# Introduction to Computing System

## Assignment 2

Due date: 23:55 on August 7, 2015
Yale N. Patt
Xiaohong Jiang
Summer 2015

**Note:** You MUST do the programming assignment by yourself. You are permitted to get help ONLY from the TAs and instructors. The file you submit should be an assembly language file called **directory.asm**.

## Office Directory

### Overview

You will design a user interface of an office directory for an office building. Your program will prompt the user for a room number and print the last-name of the faculty in that office room (only if it exists in the directory). For example, since Prof. Jiang's office is located in room 520, if the user types '520' at the prompt, your program will print 'jiang'.

### Your job

Write a program in LC-3 assembly language that will
1. Prompt the user to type a room number
2. Search the directory to find the room and the professors' names in that office
3. Print the last-names of all faculties in that office on the screen
4. Halt the machine.

We now discuss each step in turn.

1. **Prompting for Room Number:** Your program will print on the screen the string "Type a room number and press Enter: " and wait for the user to input a string (followed by an <Enter>).

Once the user has pressed the <Enter> key, your program will search the directory for the entered room number.

2. **Searching the directory:** The directory stores for each room the name of faculty in the office. You will find a match for the room number if and only if the room number exists in the directory. Note that it is possible to not find a match in the directory.

The directory is organized as a data-structure called a linked-list. The details of linked-list are described in the appendix. The address of the first node of the directory is stored in memory location x3300 before the program starts.

3. **Printing the room number:** If your program does not find a match for a room number in the directory, your program must EXACTLY print the string "No Entry" on the screen. If your program finds a match in the directory, your program will print the string containing the faculty’s last name.
4. **Halting the machine:** The machine halts after printing all the names of faculties in the office or "No Entry".

**Input/Output Requirements**

The detailed requirements about the Inputs and Outputs of your program are described below.
You should adhere to these guidelines to receive full credit for this assignment.

**Input:**

Your program should prompt the user for the last name from the keyboard, as follows:
1. Print a string "Type a room number and press Enter: ". Note that you will get zero on the assignment if you do not print this string EXACTLY.
2. The user will input a character string from the keyboard, terminating the room number with the <Enter> key. The <Enter> key is not part of the message, but you will NEED to print it back out to the console.
3. You can assume that ONLY lower-case alphabets will be entered.
4. You may assume that the room number is typed correctly without using backspace and delete.
5. The length of the entered string is between 1 and 15 characters.

**Hint:** To continually read from the keyboard without first printing a prompt on the screen, use TRAP x20 (assembler name GETC). That is, for each key you wish to read, the LC-3 operating system must execute the TRAP x20 service routine. If you follow TRAP x20 with the instruction TRAP x21 (assembler name OUT), the character the user types will be displayed on the screen.

**Output:**

Your program should output one of two strings depending on the outcome of the directory lookup:
1. When the room number entered by the user is found in the linked list, output to the screen the name of the faculty as recorded in the directory.
2. When the room number entered by the user is not found in the linked list, output to the screen: "No Entry".

**Hint:** To output a string to the console display, use TRAP x22 (assembler name PUTS). What needs to go into R0 in order to use this TRAP instruction?

A sample of what your program will produce, when supplied with the input from users trying to look up a professor's office location, is shown below:

```
Type a last name and press Enter: 520
jiang
----- Halting the processor -----
Type a last name and press Enter: 306a
No Entry
----- Halting the processor -----
Type a last name and press Enter: 306
le
----- Halting the processor -----
```

**Note:**

1. Your program must start at location x3000.
2. The linked-list representing the directory is an input to your program. The directory is loaded in memory before your program begins to run. Your program will only search the directory, not modify it. You will lose points if you modify the directory.
3. The pointer to the first node is stored in memory location x3300 before the program execution begins. Further, assume that all the nodes are stored between memory locations x4000 and xEFFF. Make no other assumptions about the location of the nodes. Note that the pointer to the first node may be set to NULL, indicating that there are no nodes in the list.
4. Your program should NOT make any assumptions about the number of nodes in the list.
5. The <Enter> key is the line feed which is the ASCII code x0A.
6. You can assume that the directory never contains two nodes with the same last name.

**Submit your Program**

The program you are to submit is the ***.asm*** file. Save your *.asm* file, and give it the name “directory.asm”. You SHOULD write a one-page report for your program to briefly describe how to use your program.

Submit your compressed file to the website: http://10.214.208.4/

## Appendix

### The Linked-List

A linked-list consists of nodes which are linked with each other via pointers. Each node in a linked-list contains a pointer to the next node. This pointer is commonly known as the next-pointer. All nodes have a valid next-pointer except the last node. The last node is unique as it does not have a next node and its next-pointer is set to a NULL pointer (the value x0000).

Each node in a linked-list is comprised of k+1 words: one word containing the next-pointer (the pointer to the next node) and k words of data which are being stored by the node.

The directory of room numbers is implemented as a linked-list. Each node consists of three words in the following order:

1. The next-pointer.
2. A pointer to an ASCII string representing the room number of the professor's office.
3. A pointer to an ASCII string representing the professor's last name (in lower case).

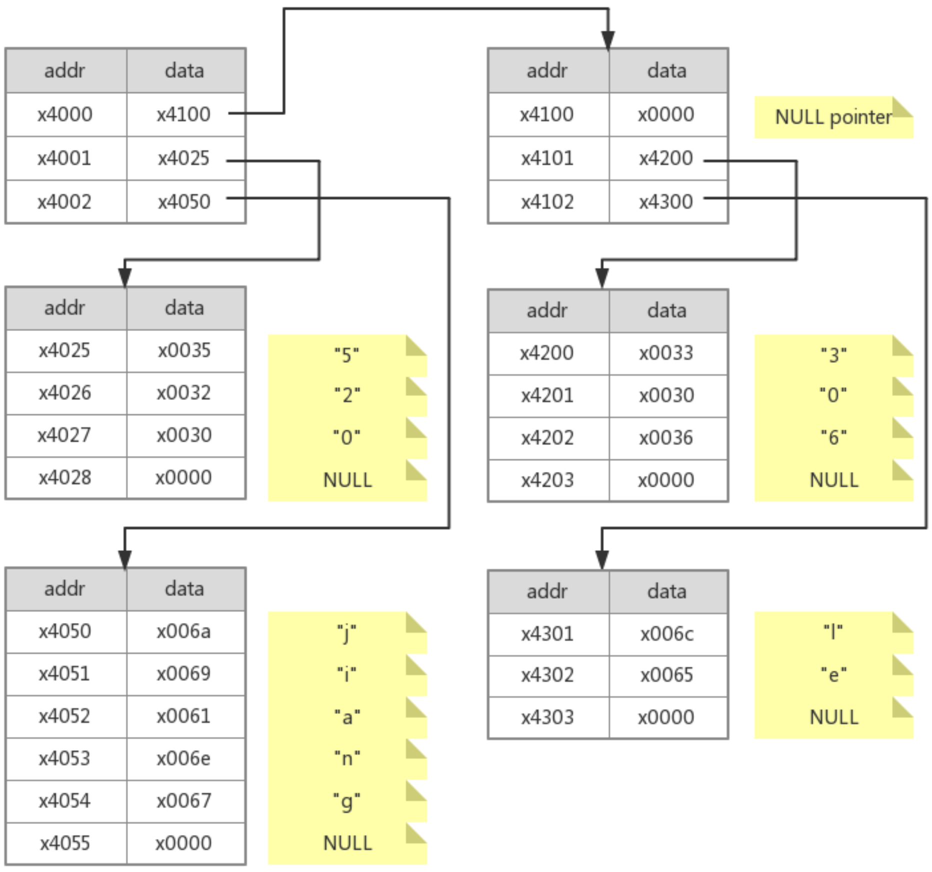
Recall that a string consists of ASCII codes stored in consecutive memory locations, one ASCII code per location. The string is null-terminated, i.e. the end of a string is signified by the NULL character which is ASCII code 0.

Below we show an example directory implemented as a linked-list. This directory contains two nodes: the room numbers of professors Jiang (room 520) and Ie (room 306).

| **Address** | **Contents** | **Description** |
|---|---|---|
| x3300 | x4000 | Pointer to the first node |
| ... | ... | ... |
| x4000 | x4100 | Next-pointer |
| x4001 | x4025 | Pointer to the first ASCII code of the office location |
| x4002 | x4050 | Pointer to the first ASCII code of the last name |
| … | … | … |
| x4025 | x0035 | ASCII code for "5" |
| x4026 | x0032 | ASCII code for "2" |
| x4027 | x0030 | ASCII code for "0" |
| x4028 | x0000 | NULL character which signifies the end of the office location |
| … | … | … |
| x4050 | x006a | ASCII code for "j" |
| x4051 | x0069 | ASCII code for "i" |
| x4052 | x0061 | ASCII code for "a" |

| | | |
|---|---|---|
| x4053 | x006e | ASCII code for "n" |
| x4054 | x0067 | ASCII code for "g" |
| x4055 | x0000 | NULL character which signifies the end of the last name |
| … | … | … |
| x4100 | x0000 | Next-pointer, x0000 means there is no next node |
| x4101 | x4200 | Pointer to the first ASCII code of the office location |
| x4102 | x4300 | Pointer to the first ASCII code of the last name |
| … | … | … |
| x4200 | x0033 | ASCII code for "3" |
| x4201 | x0030 | ASCII code for "0" |
| x4202 | x0036 | ASCII code for "6" |
| x4203 | x0000 | NULL character which signifies the end of the office location |
| … | … | … |
| x4300 | x006c | ASCII code for "l" |
| x4301 | x0065 | ASCII code for "e" |
| x4302 | x0000 | NULL character which signifies the end of the last name |

Below is a graphical representation of the linked-list.



To search the linked-list, your program will visit each node of the list and compare the room number stored at that node with the room number entered by the user. If a match is found, it will print the room number of the professor's office and terminate the search. If a match is not found, it will visit the next node and repeat the above process. It will continue to visit nodes until either a match is found or it reaches the last node (i.e. a node whose next pointer the next node is 0).
Thus, if it reaches the last node, it implies that a match was not found.