Lab2-Report

1 Algorithm

1.1 Input and store a string

- First, load one char at a time, and output to the monitor, so as to echo.
- Then test if it is `Enter`.
  - If the answer is yes, appendix `\0` to the string, indicating the input is end, and jump out of the input loop.
  - If the answer is not, store the char into reserved space pointed by the str pointer.
- Str pointer increment, and go to the next loop.

1.2 Search and Compare

- Initialize a flag, assuming not found.
- Iterate the linked-list given
- Every time access to a new node, fisrtly identify whether it is NULL.
  - If so, iteration is end, and jump out of the loop.
  - If not, go next.
- Then compare first name or last time with input string separately, and record the status.That is, both first name and last name are matched, one of them is matched, or neither is matched.
  - Compare every char from head to `\0`. If every char is matched, two strings is identical.
  - Depending on the status, output with different format.
  - Once we find the matched situation, we set the flag to identify found.
- Go to next node, loop.
- When the loop is end, check the flag. If not found, output `Notfound`.

2 Codes & Comments

```
; Input the name
            LEA     R1, NAME                    ; R1, ptr of input string
Input       GETC
            OUT                                 ; echo on the monitor
            LD      R2, Enter
            NOT     R2, R2
            ADD     R2, R2, #1
            ADD     R2, R2, R0                  ; test whether the current char is '\n'
            BRz     StopInput                   ; '\n' indicates the end of input
            STR     R0, R1, #0
            ADD     R1, R1, #1
            BR      Input                       ; read next char
StopInput   AND     R1, R1, #0                  ; add '\0' to the end of string
; Search in the Linked List
```

```
            LDI     R1, Node0                   ; R1, ptr of Linkedlist
LOOP        BRz     StopSearch                  ; if R1 == x0000, R1 points to NULL
            LDR     R2, R1, #2                  ; R2, ptr of fisrtname or lastname
            LEA     R4, NAME                    ; R4, ptr of input
            JSR     Compare                     ; identify whether firstname is equal
to input
            ADD     R0, R6, #0                  ; R0 = 1 if FIRST name match, else = 0.

            LDR     R2, R1, #3
            JSR     Compare                     ; R6 = 1 if LAST name match, else = 0
            AND     R2, R0, R6
            BRp     BothMatch                   ; fisrtname is identical to lastname
            ADD     R0, R0, #0
            BRp     OneMatch
            ADD     R6, R6, #0
            BRp     OneMatch
            BRnzp   NextLoop
;
OneMatch    AND     R0, R0, #0
            ST      R0, FLAG                    ; set FLAG = 0 to indicates Found
; Output with format
            LDR     R0, R1, #2
            PUTS
            LD      R0, Space
            OUT
BothMatch   AND     R0, R0, #0
            ST      R0, FLAG                    ; set FLAG = 0 to indicates Found
            LDR     R0, R1, #3
            PUTS
            LD      R0, Space
            OUT
            LDR     R0, R1, #1
            PUTS
            LD      R0, Enter
            OUT
;
NextLoop    LDR     R1, R1, #0                  ; Load the next Node to R1
            BR      LOOP
; Subroutine to find whether string A is equal to B
Compare     ST      R4, Save4
            AND     R6, R6, #0
            ADD     R6, R6, #1
AGAIN       LDR     R3, R2, #0                  ; test whether name string is end
            BRz     Test
            LDR     R5, R4, #0                  ; test whether input string is end
            BRz     NotMatch
            NOT     R5, R5
            ADD     R5, R5, #1                  ; cmp name with input
            ADD     R5, R5, R3
```

```
            BRnp    NotMatch
            ADD     R2, R2, #1                  ; ptr++
            ADD     R4, R4, #1
            BRnzp   AGAIN
Test        LDR     R5, R4, #0
            BRz     EndCmp
NotMatch    AND     R6, R6, #0
EndCmp      LD      R4, Save4
            RET
```

3 TA's Questions

3.1 How to iterate the linked-list?

locate Node0

```
            LDI     R1, Node0
Node0       .FILL x4000
```

the end condition of the loop: `LOOP        BRz     StopSearch`

Load the next Node,and loop

```
            LDR     R1, R1, #0
            BR      LOOP
```

3.2 How to compare two strings?

R2 is the ptr to name, R4 is the ptr to input.

R3 <- mem[R2], R5 <- mem[R4]

First test if the strings are ended.If one of the two strings is ended, turn to NotMatch. If both are at the end, jump out of the loop.

If neither are ended, calculate whether R3 is equal to R5.If not, turn to NotMatch.If so, increase the ptr(i.e. R2, R4), and go to the next comparison.