Practical Sheet 5

Computer Architecture and Assembly

1 Fetch-Execute and the Little Man Computer

1.1 Simple Programs

Exercise 1.1: Enter a Program in LMC

Consider the following Python program:

```
z = x + y
```

This can be compiled (*note: you are the compiler*) into the follow LMC program

```
        LDA   x
        ADD   y
        STA   z                 or STO      z
        HLT
x       DAT   11
y       DAT   17
z       DAT
```

Enter this program and assemble it. Answer the following questions:

1. What is the memory location for the variable 'x'?
2. The first memory location has value '504'. Explain how this value represents the first instruction in the program.
3. Describe the program in words.

Exercise 1.2: Fetch-Execute in LMC

The program of Exercise 1.1 has 4 instructions. Run the program, step by step and complete the following table to show the state of the system after each step:

Step	Program Counter	Memory Address Register	Memory Data Register	Accumulator (Calculator)
0	00	---	---	000
1				
2				
3				

Note: the names of the registers vary a bit between different LMC versions

In addition, look at which memory location is change.

Exercise 1.3: Programming Challenges

Try any of the following:

1. Input two numbers and output the sum
2. Input three numbers and output the sum
3. Input a number, double it twice and output (e.g. 7 input gives 28 out).

1.2 Example Program 2: If Statements

Exercise 1.4: Largest of Two

Enter the 'largest of two' program described in the reference notes and test it with different values. Step through it and look at the value of the program counter.

Can you explain how the 'if statement' has been compiled using the `BRP` instruction? Note: if you know about flowcharts, you might find it useful to draw a flowchart of the program.

Exercise 1.5: LMC Challenge Problems Write a MC program to solve the following problems:

1. Input two numbers and output the smaller one
2. Input three numbers and output the largest.

You are recommended to write the program in simplified Python (see the reference notes) and/or draw a flowchart before attempting to write the assembly code.

1.3 Example Program 3: Loops

Exercise 1.6: LMC Counting Down

Enter the 'down counter' loop program described in the reference notes and test it with different values. Step through it and look at the value of the program counter. Again, explain how the loop is translated to LMC using a flowchart.

Exercise 1.7: LMC Looping Challenge Problems Try any of the following:

1. Enter two numbers C, N: C is a counter and N is a number. Output the first C multiples of N, starting with 0. So if C is 5 and N is 3, the output is 0, 3, 6, 9, 12.
2. LMC does not have a multiply instructions but multiplication can be done by repeatedly adding. For example 3 x 4 is equal to 4 + 4 + 4.

2 Interpreters

Exercise 2.1: Write an Interpreter for the LMC following the approach outlined in the notes. Here is an example of a possible solution being used.

```
Load (L) Run(R) Stop(S) > R
MAR = 0 MDR = 0 ACC = 0 PC = 0
Program halted
Load (L) Run(R) Stop(S) > L
Location = 0 Enter value (or '.') 504
Location = 1 Enter value (or '.') 105
Location = 2 Enter value (or '.') 306
Location = 3 Enter value (or '.') 0
Location = 4 Enter value (or '.') 11
Location = 5 Enter value (or '.') 17
Location = 6 Enter value (or '.') .
Load (L) Run(R) Stop(S) > R
MAR = 0 MDR = 0 ACC = 0 PC = 0
Press enter to continue
MAR = 4 MDR = 11 ACC = 11 PC = 1
Press enter to continue
MAR = 5 MDR = 17 ACC = 28 PC = 2
Press enter to continue
MAR = 6 MDR = 28 ACC = 28 PC = 3
Program halted
Load (L) Run(R) Stop(S) >
```