**Instructions:**

1. **lda [address]** , load A from address
2. **add [address]** , add A with address, store in A
3. **sub [address]** , sub A with address, store in A
4. **sta [address]** , load data from A to address
5. **jmp [address]** , load prog counter with ins register data
6. **jmpc [address]** , jump if carry flag
7. **jmpz [address]** , jump if zero flag
8. **jmpn [address]** , jump if negative flag
9. **and [address]** , and A with address, store in A
10. **or [address]** , or A with address, store in A
11. **xor [address]** , xor A with address, store in A
12. **multl [address]** , multiply A with address, store lower 8 bits in A
13. **multh [address]** , multiply A with address, store high 8 bits in A
14. **ldi [number]** , load A with a byte
15. **add# [number]** , add A with a byte
16. **sub# [number]** , sub A with a byte
17. **and# [number]** , and A with a byte
18. **or# [number]** , or A with a byte
19. **xor# [number]** , xor A with a byte
20. **multl# [number]** , mult A with a byte, store lower 8 bits in A
21. **multh# [number]** , mult A with a byte, store high 8 bits in A
22. **push# [number]** , push a byte onto the stack
23. **noop** , does nothing
24. **out** , load data from A to OUT
25. **inc** , add 1 to the A register
26. **dec** , sub 1 to the A register
27. **rshift** , shift A register right
28. **lshift** , shift A register left
29. **take** , move B register content to A register
30. **pusha** , load data from A to top of stack
31. **popa** , load data from top of stack to A
32. **move** , move A register content to B register