

# Lezione 5:

# Software

Firmware  
Sistema Operativo

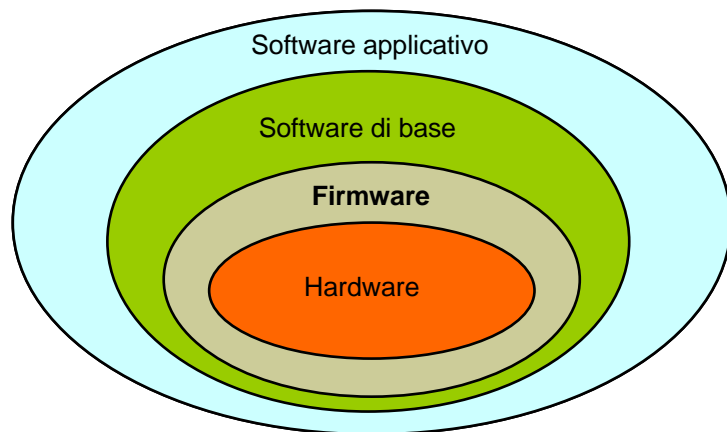
# Architettura del Calcolatore

- ◆ La prima decomposizione di un calcolatore è relativa a due macro-componenti:

Hardware e Software

- ◆ Firmware:

- strato di (micro-)programmi che agiscono direttamente sullo strato hardware
- memorizzato dal costruttore su memoria permanente (ROM)



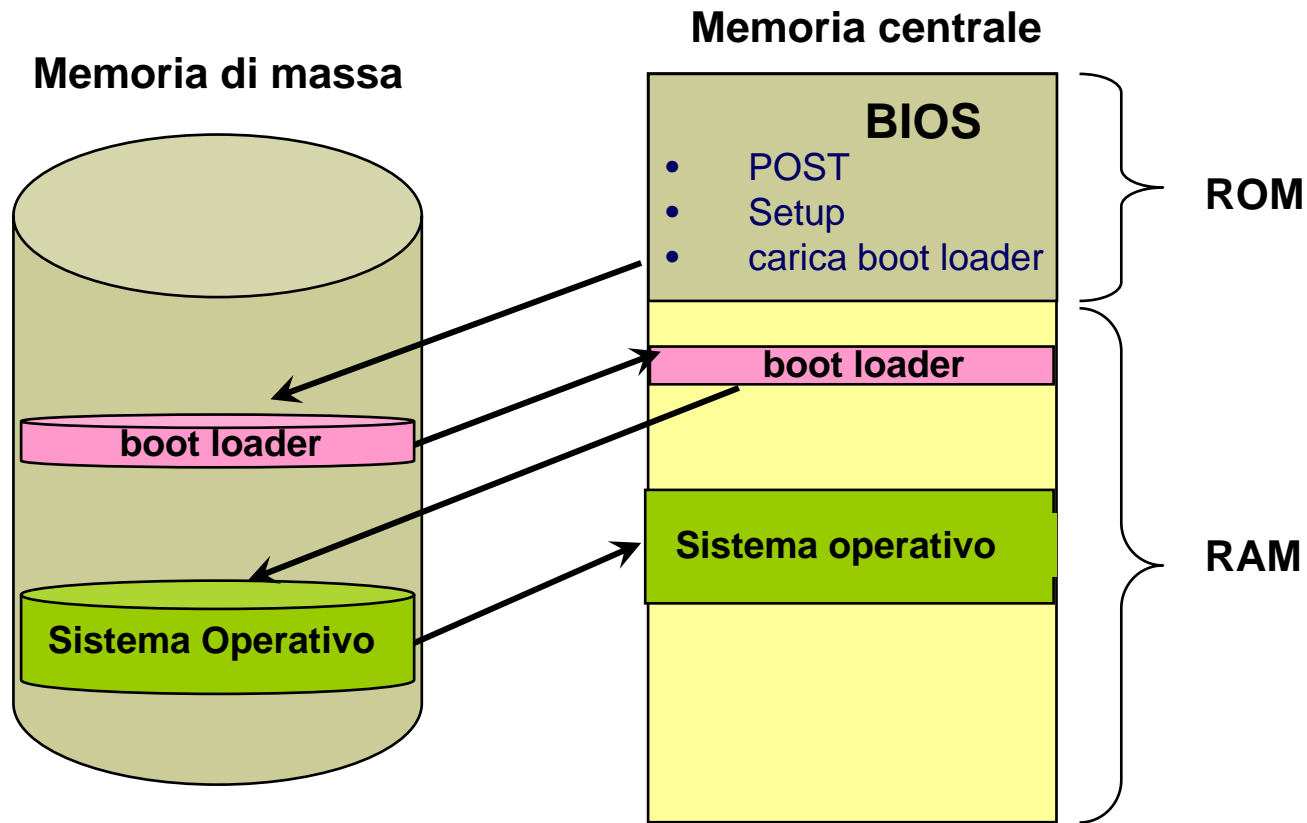
Firmware permette la comunicazione tra il dispositivo in cui il firmware è integrato e il sistema operativo di un computer, traducendo le istruzioni ricevute in operazioni concrete (ad es: sposta la testina, scrivi un dato, leggi un dato).

Il firmware della scheda madre è chiamato comunemente BIOS.

# Firmware: il BIOS

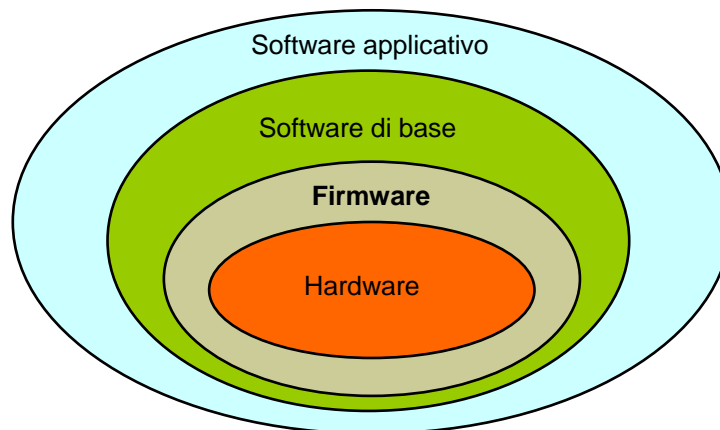
- ◆ BIOS = Basic Input-Output System
  - gestisce direttamente le risorse hardware ed offre delle funzionalità standard di accesso (*terminal driver*)
- ◆ risiede su un chip di memoria permanente
  - ROM (e/o RAM + batteria di alimentazione)
- ◆ gestisce la procedura di avviamento (bootstrap) del calcolatore, consistente delle seguenti fasi
  1. diagnostica (POST: *Power-On Self Test*)
  2. inizializzazione delle risorse hardware (*Setup*)
  3. caricamento (dal disco rigido verso la RAM) ed esecuzione della routine di bootstrap, che provvede quindi a caricare il sistema operativo

# Procedura di avvio



# Sistema Operativo

- ◆ Software = insieme (complesso) di programmi.
  - organizzato a strati, ciascuno con funzionalità di livello più alto rispetto a quelli sottostanti:
- ◆ **Sistema Operativo** è lo strato di programmi che opera al di sopra dell'hardware (e del firmware) e gestisce l'elaboratore



# Sistema operativo (SO)

## ◆ Come **Gestore delle Risorse**

- controlla e gestisce tutte le funzioni del calcolatore in modo efficiente
- accetta e soddisfa le richieste degli utenti/programmi
- funziona come mediatore tra risorse in conflitto
- tiene traccia dell'utilizzo delle risorse

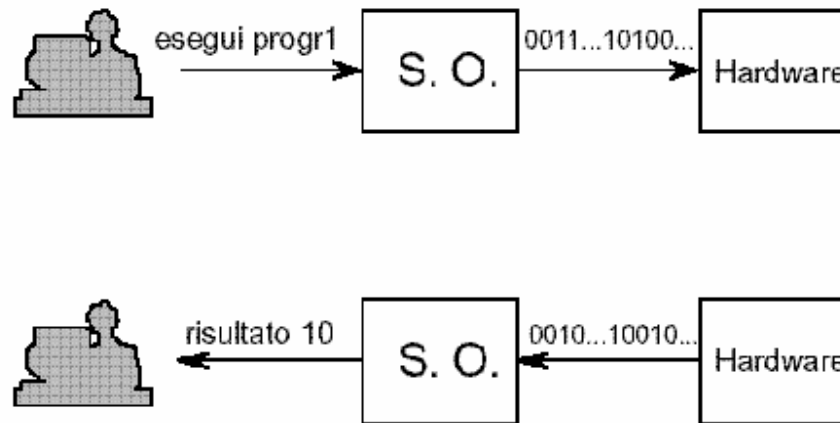
## ◆ Come **Macchina estesa**

- simula una macchina estesa più facile da programmare, astruendo dai dettagli tecnologici dell'hardware
- costituisce una base sulla quale è possibile scrivere programmi applicativi

# Interazione dell'utente con il SO

- ◆ Un utente "vede" l'elaboratore solo tramite il SO, che simula una "macchina virtuale"
  - diversi SO possono realizzare diverse macchine virtuali sullo stesso hardware
  - aumenta l'astrazione nell'interazione utente/elaboratore
    - senza SO: sequenze di bit
    - con SO: comandi, programmi, dati
- ◆ Il S.O. traduce le richieste dell'utente in opportune sequenze di comandi da sottoporre alla macchina fisica
  - Il SO esplicita qualsiasi operazione di accesso a risorse hardware, implicitamente implicata dal comando dell'utente

# Interazione dell'utente con il S.O.



<b>Utente</b>	<b>S.O.</b>
"esegui progr1"	lettura comando da dispositivo di input ricerca codice di "progr1" su disco carica in RAM codice e dati <elaborazione>
"risultato = 10"	output su video



# Interazione dell'utente con il S.O.

## ◆ Interfaccia testuale

- Il S.O. interagisce con l'utente mediante l'interpretazione di linee di comando
- esempi: DOS, Linux

## ◆ Interfaccia grafica o GUI (Graphical User Interface)

- tutti i programmi e le funzioni sono mostrati sullo schermo mediante simboli immediatamente comprensibili (icone)
- esempi: Windows, MacOS (Macintosh)

# Classificazione dei SO

## ◆ In base al numero di utenti:

### ■ mono-utente (mono-user)

- un solo utente alla volta può utilizzare il sistema

### ■ multi-utente (multi-user)

- più utenti in contemporanea interagiscono con la macchina
- il S.O. dà a ciascuno l'astrazione di un sistema "dedicato"

## ◆ In base al numero di processi:

### ■ Mono-programmato (mono-task)

- si può eseguire un solo programma per volta

### ■ Multi-programmato (multi-task)

- il SO permette di eseguire più programmi in contemporanea
- il SO gestisce la suddivisione del tempo fra i vari processi

# Moduli del S.O.

## ◆ Moduli di un Sistema Operativo

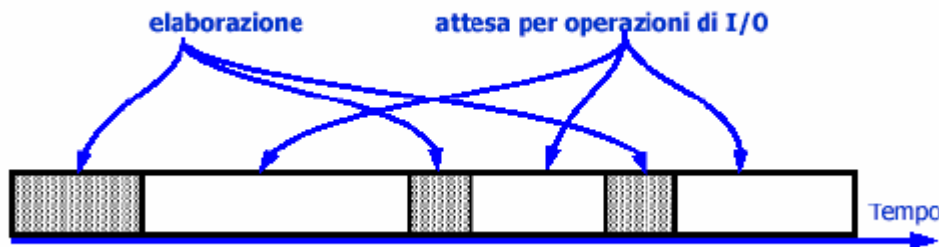
- gestore dei processi
- gestore della memoria
- gestore delle periferiche
- gestore dei file (*File system*)
- interprete dei comandi

## ◆ Le funzioni del S.O. dipendono dalla complessità del sistema di elaborazione:

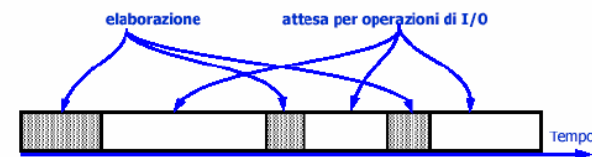
- gestione delle varie risorse hardware
- gestione della multi-utenza e del multi-tasking
- gestione della memoria centrale
- organizzazione e gestione della memoria di massa
- interpretazione ed esecuzione di comandi elementari

# Gestione dei processi

- ◆ La CPU esegue programmi
  - Si chiama **processo** l'esecuzione di un programma
- ◆ Qualunque processo alterna fasi di esecuzione a fasi in cui è bloccato in attesa di qualche evento esterno
  - attesa che sia terminata un'operazione di input
  - attesa per usare una risorsa al momento occupata



# Strategie di esecuzione



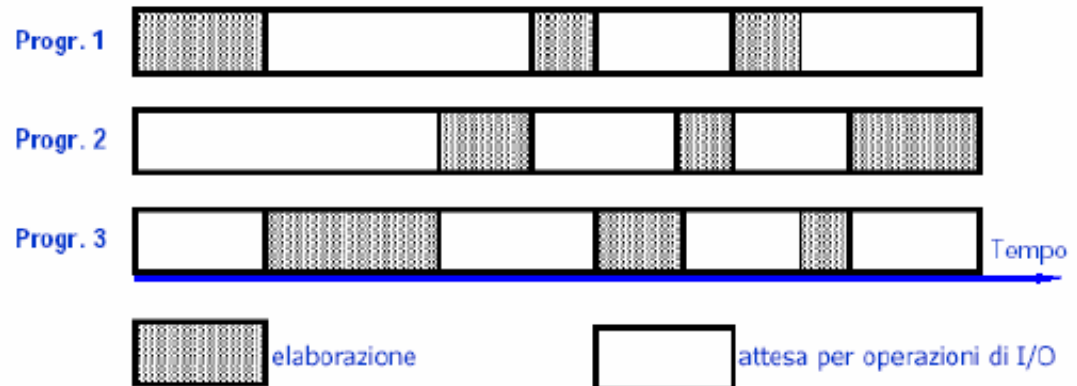
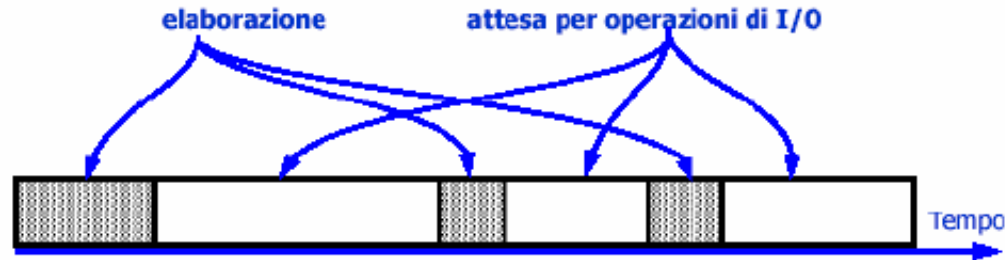
## ◆ Strategie di esecuzione dei programmi

- **mono-tasking**: un processo per volta
- **multi-tasking**: più processi in contemporanea

## ◆ Limiti del mono-tasking

- Limitazione all'uso del calcolatore
  - un solo utente e una sola applicazione per volta
- Sotto-utilizzo del processore
  - mentre il processo è bloccato in attesa di eventi esterni, il processore rimane inattivo (idle)
  - i tempi di lavoro delle periferiche di I/O, e ancor più i tempi di reazione umani, sono maggiori di molti ordini di grandezza della velocità del processore

# Multi-tasking

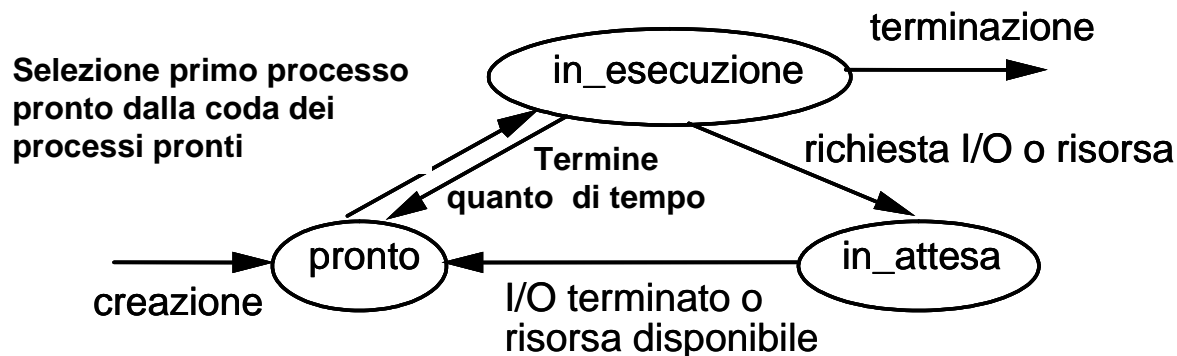


# Multi-tasking

- ◆ Il tempo di lavoro della CPU è diviso tra i vari processi
  - Ad ogni istante vi è un solo processo attivo
  - Il processore alterna l'esecuzione dei vari programmi
- ◆ Se l'alternanza tra i processi è frequente (es. 10 ms), si ha l'impressione di un'esecuzione simultanea
  - a livello macroscopico si ha quindi l'impressione della contemporaneità, mentre a livello microscopico si ha una semplice alternanza sequenziale molto veloce
- ◆ Il tempo totale di esecuzione di un singolo processo aumenta rispetto al caso mono-tasking
  - a causa dell'alternanza con gli altri processi

# Gestione dei processi

## ◆ Stati di un processo:



## ◆ Problematiche di concorrenza fra processi

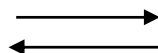
### ■ Starvation:

più processi potrebbero accedere alla risorsa condivisa ma solo alcuni di essi vi riescono

### ■ Deadlock (blocco critico)

un gruppo di processi rimane permanentemente bloccato senza che la risorsa contesa viene utilizzata da alcuni di essi

Processo A: per concludere mi serve il risultato di B



Processo B: per concludere mi serve il risultato di A



# Esempio di esecuzione di un programma (vedi Lezione 3)

Memoria centrale

Registri CPU

1000	Load	3568	R1
1001	Add	R1	R2
1002	Store	R1	3568
1003	Jump	1000	

3568

44

R1

R2

30

PC

1000

IR

--	--	--

MAR

--

MDR

--	--	--

Livello HARDWARE

# Gestione della memoria principale

## ◆ Visione astratta della memoria:

- un programma non deve conoscere la configurazione e le dimensioni della memoria reale e può essere eseguito su computer con dotazioni di memoria differenti
- il programma ignora gli indirizzi (fisici) delle celle di memoria effettivamente usate

## ◆ Nel caso multi-tasking la memoria deve essere condivisa da più processi:

- la memoria viene suddivisa in blocchi (paginazione)
- ad ogni programma si assegna un certo numero di blocchi (non necessariamente contigui)
  - si può caricare un numero maggiore di programmi

# Risoluzione degli indirizzi

## ◆ Indirizzi logici e indirizzi fisici:

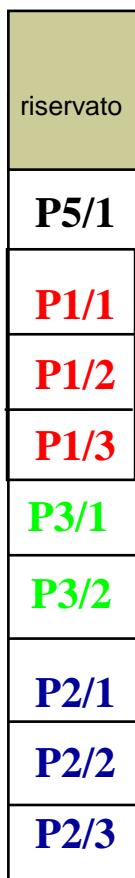
- **indirizzi logici**: gli indirizzi presenti nei programmi
- **indirizzi fisici**: gli indirizzi RAM assegnati al programma quando viene caricato dal disco

## ◆ Risoluzione degli indirizzi

- Per poter essere caricato a blocchi il programma viene suddiviso in blocchi logici
- il SO associa ogni blocco logico ad uno fisico trasformando gli indirizzi logici in quelli fisici

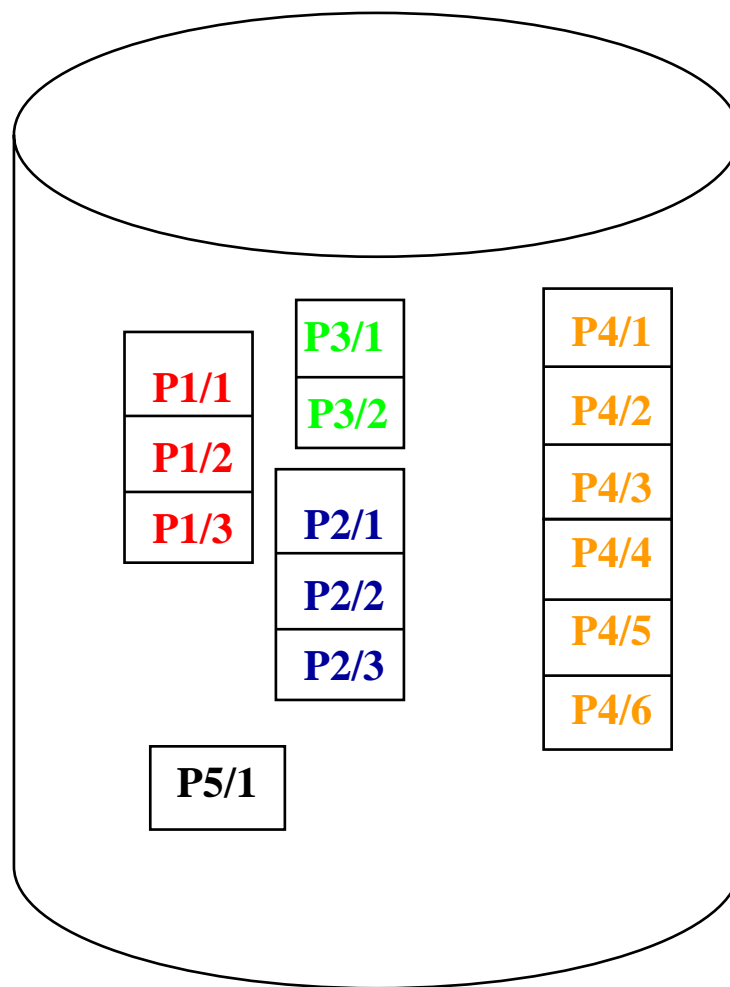
# Gestione della memoria principale: esempio

Memoria centrale



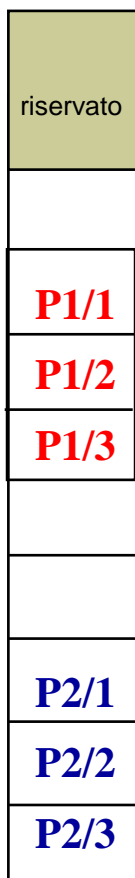
Utente chiede di eseguire:

- 1) P5
- 2) P1
- 3) P3
- 4) P2



# Gestione della memoria principale: esempio

Memoria centrale

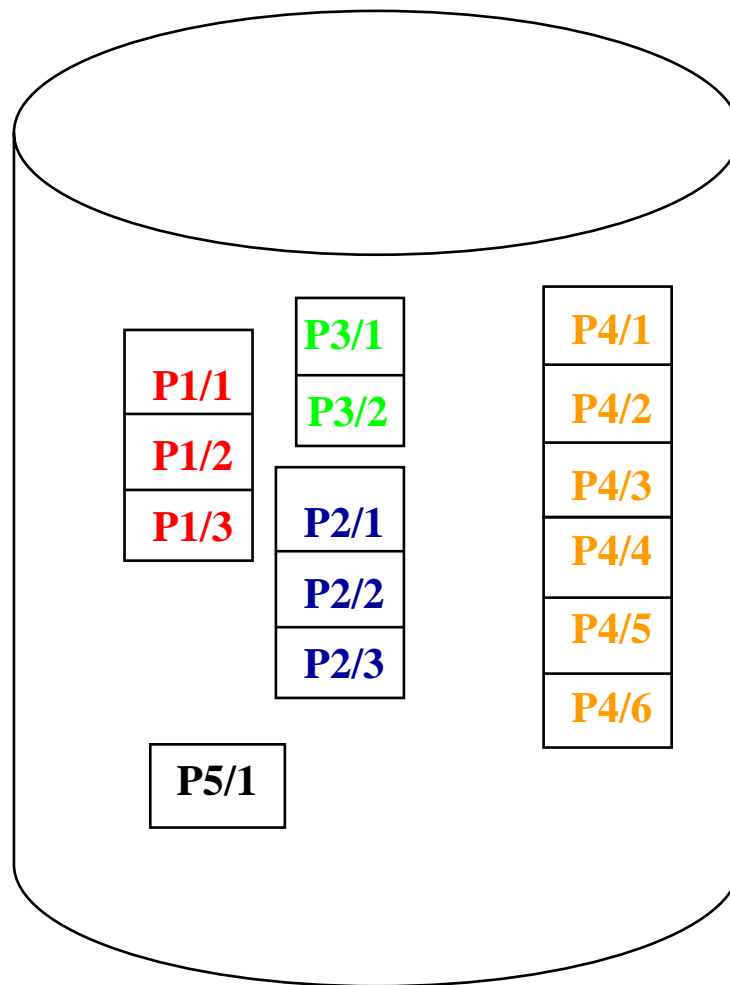


Utente chiude:

1)P5

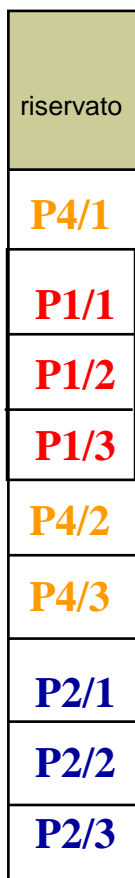
2)P3

Utente chiede di  
eseguire P4

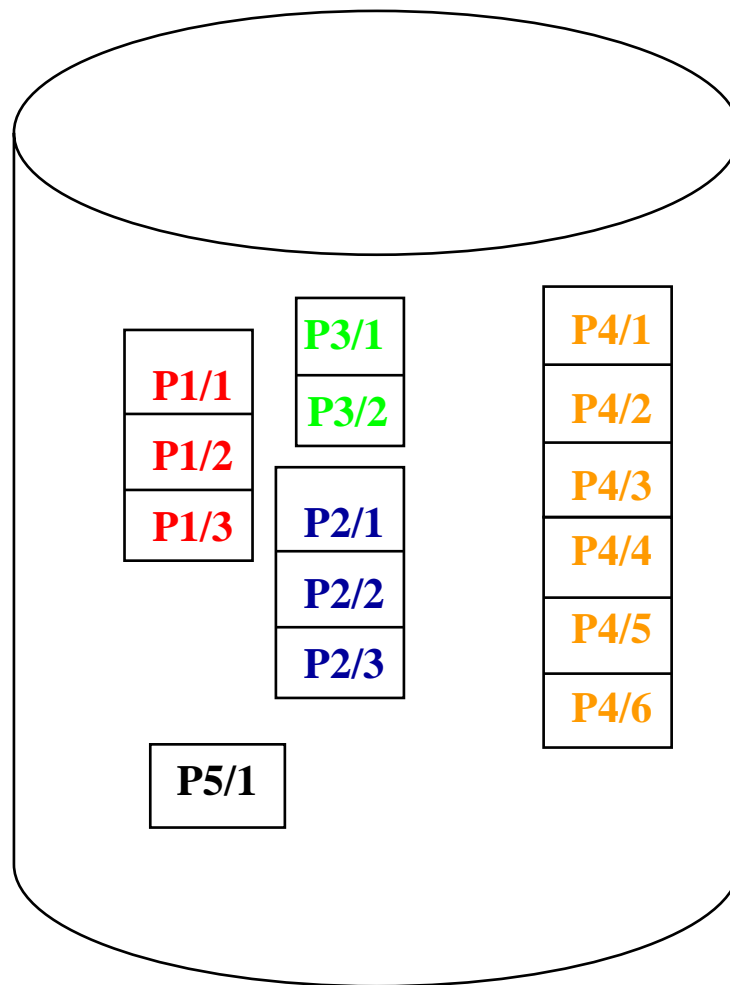


# Gestione della memoria principale: esempio

Memoria centrale



Utente chiede di eseguire P4



# Memoria virtuale

- ◆ Come è possibile eseguire uno o più programmi contemporaneamente che richiedono più memoria di quanta sia disponibile?
- ◆ Per eseguire un programma non è necessario caricarlo completamente in memoria:
  - basta caricare in memoria principale solo le parti del programma e dei dati che servono durante una certa fase dell'esecuzione
- ◆ Per gestire la memoria in modo virtuale, si usa:
  - la memoria principale
    - in cui tenere solo i programmi, o i pezzi di programmi, e i dati che servono in un certo istante (compreso il SO)
  - un supporto di memoria secondaria
    - in cui mantenere tutte le informazioni relative ai processi in esecuzione, non contenute nella RAM
    - si usano i dischi rigidi perché sono abbastanza veloci e hanno accesso diretto

# Memoria virtuale

- ◆ Le pagine sono caricate nella RAM indipendentemente, quando sono richieste per l'esecuzione (**on demand**)
  - Il SO stabilisce quali pagine eliminare dalla RAM per far posto a nuove pagine di processi in esecuzione
  - se le pagine sono state modificate devono essere ricopiate sul disco
- ◆ Il processo di scambiare pagine tra memoria e disco si chiama **swapping**
  - Lo swapping è costoso in termini di tempo e rallenta l'esecuzione di un programma



# Gestione delle periferiche

- ◆ **Driver:** programmi del SO per la gestione delle periferiche
- ◆ Funzioni assolute:
  - Sincronizzazione fra calcolatore e ambiente esterno
    - Asincronicità fra CPU-RAM e periferiche
    - Accesso contemporaneo al calcolatore da parte di diverse periferiche
  - Gestione di accessi contemporanei alle periferiche
    - esempio: gestione delle richieste di stampa da parte di più processi attraverso code di spooling
  - Astrazione/standardizzazione
    - mascherare le differenze fra dispositivi dello stesso tipo

# Gestione dei file (file system)

- ◆ Si basa su una strutturazione logica del contenuto delle memorie di massa
  - *File*: sequenze di bit, identificate da un nome
  - *Cartelle* (directory): contenitori di file
  - *Unità di memoria di massa* (*a:*, *b:*, *c:*, *d:*)
- ◆ Funzioni assolute:
  - Astrazione/standardizzazione
  - Reperimento efficiente
  - Gestione degli accessi contemporanei
  - Sicurezza e protezione

# Alcuni sistemi operativi reali

**DOS**

**IBM compatibili con architettura 80x86, monoutente, monoprogrammato**

**Windows 3.1**

**Per le architetture a partire da 80386, monoutente, multiprogrammato**

**Windows '9x**

**Per le architetture a partire da 80486, monoutente, multiprogrammato**

**Windows NT**

**Ambiente di rete per architetture da 80486, multiutente, multiprogrammato**

**OS/2**

**Ambiente per le architetture a partire da 80486, monoutente, multiprogrammato**

**Unix**

**Ambiente vasto e potente, capace di ospitarne altri, montato su gran parte delle workstation, multiutente, multiprogrammato**

**MacOS**

**Noto come Macintosh, monoutente, multiprogrammato**

# Introduzione all'Informatica - corso E

