6.4
The criteria are the stored program concept, so the program is stored in memory. Each memory is addressed with a numeric. Memory is referenced to the location number despite data.

1. take the first number and store it
2. Counter is incremented
3. data does not affect the instruction.

6.5

Mailbox	numeric code	mnemonic code
00	901	Input 1st number
01	300	Store what's in mailbox
02	901	Input second number
03	100	Add numbers
04	902	Output
05	000	Halt

If we executed this a second time the number would not be found. This is because of stored procedures being destroyed after Halting.

6.6

Instruction	Mailbox	Description
Input Store	97	Give 1st Number
Input Store	98	Give 2nd Number
Input Store	99	Give 3rd number
Subtract	98	Compare third and Second
Branch if Negative	15	If Third ?= Second
Load	99	load third
Subtract	97	Compare third with First
Branch	13	Go to the output code
Load	99	First is the maximum
Branch	21	Go to Output Code
Load	98	Load Second
Subtract	97	Compare Second and First
Branch if negative	21	if second ?= First
Load	98	Second is the maximum
Branch	21	Go to output Code
Load	97	First is the Maximum
Output		Print Maximum
Halt		Stop

The three numbers are stored in 97, 98, and 99. Then compare the numbers with subtraction - if it's greater compare it with the first number. If it's greater than the first then the number at 99 is greatest.

I.e. If one number is greater than another such that y - x yields a negative number, so x must be greater. If y - x yields a positive number y must be greater.

15-21
6.15

```
if (condition)
   execute order1
else
   execute order2
end if
Stop
```

```
LDA
IF: test condition (x == y)
SUB Y
JMP ELSE
LDX
ADD Y
STO X
ENDIF
ELSE
LDA Y
SUB X
STO Y
ENDIF
HALT
```

6.16

XY LDA X	Body
ADD Y	Add Y
STO X	store X
LDA X	Perform operations
SUB Y	Subtract Y
SKNZ	Skip out of loop if X != Y
JMP XY	Jump to Loop body
LDA X	End of Loop
OUT	Output

6.17
We could swap numbers in storage locations by using something similar to bubble sort

6.18. There are 10 opcode values so We would need at leas 4bits for that portion.

This would leave 12 bits for mailboxes
Some could accomadate 10^{12} mailboxes or 4096 mailbo

We could accomadate 2048 range in twos Complement because 1 bit would be used for negative numbers.

6.19. Well it may be easier to remember And if the coder forgets to put a COB at the end it ~~will~~ could stop the execution by reaching an empty mailbox.

	Mailbox	Instruction	OpCode	Mnemonic (x+y)
6.20	41	IN	901	INPUT x
	42	STO	399	STORE
	43	IN	901	INPUT y
	44	ADD	199	ADDITION (x+y)
	45	STO	399	STORE
	46	IN	901	INPUT (z)
	47	SUB	299	SUBTRACT z-(x+y)
	48	BRP	841	BRANCH IF POSITIVE if (z-(x+y)≥0)
	49	BRZ	741	BRANCH IF ZERO if (z-(x+y)=0)
	50	OUT	902	OUTPUT
	51	COB	000	COFFEE BREAK Stop
6.21	75	IN	901	INPUT x
	76	STO	399	STORE
	77	IN	901	INPUT y
	78	ADD	199	ADDITION x+y
	79	STO	399	STORE
	80	IN	901	INPUT z
	81	SUB	299	SUBTRACT z-(x+y)
	82	BRP	875	BRANCH IF POSITIVE if (z-(x+y)≥0)
	83	OUT	902	OUTPUT
	84	COB	000	COFFEE BREAK STOP