MARIE Examples

Sebastien Bah

March 2019

Setup

Setup

Things you will need:

- Documentation

Setup

Things you will need:

- Documentation
- Simulator

Multiplication

Multiplication

How do you multiply? Let's say you want to do $x = 5x + y$?

Write a quick function?

Code

```
// Fill out the code to have the function do x = 5x + y

// Code here


// Variables
X,      DEC 6
Y,DEC 8
```

Candidate

```
// Fill out the code to have the function do x = 5x + y
Load X
        Add X
        Add X
        Add X
        Add X
        Add Y
        Store X
// Variables
X,      DEC 6
Y,DEC 8
```

Jumps

Jumps

```
// Jumps
// What is the final values of X and Y AND what is the outp

A,Clear
B,Input
C,Jump K
D,Jump F
E,Jump H
F,Store X
G,Jump L
H,Store Y
I,Output
J,Halt
K,Jump D
L,HEX 9008

// Variables
X,  DEC 0
```

Subroutines

Subroutines

Recall:

- **JnS** –> *Stores* the value of PC at address X and *Jumps* to X+1

Subroutines

Recall:

- **JnS** –> *Stores* the value of PC at address X and *Jumps* to X+1
- **JumpI** –> Uses the *value at the memory address X* as the addres to *Jump* to

Code

```
// Subroutines Aka Functions

Start,Jns Function
Load X
                Output
Jns Function
                Load X
                Output
                Jns Function
                Output
                Halt

// Just a memory location where we are going to
// store the address of the address of the instruction
// we skipped to get to this point
Function,HEX 0
Load X
                Add  X
```

Skip conditions

Skip conditions

Recall:

- Skipcond 000 -> Skips next line if $AC < 0$

Skip conditions

Recall:

- Skipcond 000 -> Skips next line if $AC < 0$
- Skipcond 400 -> Skips next line if $AC = 0$

Skip conditions

Recall:

- Skipcond 000 -> Skips next line if $AC < 0$
- Skipcond 400 -> Skips next line if $AC = 0$
- Skipcond 800 -> Skips next line if $AC > 0$

Code

```
// Decoding memory content
A,LOAD Z
Output
        Subt X
        Skipcond 800
        Jump B
        Jump End
B,Load Z
Add ONE
        Store Z
        Jump A
End,  Output
Halt

// Variables
ONE,DEC 1
X,  DEC 5
Y,DEC 3
```

Loops

Loops

Let's do the multiplication function $x = Nx + y$ but with loops.

(Tools needed: Jump and Skipcond)

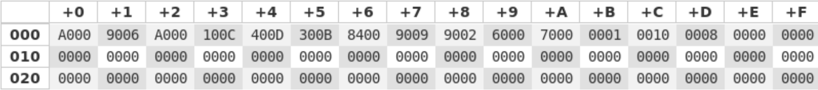
Code

```
// Fill out the code to have the function do x = Nx + y
Input
        Store N

Loop,Load N
        Skipcond 800
        Jump End
        Subt ONE
        Store N

XPart,Load TEMP
Add X
        Store TEMP
        Jump Loop

End,Load TEMP
Add Y
        Store X
```

Decoding Memory Content

Decoding Memory Content

Given that the starting value of $PC = 000$, what does this program do?



	+0	+1	+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
000	A000	9006	A000	100C	400D	300B	8400	9009	9002	6000	7000	0001	0010	0008	0000	0000
010	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
020	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Figure 1: Decoding Memory Content

Code

```
// Decoding memroy content
Clear
        Jump C
B,Clear
Load X
        Subt Y
        Add ONE
C,Skipcond 400
Jump End
        Jump B
End,  Output
Halt

// Variables
ONE,DEC 1
X,  DEC 16
Y,DEC 8
```

Finding the unknown

Finding the unknown

What is the value of UNKNOWN for the function to be $A = 2Y - Z$

Code

```
// What is the value of UNKNOWN for the
//      function to be  A = 2Y - Z
                Input
                Store UNKNOWN

Start,          Load UNKNOWN
Subt ONE
                Store UNKNOWN
                Skipcond 800
                Jump End

Ops,Load Y
Add ONE
Subt Z
                Store A
                Jump Start

// End of your function
```

Array

Array

```
// Stores the amount of numbers that you want (determined b
// And prints them out at the end

// EX: first input 8 (dec)
// Subsequent: [67, 79, 77, 80, 32, 50, 50, 56] (all dec)
// Put your output mode to unicode
ORG 010
Start,      Input
Store LenArray
Loop,Load LenArray
Subt Index
Skipcond 800
                Jump DisplayAll

// Gets the next number to store
GetInput,  Load StartIndex
Add Index
                Store NextPos
```