IN0013 week 3 worksheet

1. Complete the following table with the correct mnemonics and machine code. Maybe write somewhere what it does too!

Instruction	Mnemonic	MachineCode
Load	LDA	5xx
Store	STA	3xx
Add	ADD	1xx
Subtract	SUB	2xx
Input	INP	901
Output	OUT	902
End	HLT	000
Branch if zero	BRZ	7xx
Branch if zero or positive	BRP	8xx
Branch always	BRA	6xx
Data storage	DAT	

2. Read the von Neumann architecture and Harvard architecture. explain a bit what their differences are. Include a picture of what they both look like.

The Von Neumann and Harvard architectures are two fundamental designs used in the construction of computer systems. They differ primarily in how they store and access memory, particularly regarding the treatment of data and instructions.

Harvard architecture

Advantages

- Faster processing
- Improved security
- Efficient use of resources

Disadvantages

- Complexity
- Higher cost
- Less flexibility competitors

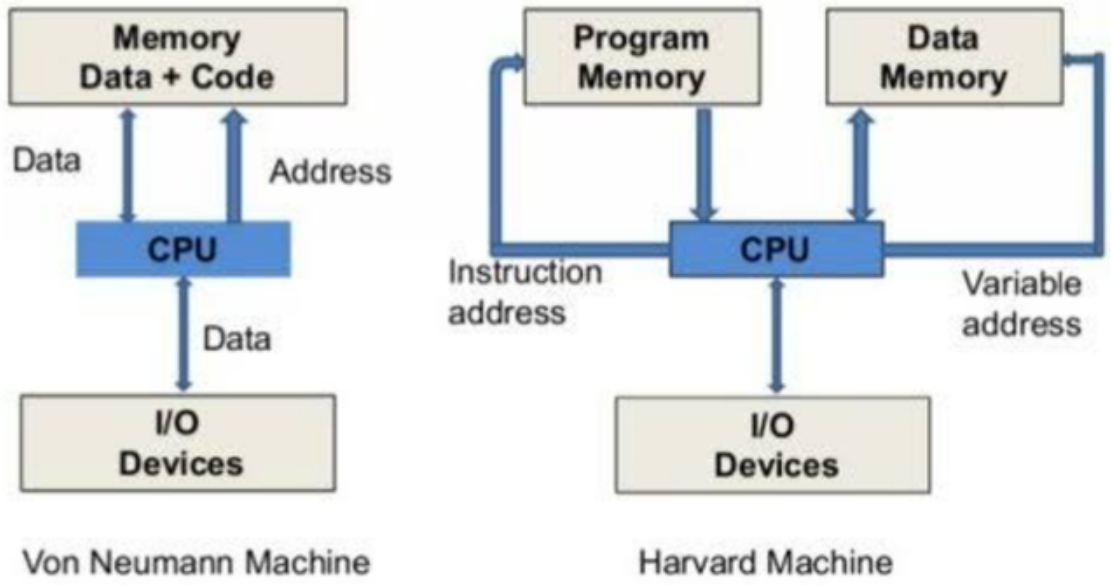
Von Neumann architecture

Advantages

- Simplicity
- Cost-effective
- Flexibility

Disadvantages

- Bottleneck issues
- Memory consumption



3. Read the following assembly code and explain in 1 sentence what it does. Use the column comments to write what each line of code does.

Code	Comments (write what the code does below)
Code 1: summary	Takes two inputs and output them in reverse

Code	Comments (write what the code does below)
`INP`	Read the first input into the accumulator.
`STA 99`	Stores the first input in mailbox 99.
`INP`	Read the second input into the accumulator.
`OUT`	Output the second input which is already in the accumulator.
`LDA 99`	Loads the first input from mailbox 99 into the accumulator.
`OUT`	Output the first input.
`HLT`	Halt the program.
`99 DAT`	Reserve mailbox 99 for the first input.
Code 2: summary	Taking 1 input and subtracting it by 1
`INP`	Read the input into the accumulator
`SUB 99`	Subtract 1 from the input in the accumulator
`OUT`	Output the result
`HLT`	Halt the program
`99 DAT 1`	Reserve mailbox 99 with the value 1 for subtraction
Code 3: Summary	Taking 2 inputs and swapping their values in output
`INP`	Read the first input into the accumulator
`STA 98`	Store the first input
`INP`	Read the second
`STA 99`	Store the second input in mailbox 99
`LDA 98`	Load the first input from mailbox 98 into the accumulator
`OUT`	Output the first input which acts as a second input
`LDA 99`	Load the second input from mailbox 99 into the accumulator
`OUT`	Output the second input which acts as the first input
`HLT`	Halt the program.
`98 DAT`	Reserve mailbox 98 for the first input.
`99 DAT`	Reserve mailbox 99 for the second input.

4. Write an assembly code for the following instructions: screenshot the code and output of your LMC
    a. Take 2 inputs x and y and output x+y

    b. Take 2 inputs x and y and output x – y

c. Take 3 inputs x y, and z and output z, y , x (reverse order).

a. Take 2 inputs x and y and output x+y

Code	Comments
INP	Read the first input (x) into the accumulator.
STA X	Store the value of x in mailbox labeled X.
INP	Read the second input (y) into the accumulator.
ADD X	Add the value of x to y (in the accumulator).
OUT	Output the result (x+y).
HLT	Halt the program.
X DAT	Reserve a mailbox for the first input x.

b. Take 2 inputs x and y and output x – y

Code	Comments
INP	Read the first input (x) into the accumulator.
STA X	Store the value of x in mailbox labeled X.
INP	Read the second input (y) into the accumulator.
SUB X	Subtract y (in the accumulator) from x.
OUT	Output the result (x-y).
HLT	Halt the program.
X DAT	Reserve a mailbox for the first input x.

c. Take 3 inputs x y, and z and output z, y , x (reverse order).

Code	Comments
INP	Read the first input (x) into the accumulator.
STA X	Store the value of x in mailbox labeled X.
INP	Read the second input (y) into the accumulator.
STA Y	Store the value of y in mailbox labeled Y.
INP	Read the third input (z) into the accumulator.
OUT	Output z (already in the accumulator).
LDA Y	Load the value of y into the accumulator.
OUT	Output y.
LDA X	Load the value of x into the accumulator.
OUT	Output x.
HLT	Halt the program.
X DAT	Reserve a mailbox for x.
Y DAT	Reserve a mailbox for y.

5. (branching question) read the following code and explain what it does.

Code	Comments
Summary of code	
`INP`	
`LOOP BRZ END`	
`OUT`	
`SUB ONE`	
`BRA LOOP`	
`END HLT`	
`ONE DAT 1`	

Code	Comments
Summary of code	**It takes an input x and print x-1 every time until it reaches 0. It does not print the 0.**
`INP`	Read the input into the accumulator.
`LOOP BRZ END`	Label loop then If the accumulator is 0, jump to END.
`OUT`	Output the current value in the accumulator.
`SUB ONE`	Subtract 1 from the accumulator.
`BRA LOOP`	Unconditionally jump back to LOOP.
`END HLT`	Label for the end of the program. Then Halt the program.
`ONE DAT 1`	Store the value 1 in memory for subtraction.