after execution of the instruction:

```
LDX 800
```

Index Register:

0	0	0	0	1	0	0	1

Accumulator:

Explain how you arrived at your answer.

....

....

....

....[3]

(b) (i) Complete the trace table below for the following assembly language program. This program contains denary values

100	LDD 800
101	ADD 801
102	STO 802
103	LDD 803
104	CMP 802
105	JPE 107
106	JPN 110
107	STO 802
108	OUT
109	JMP 112
110	LDD 801
111	OUT
112	END
⋮	
800	40
801	50
802	0
803	90

Selected values from the ASCII character set:

ASCII code	40	50	80	90	100
Character	(	2	P	Z	d

Trace table:

ACC	Memory address				OUTPUT
	800	801	802	803	
	40	50	0	90	

[4]

(ii) There is a redundant instruction in the code in part (b)(i).
State the address of this instruction.

....[1]

Q#1 Answer:

(a) Answer:

0	1	0	0	1	0	1	1

- Memory address 103 contains the value 107
- So address 107 is the address from which to load the data

(b) Answer:

Accumulator:	**1**	**1**	**0**	**0**	**0**	**0**	**1**	**0**

- Index Register contains: **00001001 = 9**
- **800 + 9 = 809**

9608/11/M/J/16

Q.

9 **(a) (i)** One mark for the contents of the accumulator and one mark for the reason. [2]
Accumulator contents: 0100 0101

Reason:

Address is 60 Contents of the index register is 8 And 60 + 8 = 68 in denary gives the address The contents of which is 0100 0101 in binary.

(ii) 0000 0111 [1]

(b)

One mark for each shaded block.

- Contents of the Accumulator in first 2 lines (instruction addresses 50 and 51)
- Updating address 103 (instruction 52)
- Loading the Accumulator and addition (instructions 53 and 54)Not executing instruction 58
- Incrementing the index register (instruction 59)
- Loading the Accumulator and addition (instructions 60 and 61)
- Correct output of 'x' (instruction 62)

[7]

Instruction address	Working space	ACC	Memory address				IX	OUTPUT
			100	101	102	103		
			20	100	1	0	1	
50		20						
51		21						
52						21		
53		100						
54		120						
55								
56								
57								
59							2	
60		20						
61		120						
62								'x'
63								

9608/13/M/J/16
Answer

4 (a) 11001110 **[1]**

(b) **[7]**

Instruction	Working space	ACC	Memory address				IX	OUTPUT
			90	91	92	93		
			2	90	55	34	2	
20		55						
21		54						
22			54					
23							3	
24		34						
25		33						
26								
27								
28								
31		67						
32						67		
33								'C'
34								

Q.2 Answer

- **LDD 810** (28 Loaded in ACC)
- **INC ACC** (Accumulator incremented with 28++1 = 29, 29 written in ACC)
- **STO 812** (29 Stored at Memory Location 812)
- **LDD 811** (Loaded contents of memory location 811 in ACC)
- **ADD 812** (Added 41 with 29, Contents of ACC added with memory loc 812)
- **STO 813** (Stored contents of ACC in 813 memory location)

ACC	Memory address			
	810	811	812	813
	28	41	0	0
28				
29				
			29	
41				
70				
				70

Answer

9608/11/M/J/17

Q.4/-

Question	Answer	Marks
4(a)(i)	500	1
4(a)(ii)	496	1
4(a)(iii)	502	1
4(a)(iv)	86	1
4(b)	0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 1 1 0 0 1 1 0 0 0 0 1 Both correct op codes 1 Operand 0001 0001 1 Operand 0110 0001 1	3
4(c)	256	1
4(d)(i)	07 C2 07 1 C2 1	2
4(d)(ii)	LDI 63 LDI 1 63 1	2

9608/12/M/J/17

Q5

Question	Answer	Marks
5(a)(i)	351	1
5(a)(ii)	355	1
5(a)(iii)	22	1
5(a)(iv)	86	1
5(b)	Op code Operand 0 0 0 1 0 0 1 0 0 1 0 0 0 0 1 1 0 0 0 1 0 1 0 1 0 0 0 0 0 1 1 1 Both correct op codes 1 Operand 0100 0011 1 Operand 0000 0111 1	3
5(c)(i)	14 5E 14 1 5E 1	2
5(c)(ii)	LDR #77 LDR 1 #77 1	2

9608/11/O/N/16
Answer Q.8/-

8 (a) (i)

Accumulator:	1	0	0	1	0	1	1	1

[1]

(ii) **One mark** for answer and **two marks** for explanation

Accumulator:	1	1	0	0	0	0	1	0

- Index Register contains 1001 = 9
- 800 + 9 = 809

[3]

(b) (i) **ONE** mark for each correct row.

ACC	Memory address				OUTPUT
	800	801	802	803	
	40	50	0	90	
40 90			 90		
90			90		
					Z

[4]

(ii) **107** [1]