# Homework 4 Assembly language

1. The following program is written for the Little Man Computer.   
The user enters the values 3, 5, 6.

INP

STA p

INP

STA q

INP

STA r

LDA p

ADD r

SUB q

OUT

HLT

p DAT

q DAT

r DAT

(a) What is the effect of the statement **STA p** in the second line of the program? [1]

Stores the value currently in the accumulator (so the inputed value from previous line) into the variable named p

(b) What does the statement **SUB** **q** do? [1]

Subtract from the value in the accumulator, the value of the variable q

(c) What mathematical expression does the program calculate? [1]

P+R-Q

(d) What is the output of the program? [1]

1

2. The following program is written for the Little Man Computer. The user enters the value 3.

INP

STA a

start  LDA a

OUT

SUB one

STA a

BRZ finish

BRA start

finish LDA a

SUB a

OUT

HLT

a   DAT

one DAT 1

(a) What does the statement **BRZ** finish do? [2]

Branch (basically skip) to the line with the label that has been referenced by the BRZ if the value in the accumulator is 0

(b) What does the statement **one DAT 1** do? [1]

Defines a variable named “one”, with the starting value of 1

(c) What is the output of the program? [2]

3, 2, 1, 0

(d) Some of the statements are not needed. Rewrite the assembly language program in as few statements as possible. [4]

INP

STA a

start  LDA a

OUT

SUB one

STA a

BRP start

HLT

a   DAT

one DAT 1

3. (a) Explain what is meant by the terms **opcode** and **operand.** [2]

Opcode:

The type of instruction is, like for example if the instruction is LDA 06, LDA is the opcode which is the type of instruction, and it says to load the operand into accumulator

Operand:

Operand is the data or location of the data we want to do the instruction on.

(b) Explain what is meant by each of the following addressing modes:

(i) Direct [1]

Addresses the data or value held by the operand, or memory address we have

(ii) Indirect [1]

The operand we have is the location of the memory address that leads to the data we want, so if you indirectly address an operand, we go to the memory location being stored in the location of the operand, and then address the value stored in that memory location in the location of the operand

(iii) Immediate [1]

Takes the value of the operand as the data.

[Total 18 marks]