

## Sample Assignment 1

In this assignment, you are asked to write a program in LC-3 machine language that checks a positive 2's complement number to determine if it is a power of 2. The number is stored in memory location x3050. If the number is a power of 2, your program should store x0001 in memory location x3051. If the number is not a power of 2, your program should store x0000 in memory location x3051. Your program should start at memory location x3000, and should halt the machine after storing the output.

Solution

<https://github.com/chiragsakhuja/lc3tools/blob/master/frontend/grader/solutions/pow2.bin>

## Sample Assignment 2

Given an interval ( $x_{low}$ ,  $x_{high}$ ), a function  $f(x)$  that is monotonic in that interval, and either  $f(x_{low})$  is positive and  $f(x_{high})$  is negative or  $f(x_{low})$  is negative and  $f(x_{high})$  is positive, write an LC-3 assembly language program that uses binary search to find the zero of  $f(x)$  in the interval ( $x_{low}$ ,  $x_{high}$ ), and store that value of  $x$  in  $x4000$ .

Input to your program will be found in a table in memory, starting at  $x4001$ :

Address	Content
$x4001$	$x_{low}$
$x4002$	$x_{high}$
$x4003$	degree of the polynomial, say $n$
$x4004$ to $x4004 + n$	polynomial coefficients

For example, if  $f(x)$  is the polynomial  $Ax^3 + Bx^2 + Cx + D$ , the table will take the form:

Address	Content
$x4000$	output: $x$ -position of zero
$x4001$	left bound
$x4002$	right bound
$x4003$	degree: 3
$x4004$	A
$x4005$	B
$x4006$	C
$x4007$	D

Solution

<https://github.com/chiragsakhuja/lc3tools/blob/master/frontend/grader/solutions/polyroot.asm>

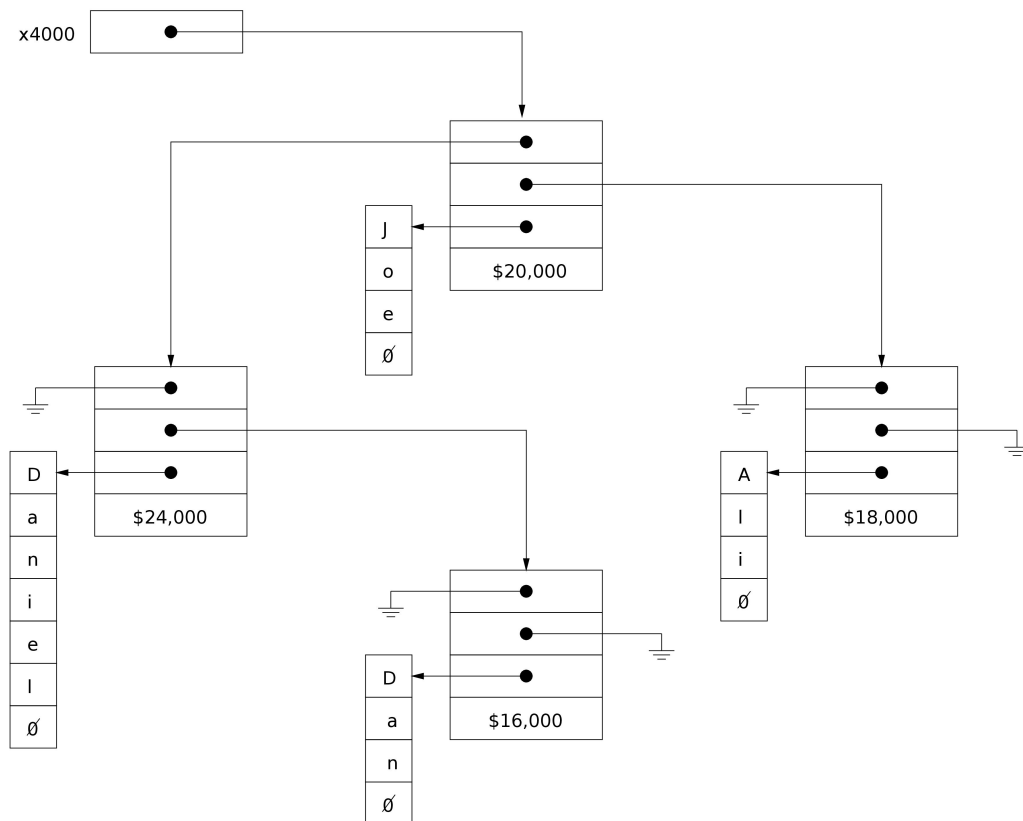
## Sample Assignment 3

Write a program in LC-3 assembly language that prompts the user to enter the name of a professor, searches a database for that professor, and prints to the screen that professor's salary. The database of professors will be stored in memory as a binary tree. We refer to the "top" node of a binary tree as the ROOT of the tree. The address of the root can be found at memory location x4000.

### The Binary Tree

A binary tree is a special case of a tree where each node has at most two child nodes. We refer to the two children as the left child and the right child. In this lab, each node of the tree will consist of 4 words: a pointer to the left child if one exists, a pointer to the right child if one exists, a pointer to the ASCII string containing the professor's name, and a 2's complement number representing the professor's salary. If there is no left or right child, the corresponding word will contain the null pointer.

Note that a binary tree consists of a root and up to two children, each of which is in itself the root of a binary tree. In the example below, Joe is the root of the binary tree, its child Daniel is the root of a binary tree often referred to as the left subtree, and its child Ali is the root of a binary tree often referred to as the right subtree. Here is an example tree:



### Input/Output Requirements

Described below are detailed requirements about the inputs and outputs of your program. All prompts and outputs should be printed to the screen EXACTLY as shown below to receive full credit.

**Input:** Your program should print the following prompt to the screen: Type a professor's name and then press Enter:

The user will input either a professor's name or the letter 'd', signifying that the user has no more professors to look up, followed by the Enter key. You must echo each character (including the enter key) to console.

**Output:** If there is a match, print the professor's salary to the console. If there is no match, print No Entry to the console. If the letter 'd' is typed (followed by the enter key), do not print anything to the console. To assist with printing salaries larger than a single ASCII character, we have provided you with a [subroutine](#) that takes as input a 2's complement number in R0 and prints it to the console. You may use this subroutine in any way you choose.

### Example:

Type a professor's name and then press Enter:Dan  
16000

Type a professor's name and then press Enter:Dani  
No Entry

Type a professor's name and then press Enter:Daniel  
24000

Type a professor's name and then press Enter:dan  
No Entry

Type a professor's name and then press Enter:d

----- Halting the processor -----

Solution

<https://github.com/chiragsakhuja/lc3tools/blob/master/frontend/grader/solutions/binsearch.asm>

## Sample Assignment 4

### The user program

Your user program will consist of continually printing "Input a capital letter from the English alphabet:" on the display, ending each line with a newline character (ASCII code x000A).

Example output:

```
Input a capital letter from the English alphabet:
Input a capital letter from the English alphabet:
Input a capital letter from the English alphabet:
Input a capital letter from the English alphabet:
Input a capital letter from the English alphabet:
```

To ensure the output on the screen is not too fast to be seen by the naked eye, the user program should include a piece of code that will count down from 40000 each time a line is output on the screen. A simple way to do this is with the following subroutine DELAY:

```
DELAY    ST  R1, SaveR1
          LD  R1, COUNT
REP       ADD R1,R1,#-1
          BRnp REP
          LD  R1, SaveR1
          RET
COUNT   .FILL #40000
SaveR1    .BLKW 1
```

Feel free to change the number 40000 to speed up or slow down outputting the prompt.

### The keyboard interrupt service routine

The keyboard interrupt service routine will examine the key typed to see if it is a capital letter in the English alphabet.

If the character typed is NOT a capital letter, the interrupt service routine will, **starting on a new line on the screen**, print "<the input character> is not a capital letter in the English alphabet.". For example, if the input key is '#', interrupt service routine will print

```
# is not a capital letter in the English alphabet.
```

If the character typed IS a capital letter, the interrupt service routine will, **starting on a new line on the screen**, print "The lower case character of *<the input character>* is: *<the lower case of the input>*". For example, if the input key is 'G', interrupt service routine will print

The lower case character of G is g.

**The service routine would then print a line feed (x0A) to the screen**, and finally terminate with an RTI.

Your interrupt service routine should start at the location x1500.

**Hint:** Don't forget to save and restore any registers that you use in the interrupt service routine.

### **The operating system enabling code**

Unfortunately, we have not YET installed Windows or Linux on the LC-3, so we are going to have to ask you to do the following two enabling actions in your user program first, before your user program starts outputting the prompts. Normally, these would be done by the operating system before your user program starts executing.

1. Normally, the operating system establishes the interrupt vector table to contain the starting addresses of the corresponding interrupt service routines. You will have to do that for the keyboard interrupt. The starting address of the interrupt vector table is x0100 and the interrupt vector for the keyboard is x80. It is necessary for you to only provide the one entry in the interrupt vector table that is needed for this programming lab assignment.
2. Normally, the operating system would set the IE bit of the KBSR. You will have to do that as well.

You should write a TRAP routine to perform the two tasks listed above, since they require privileged access. The TRAP routine should start at location x2000 and it should be accessed by vector x30. The starting address of the trap vector table is x0000.

Solution

<https://github.com/chiragsakhuja/lc3tools/blob/master/frontend/grader/solutions/interrupt.asm>