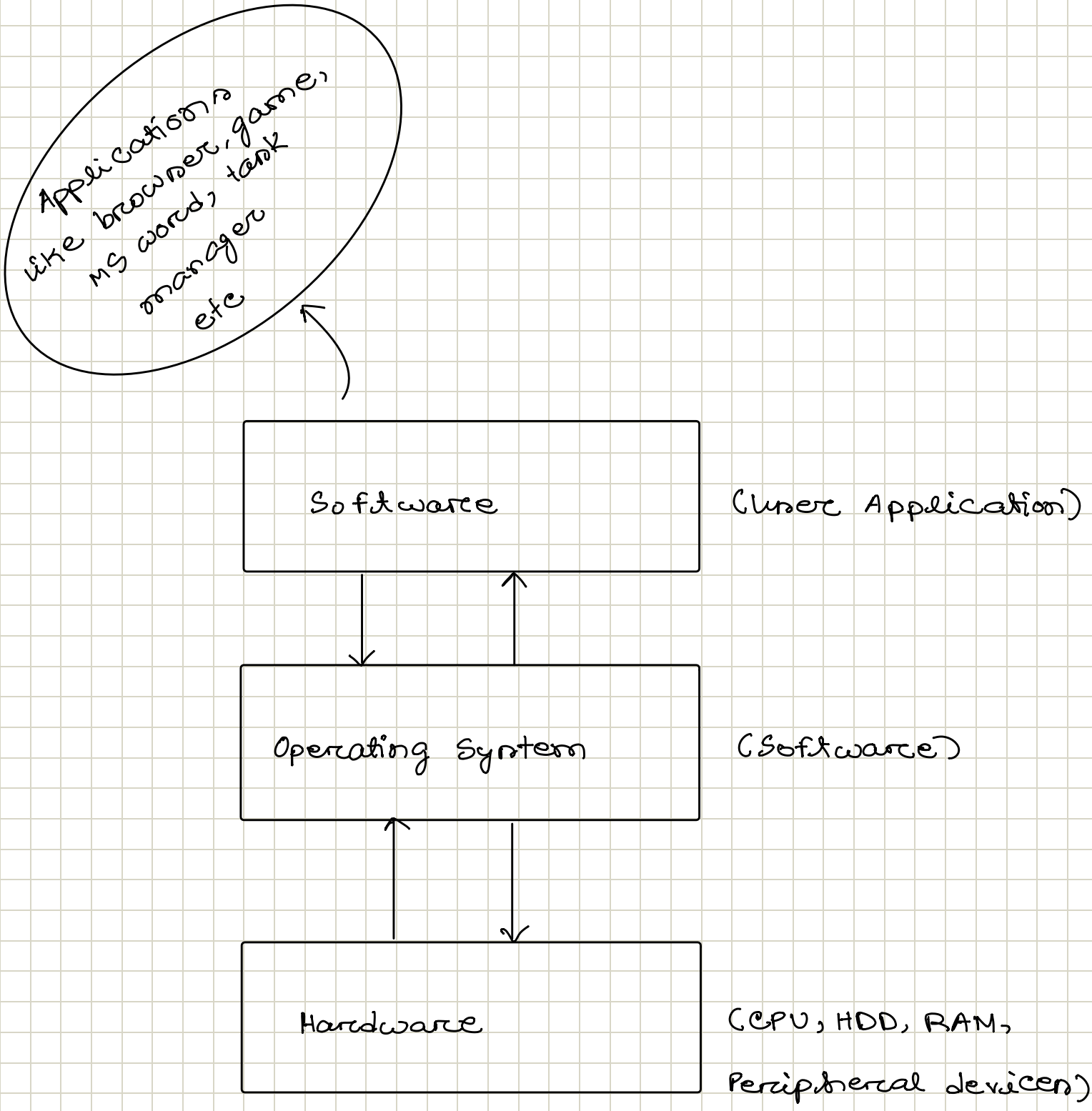
Mid Syllabus

(1) Basics of OS
(2) Process
(3) Scheduling
(4) Inter Process communication (IPC)



Responsibilities of OS:

(1) Resource Management
(2) Ensuring Security
(3) Managing Multi-tasking

} (1) Virtualization of CPU
(2) Virtualization of Memory

Virtualization of CPU

Time sharing of CPU

0.1s = 100ms
= 10^5 μsec

(1) Process: Process State, Process Creation (fork, wait, exec)
(2) Scheduling: Algorithm & Simulation
(3) Inter Process Communication (IPC): Lock Implementation & threading

Virtualization of Memory

Space sharing of Memory components

Virtual Memory
Memory Mapping

SPL Basics for OS

- if $C \neq 0$ হয় তাহলে number থাকলে → true
- if $C = 0$ হয় তাহলে number থাকলে → false

- if (a == 2 || b == 3)

এটা true হলে পরেরটা check করে না
এটা false হলে পরেরটা check করে

- if (a == 2 && b == 3)

এটা false হলে পরেরটা check করে না
এটা true হলে পরেরটা check করে

```
int a() {
  printf("OS");
  return 1;
}

int b() {
  printf("SPL");
  return 0;
}
```

①

```
int main() {
  if (a()) {
    printf("true");
  } else {
    printf("false");
  }
```

Output

OS

true

(2)

```
int main() {
  if ( b() ) {
    printf("tree");
  } else {
    printf("false");
  }
```

Output

SPL

false

(3)

```
int main() {
  if (a() || b()) {
    printf("tree");
  } else {
    printf("false");
  }
```

Output

OS

tree

(4)

```
int main() {
  if (a() && b()) {
    printf("tree");
  } else {
    printf("false");
  }
```

Output

OS

SPL

false

Process

Program : Any application that is saved in our device, not running

Process : A running program

Memory = RAM

Registers

Program counter (PC)

Instruction Pointer (IP)

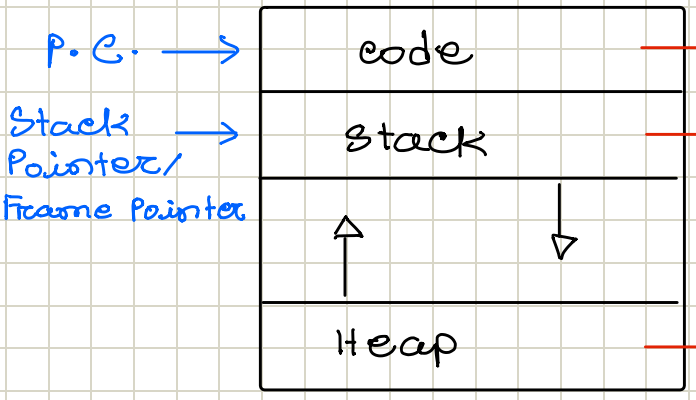
data = variables

```
int main() {
    int a=5;
    printf("%d", a);
}
```

executing

P.C. (Always points to next instruction)

Virtual Memory of a process



Interface : provides the functionalities for OS

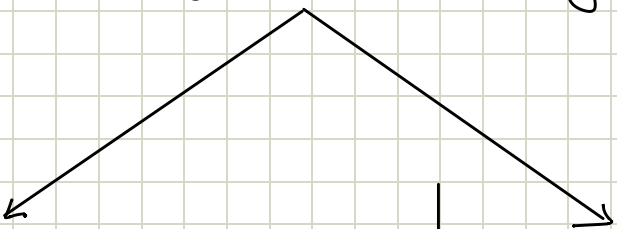
(1) create

(2) Destroy

(3) Wait

(4) Status

Program loading in Memory (RAM)



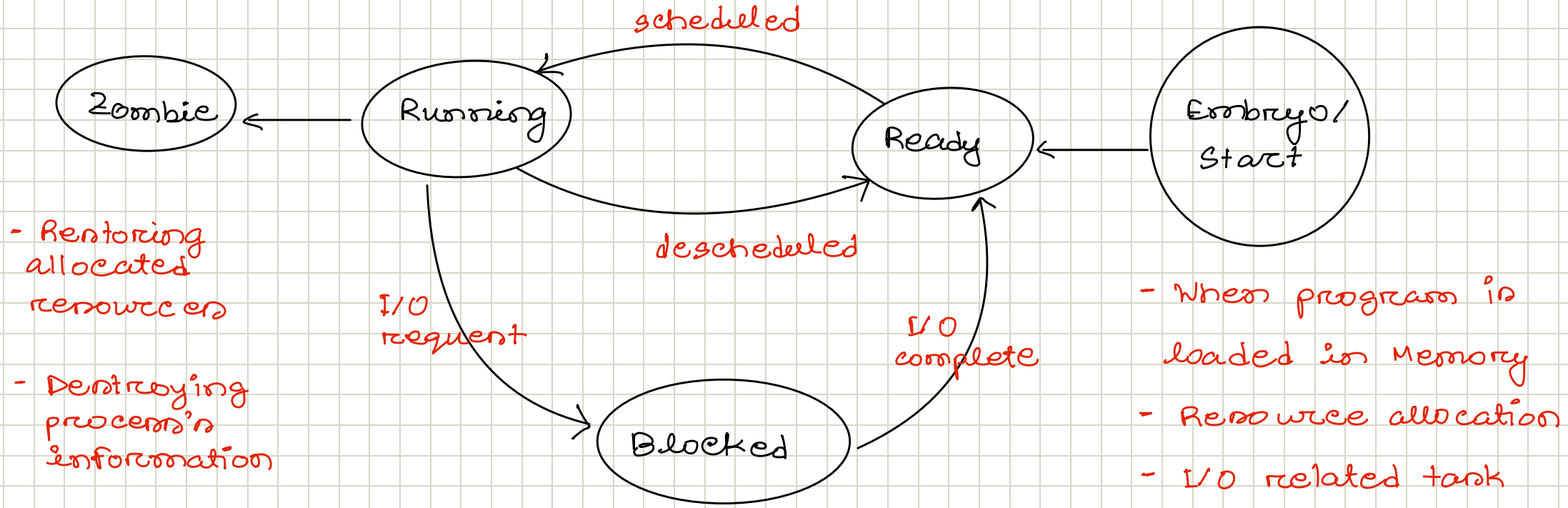
Eagerly loading

- loads whole program to RAM at once
- Enough memory may not be available for large programs

Lazy loading

- loads program part by part on demand
- No difficulties in loading large programs

Process States

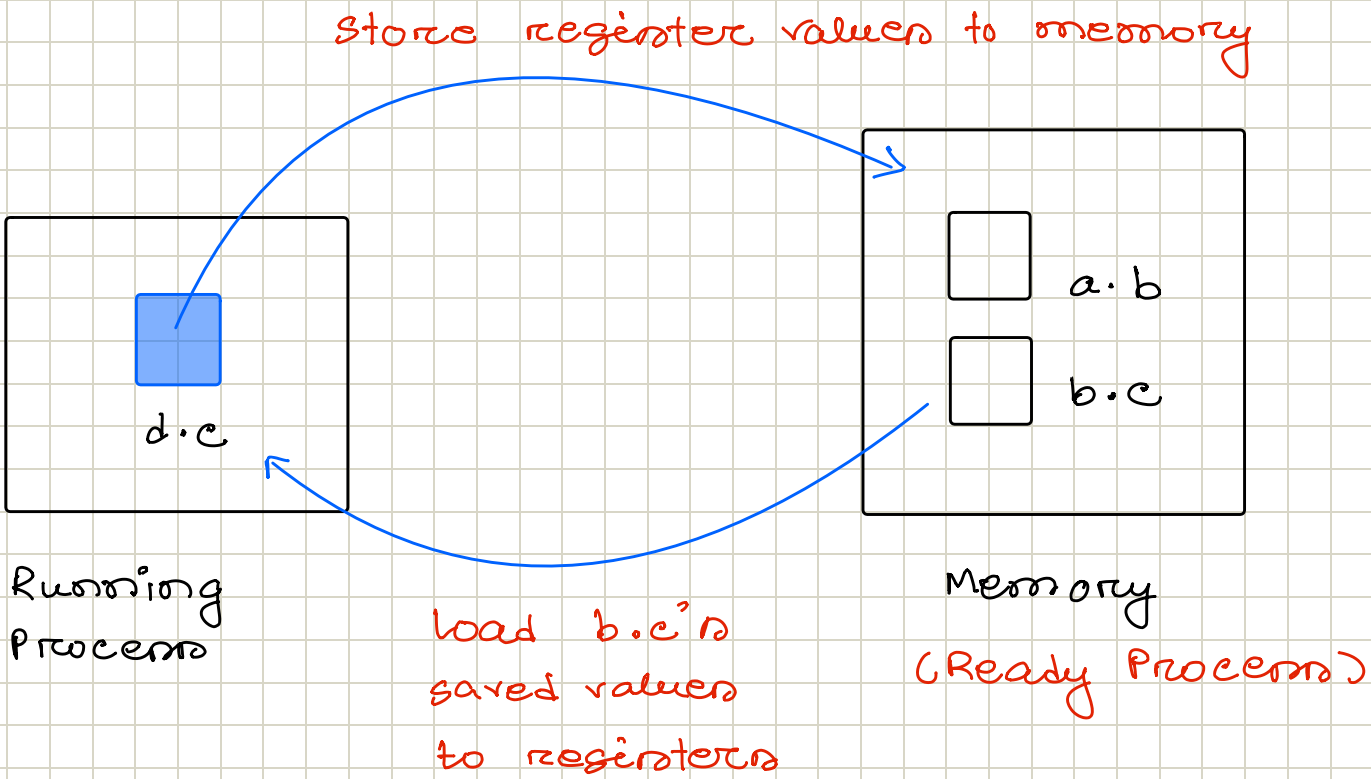


- Restoring allocated resources
- Destroying process's information

- When program is loaded in Memory
- Resource allocation
- I/O related task

context Switch

- Stopped (not killed)
- Save progress in memory
- Register load



Process Creation

request OS

3 system calls are used:

(code, stack, heap)

Process which called fork()

from parent process

(1) fork(): creates an exact copy of parent process. Only process id, memory and I/O pointers are different.

Newly created process is called child process.

called

(2) wait(): waits until a child process terminates

(3) exec(): overwrites the contents of child process (code, stack, heap) with a new program's content

called from child process