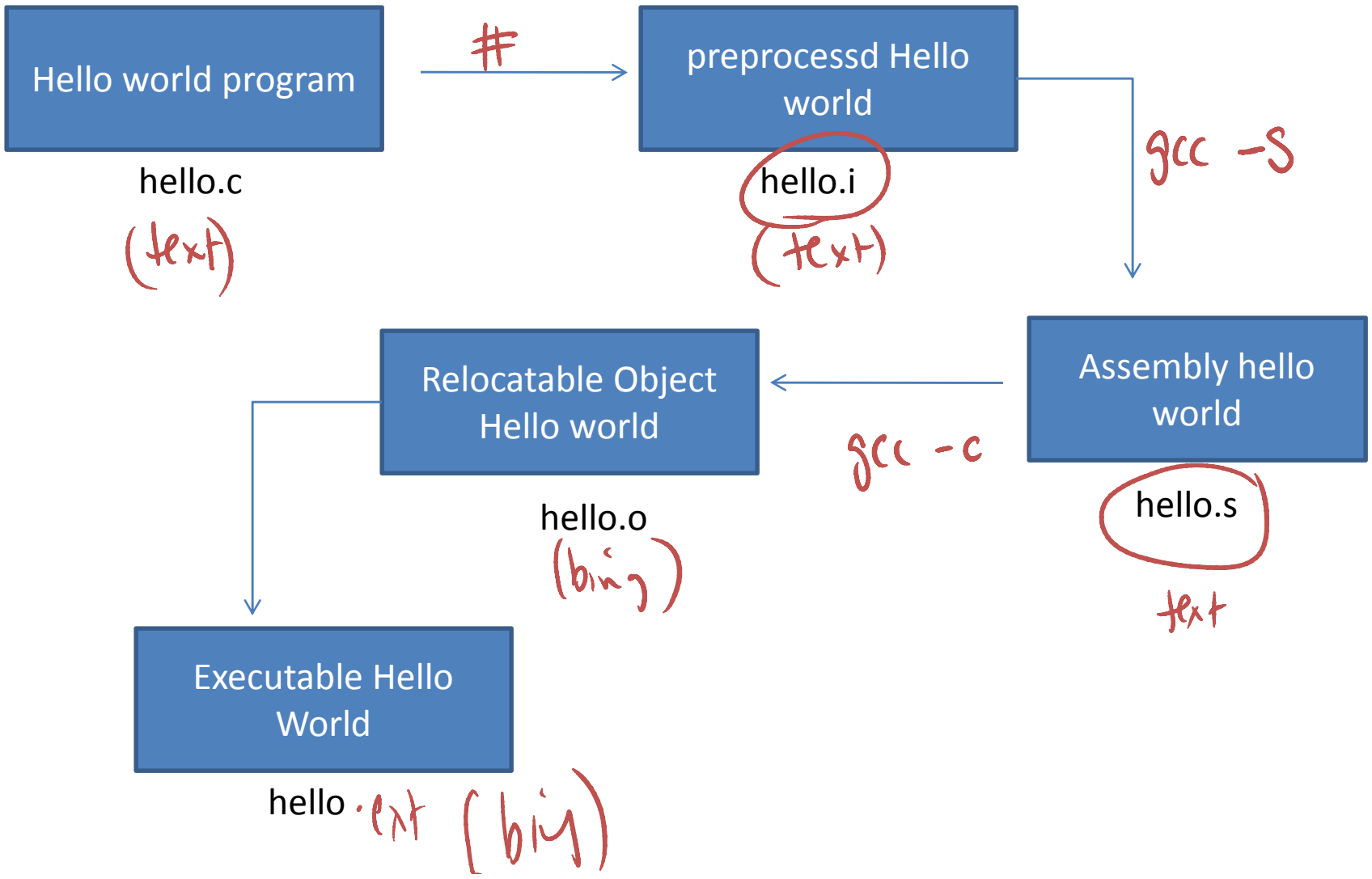
Assembler Fundamentals

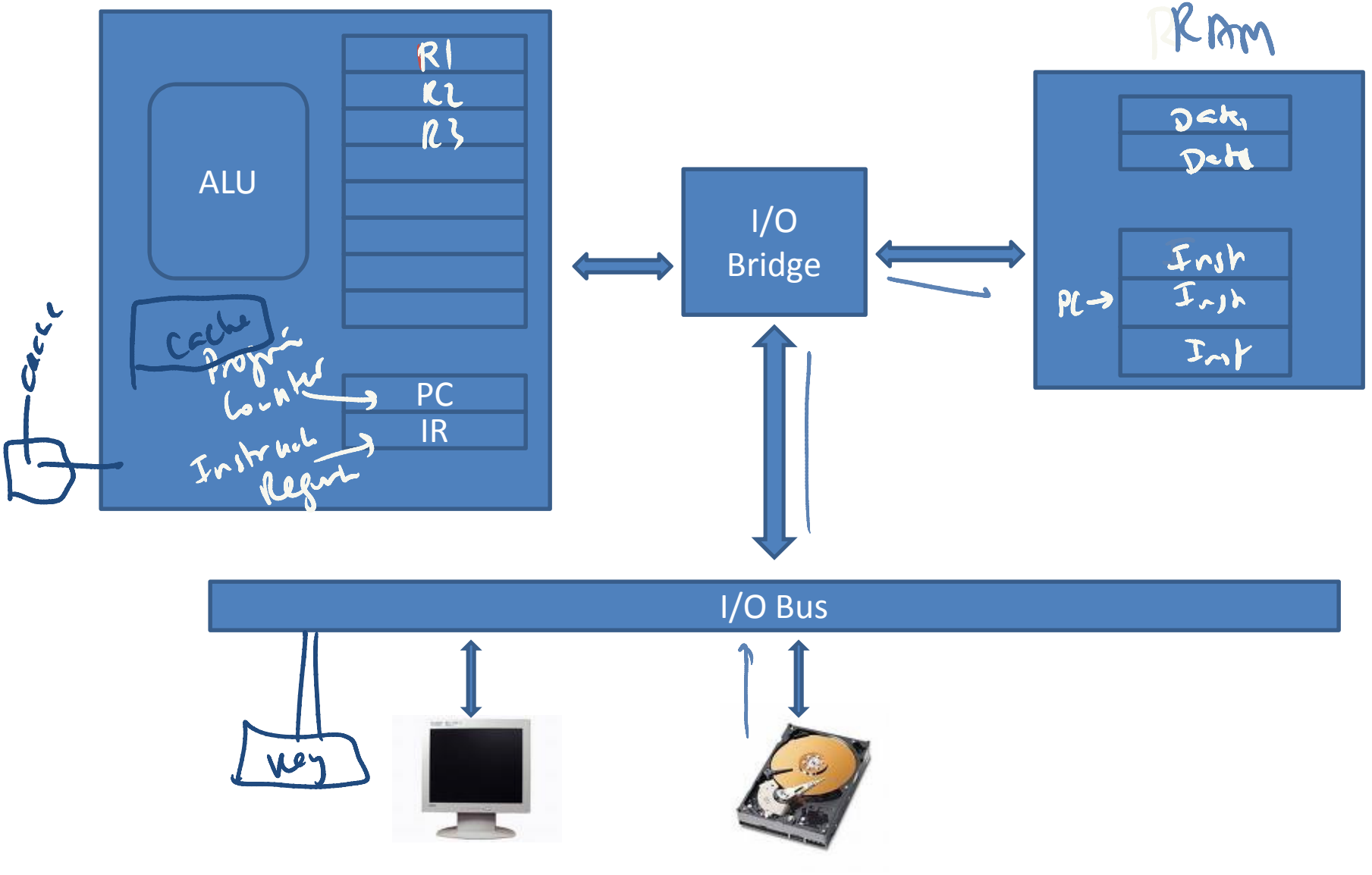
15-123

Systems Skills in C and Unix

Programs are Translated



How programs get executed

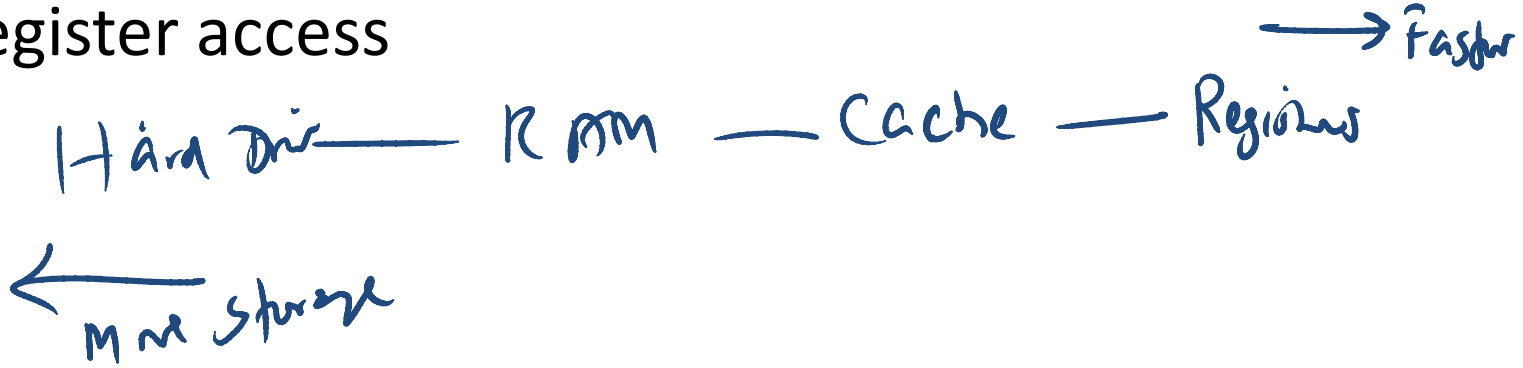


Cost of computing

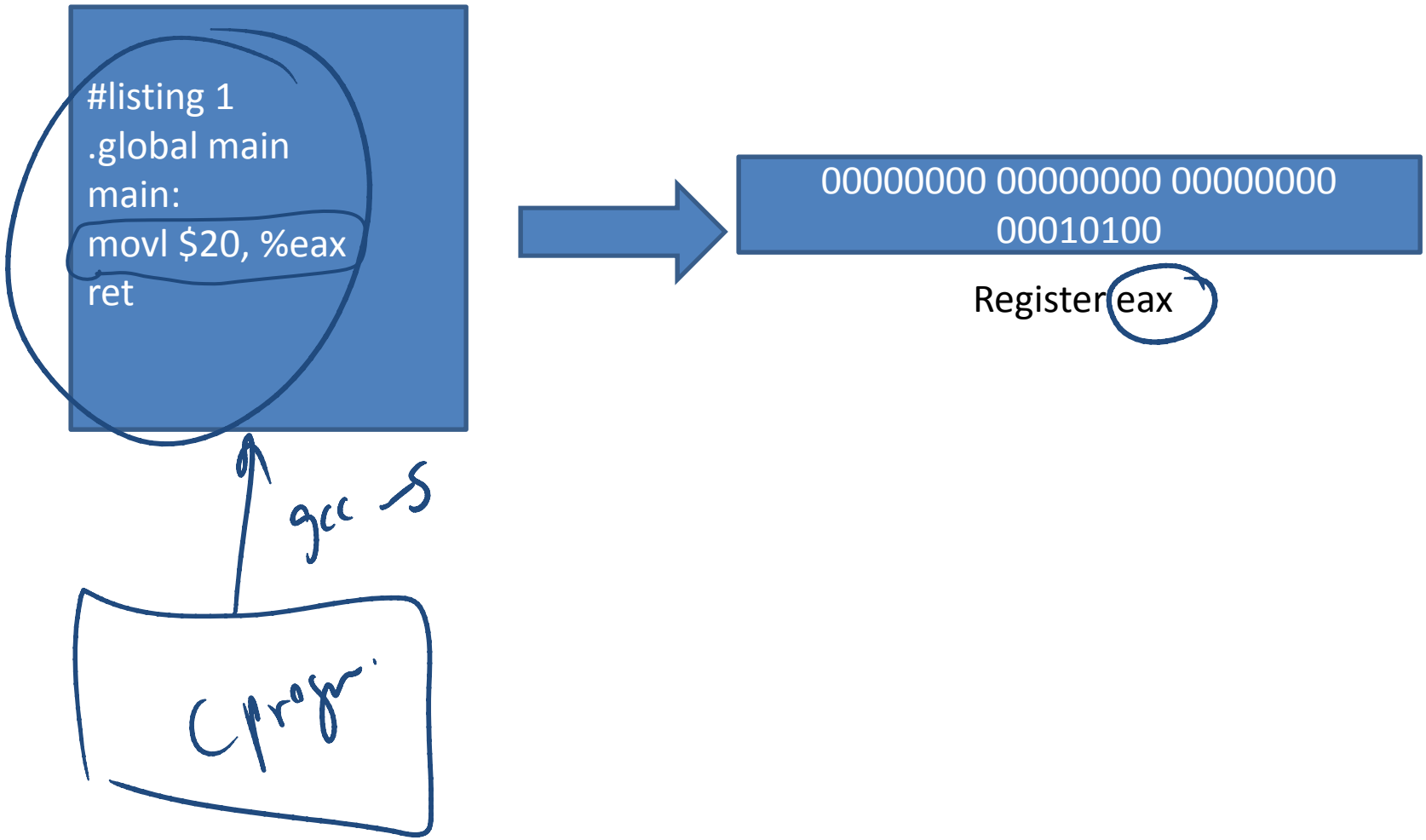
- Moving data is expensive
- Large storage devices are slower than smaller storage devices (hard drive vs RAM)
  - Capacity: RAM/hard drive = 1/100
  - Access : RAM/Hard drive = 1/10,000,000
- RAM versus Registers
  - Capacity: Register/RAM = 32 bits/2 GB
  - Access: Register :RAM access = 100:1

Speeding up with Cache

- Cache memories
  - Smaller faster storage devices
  - Stores data that the processor is likely to need in the near future
  - Cache memory is directly connected through bus interface
  - Goal is to make cache memory access as fast as register access



C to assembly



Instruction set architecture

- provides a perspective of the processor from assembly language or machine language programmer's point of view.
- ISA describes the instructions that processor understands, including register set and how the memory is organized.
- A real world processor ISA would include few additional items such as data types; interrupt handlers, exception handling etc. ISA is part of the computer architecture specific to a particular hardware.

Registers

- special purpose memory locations
- Most assembly instructions directly operate on registers
  - loading values into registers from memory
  - performing operations on them and storing
- The registers are named like
  - eax, ebx, ecx
  - ebp and esp - for manipulating the base pointer and stack pointer
- The size of a register (say 32-bit) and number of registers (say 8) depends on particular computer architecture.
- A typical instruction in assembly
  - **movl $10, %eax**

A Hypothetical Machine

Register	Number	Notes
Z	000	Constant: Always zero (0)
A	001	
B	010	
C	011	
D	100	
E	101	
F	110	
G	111	
PC		Program Counter. 24 bits wide. Not addressable
IR		Instruction Register. 32 bits wide. Not addressable.

Question: How many addressable units are in our memory model?
Answer based on PC

$RAM = 2^{24}$ bytes

Instructions? $= \frac{2^{24}}{4}$

Basic Instructions

- The basic instructions for a computer are
  - *branch instructions*
    - *jmp*
  - *I/O instructions*
    - *Load and save*
  - *Arithmetic instructions*
    - *add, mul*
  - *Device instructions*
    - *Read, write*
  - *comparison instructions*
    - *If (x > y)*

Instructions

CONTROL INSTRUCTIONS

Instruction	Op	Address	Function
HLT	0000	XXXX XXXX XXXX XXXX XXXX XXXX XXXX	Stop simulation
JMP	0001	0000 AAAA AAAA AAAA AAAA AAAA AAAA	Jump (line number)
CJMP	0010	0000 AAAA AAAA AAAA AAAA AAAA AAAA	Jump if true
OJMP	0011	0000 AAAA AAAA AAAA AAAA AAAA AAAA	Jump if overflow

LOAD-STORE INSTRUCTIONS

Instruction	Op	Register	Value	Function
LOAD	0100	0RRR	AAAA AAAA AAAA AAAA AAAA AAAA	Load (hex address)
STORE	0101	0RRR	AAAA AAAA AAAA AAAA AAAA AAAA	Store (hex address)
LOADI	0110	0RRR	0000 0000 IIII IIII IIII IIII	Load Immediate
NOP	0111	0000	0000 0000 0000 0000 0000 0000	no operation

MATH INSTRUCTIONS

Instruction	Op	Reg0	Reg1	Reg2	Function
ADD	1000	0RRR	0RRR	0RRR 0000 0000 0000 0000	Reg0 = (Reg1 + Reg2)
SUB	1001	0RRR	0RRR	0RRR 0000 0000 0000 0000	Reg0 = (Reg1 - Reg2)

DEVICE I/O

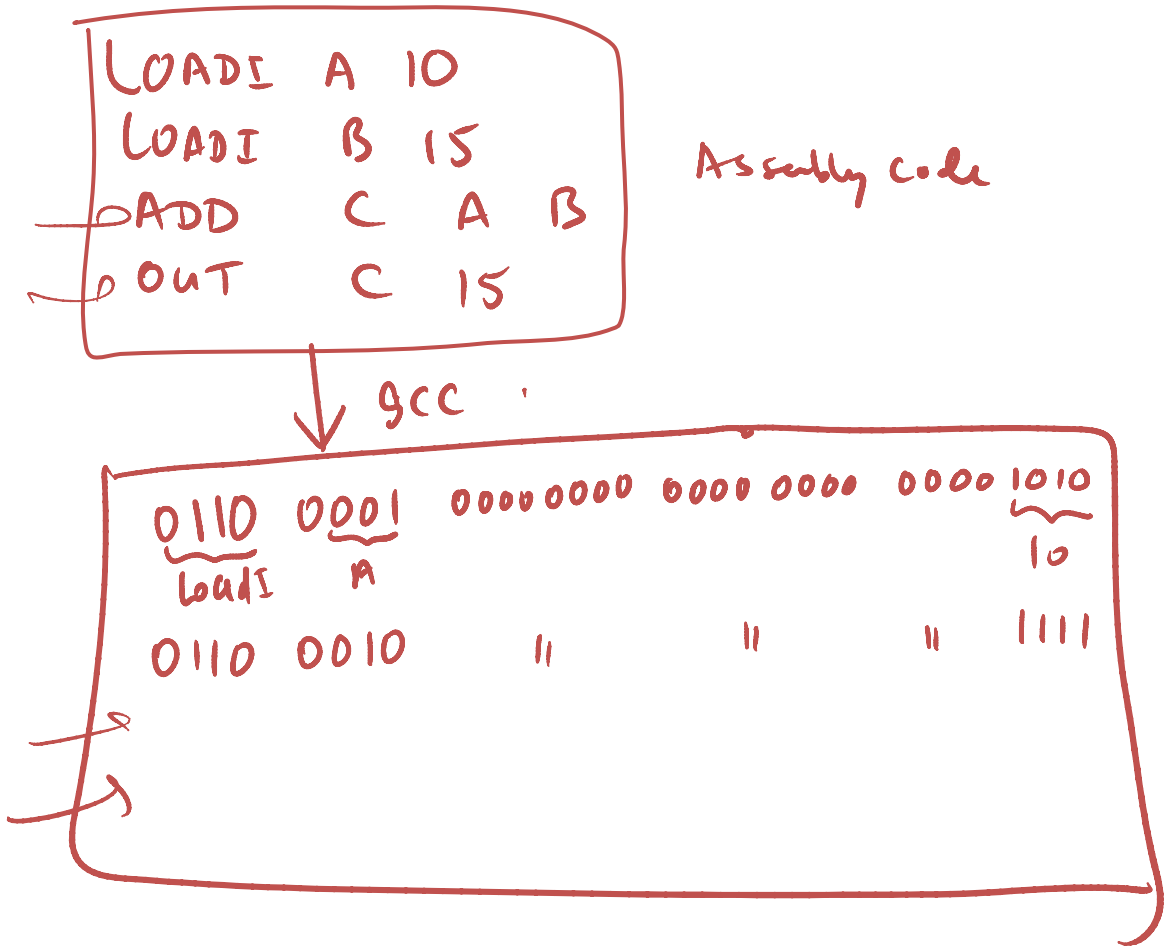
Instruction	-Op-	Reg0	0000 0000 0000 0000	Port	Function
IN	1010	0RRR	0000 0000 0000 0000	PPPP PPPP	Read Port into Reg0
OUT	1011	0RRR	0000 0000 0000 0000	PPPP PPPP	Write Reg0 out to Port

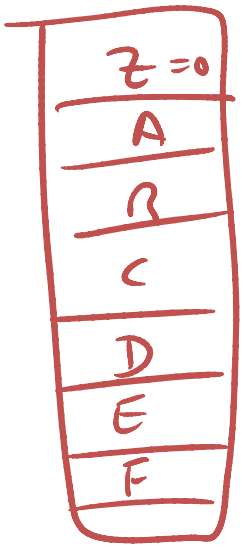
COMPARISON

Instruction	-Op-	Reg0	Reg1	Function
EQU	1100	0RRR	0RRR 0000 0000 0000 0000 0000	Cflg = (Reg0 == Reg1)
LT	1101	0RRR	0RRR 0000 0000 0000 0000 0000	Cflg = (Reg0 < Reg1)
LTE	1110	0RRR	0RRR 0000 0000 0000 0000 0000	Cflg = (Reg0 <= Reg1)
NOT	1111	0000	0000 0000 0000 0000 0000 0000	Cflg = (!Cflg)

Exercise 1

- Write a program to add the numbers 10 and 15 and output to port #15 (output port). Then convert to machine code.

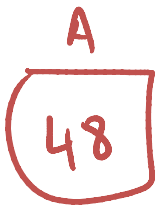




Exercise 2

- Write a program that reads a single digit integer from keyboard and output.

```
1  IN   A  0        <- port
2  LOADI B 48
3  SUB  C  A  B
4  LT   C  Z
5  CJMP 10
6  LOADI D 10
7.  LT  C  D
8.   NOT
9.   CJMP 10
10.  OUT A 15
11.  HLT
```



'0' = 48

'1' = 49

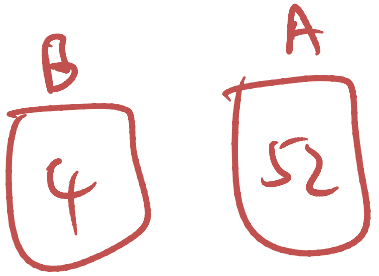
Exercise 3

- Write a program that reads a single digit integer from keyboard and output the number if the number is greater or equal to 5.

Exercise 4

- Write a program that reads a single digit integer from keyboard and output all numbers between 1 and number

```
1.   IN A 0
1.1.  LOADI D 1
2.   LOADI B 48
3.   SUB B A B
4.   #check for valid input
5.   LOADI C 1
6.   LTE C B
7.   NOT
8.   CJMP {15}
9.   ADD D D B
10.  OUT D 15
11.  ADD C C D
12.  JMP {6}
15.  HLT
```



1
2
3
4

Coding Examples