The University of Nottingham

SCHOOL OF COMPUTER SCIENCE

A LEVEL 1 MODULE, AUTUMN SEMESTER 2020-2021

COMPUTER FUNDAMENTALS (COMP1036)

Time allowed: **60 Mins**

Candidates may complete the front cover of their answer book and sign their desk card but must NOT write anything else until the start of the examination period is announced

Answer All Questions

Only silent, self contained calculators with a Single-Line Display or Dual-Line Display are permitted in this examination

Dictionaries are not allowed with one exception. Those whose first language is not English may use a standard translation dictionary to translate between that language and English provided that neither language is the subject of this examination. Subject specific translation dictionaries are not permitted.

No electronic devices capable of storing and retrieving text, including electronic dictionaries, may be used.

DO NOT turn your examination paper over until instructed to do so

Question 1 (25 marks)

a. Convert the signed decimal number -120 to Octal, Hexadecimal and 8-bit 2s complement binary numbers.

[4 marks]

b. Draw gate diagrams for the **`Or`** gate and the **`Not`** gate using only **`Nand`** chips.

[4 marks]

c. Describe the key components of a 1-bit register, including how its output is related to its input at each time interval.

[5 marks]

d. Using only canonical chips, draw the simplest possible circuit for the following truth table where A, B, C are inputs and y is the output. Show how you derive this circuit.

A	B	C	y
0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

[6 marks]

e. What is a Turing machine and how does it relate to the von Neumann architecture.

[4 marks]

f. What is the order of precedence for the logical operators: **`Not And () Or`**

[2 marks]

Question 2 (25 Marks)

a. This question is about machine language. Based on the symbolic assembly code below,

```
    @R0
    D=M
    @x
    M=D
    @i
    M = 0
    @2
    D=A
    @mResult
    M=D

(LOOP)
    @mResult
    D=M
    @x
    D=M-D
    @STOP
    D;JLT
    @mResult
    D=M
    M=D+M
    @i
    M = M + 1
    @LOOP
    0;JMP

(STOP)
    @i
    D=M
    @R1
    M=D

(END)
    @END
    0;JMP
```

I. if RAM[0] is initially set to 60, derive the value of RAM[1] after the execution of this piece of code.

[4 Marks]

II. Please convert the last two lines of the symbolic assembly code in (a),

```
@END
0;JMP
```

to **binary machine code**. You may refer to APPENDIX 1 and 2 for this conversion.

[2 Marks]

b. Given the following pseudo-code, implement it in **symbolic assembly language**. You **MUST** use built-in symbols. The built-in symbols are given in APPENDIX 3. You **MUST** use labels. You **MUST** terminate the program properly.

// This program implements an infinite loop to listen to the keyboard input

// Step 1: check keyboard input and use R0 to record keyboard input

// Step 2: if R0 = 'Esc', goto END. (The scan-code for 'Esc' is 140)

// Step 3: Jump back to the beginning of the code to form an infinite loop.

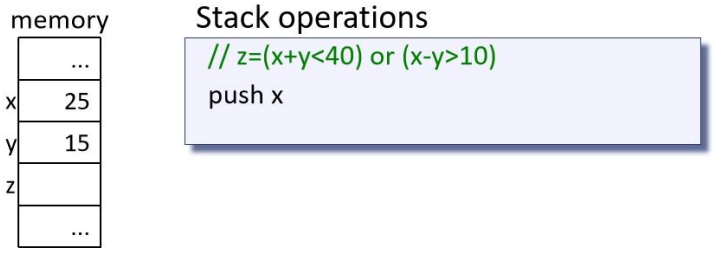
// Step 4: Terminate the program

[4 Marks]

c. This question is about stack operation. Show the **stack operations** and the **final memory status** for the following operations:

z = (x-y>10) and (x<15).

The initial memory status and the first operation are shown below.



[6 Marks]

d. Given the following VM commands, show the **final** stack status. Assume initially *local 0* takes the value of 3 and *local 1* takes the value of 5.

```
push constant 0
pop local 2
label LOOP_START
push local 2
push local 0
add
pop local 2
push local 1
push constant 1
sub
pop local 1
push local 1
if-goto LOOP_START
push local 2
```

[4 Marks]

e. Translate the VM command "pop local 2" into **hack symbolic assembly code.**

[5 Marks]

End of Question 2

APPENDIX

1. A-instruction specification

Symbolic syntax:

@value

Binary syntax:

0value

2. C-instruction specification

Symbolic syntax: *dest* = *comp* ; *jump*

Binary syntax: 1 1 1 a c1 c2 c3 c4 c5 c6 d1 d2 d3 j1 j2 j3

opcode — not used — *comp* bits — *dest* bits — *jump* bits

comp		c1	c2	c3	c4	c5	c6
0		1	0	1	0	1	0
1		1	1	1	1	1	1
-1		1	1	1	0	1	0
D		0	0	1	1	0	0
A	M	1	1	0	0	0	0
!D		0	0	1	1	0	1
!A	!M	1	1	0	0	0	1
-D		0	0	1	1	1	1
-A	-M	1	1	0	0	1	1
D+1		0	1	1	1	1	1
A+1	M+1	1	1	0	1	1	1
D-1		0	0	1	1	1	0
A-1	M-1	1	1	0	0	1	0
D+A	D+M	0	0	0	0	1	0
D-A	D-M	0	1	0	0	1	1
A-D	M-D	0	0	0	1	1	1
D&A	D&M	0	0	0	0	0	0
D|A	D|M	0	1	0	1	0	1
a==0	a==1						

dest	d1	d2	d3	effect: the value is stored in:
null	0	0	0	The value is not stored
M	0	0	1	RAM[A]
D	0	1	0	D register
MD	0	1	1	RAM[A] and D register
A	1	0	0	A register
AM	1	0	1	A register and RAM[A]
AD	1	1	0	A register and D register
AMD	1	1	1	A register, RAM[A], and D register

jump	j1	j2	j3	effect:
null	0	0	0	no jump
JGT	0	0	1	if out > 0 jump
JEQ	0	1	0	if out = 0 jump
JGE	0	1	1	if out ≥ 0 jump
JLT	1	0	0	if out < 0 jump
JNE	1	0	1	if out ≠ 0 jump
JLE	1	1	0	if out ≤ 0 jump
JMP	1	1	1	Unconditional jump

3. Built-in symbols of Hack assembly code.

symbol	value
R0	0
R1	1
...	...
R15	15
SCREEN	16384
KBD	24576

symbol	value
SP	0
LCL	1
ARG	2
THIS	3
THAT	4