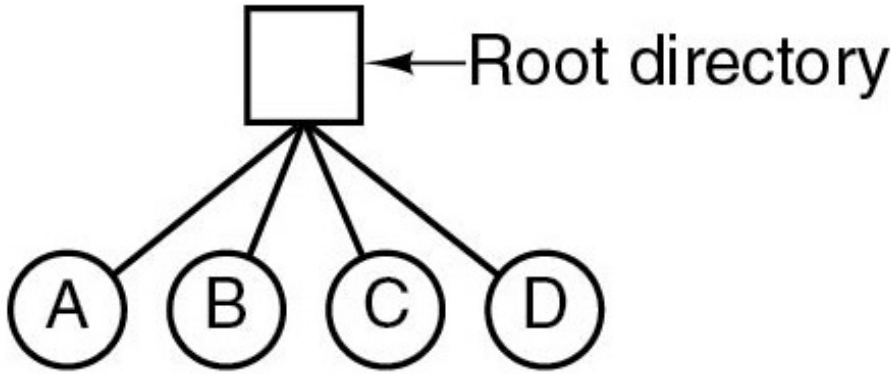
DIRECTORY

LE DIRECTORY NEI FILE SYSTEM

- **Concetto di Directory**:
  - Le directory, o cartelle, sono file che tengono traccia degli altri file all'interno di un file system.
- **Sistemi di Directory a Livello Singolo**:
  - **Struttura Semplice**: Una singola directory, talvolta chiamata root directory, contiene tutti i file.
  - **Esempi Storici**: Comune nei primi PC e nel supercomputer CDC 6600.
  - **Vantaggi**: Semplicità e rapidità nella localizzazione dei file.



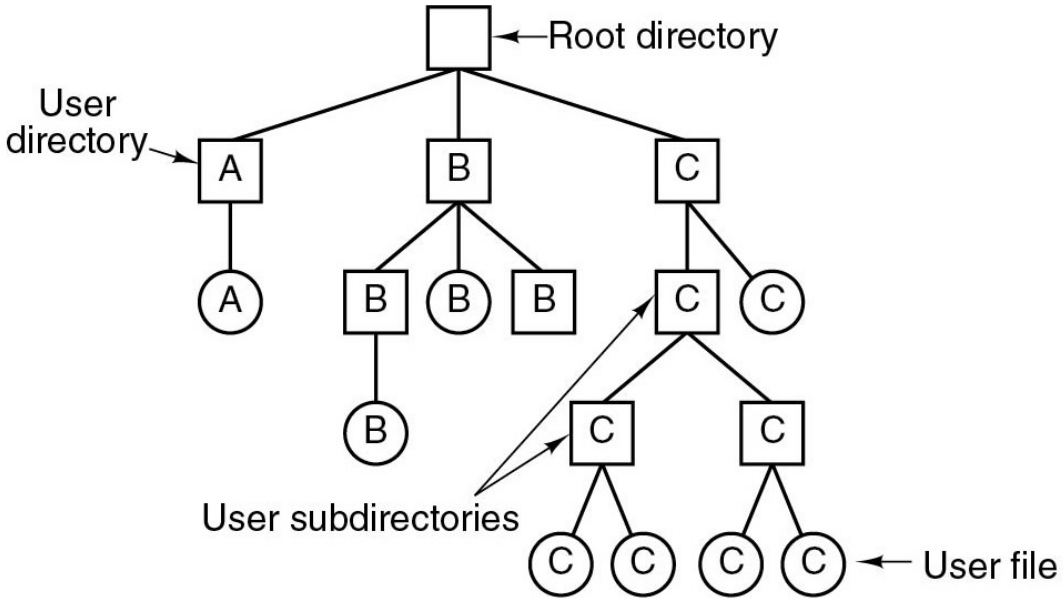
Esempio con quattro file in un sistema a directory singola.

EVOLUZIONE E APPLICABILITÀ DEI SISTEMI DI DIRECTORY A LIVELLO SINGOLO

- **Evoluzione dei Concetti di File System**:
  - Molti concetti, come la directory singola, sono ciclici: emergono, cadono in disuso, e riemergono in nuovi contesti.
- **Applicabilità Moderna**:
  - **Dispositivi Embedded**: Concetti semplici di file system sono ancora utili in dispositivi come fotocamere digitali o riproduttori MP3.
  - **Tecnologie RFID**: Sistemi di directory semplici possono essere adatti per chip RFID o carte di credito e tessere di trasporto.
  - **Riflessione**: Idee apparentemente obsolete possono essere rilevanti in contesti moderni e dispositivi a basso costo.

SISTEMI DI DIRECTORY GERARCHICI

- **Limiti dei Sistemi a Singolo Livello**:
  - **Non pratici** per utenti con **migliaia di file**.
  - Difficoltà nel rintracciare i file in un unico spazio.
- **Introduzione della Gerarchia**:
  - Organizzazione dei file in gruppi correlati mediante directory ramificate.
  - Struttura ad albero per separare e organizzare logicamente i file.
  - **Ogni utente può avere una directory principale** privata in ambienti condivisi come reti aziendali.
- **Importanza nei File System Moderni**:
  - Tutti i file system moderni utilizzano una struttura gerarchica per la loro flessibilità e potenziale organizzativo.
  - Storicamente, il file system gerarchico è stato sperimentato inizialmente in Multics negli anni '60.



La directory principale (Root) divisa in directory A, B e C, ognuna appartenente a utenti diversi.

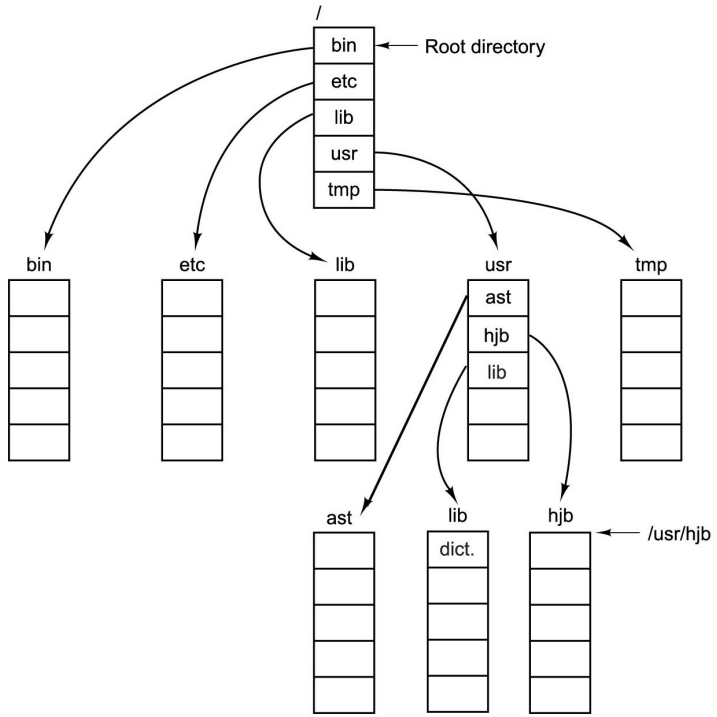
- Possibilità di creare sottodirectory per progetti specifici o categorie di file.

NOMI DI PERCORSO NEI FILE SYSTEM GERARCHICI

- **Specificare i Nomi dei File**:
  - Necessità di definire i percorsi dei file in un sistema di directory ad albero.
- **Nomi di Percorso Assoluti**:
  - Percorsi che iniziano dalla directory principale e conducono al file.
  - Unici per ogni file (es. `/usr/ast/mailbox`).
  - Separatore di percorso: / per UNIX, \ per Windows, > per MULTICS.
- **Nomi di Percorso Relativi**:
  - Basati sulla directory di lavoro (directory corrente) dell'utente.
  - Percorsi non iniziano con il separatore sono considerati relativi (es. mailbox).
  - Esempi di comandi equivalenti in una data directory di lavoro:
    - `cp /usr/hjb/mailbox /usr/hjb/mailbox.bak`
    - `cp mailbox mailbox.bak`

UTILIZZO PRATICO E IMPLICAZIONI

- **Directory di Lavoro (Working Directory)**:
  - Cambia dinamicamente per ciascun processo.
  - Non influisce sugli altri processi o sul file system dopo l'uscita del processo.
- **Procedure di Libreria**:
  - Evitano di cambiare la directory di lavoro o la ripristinano dopo il loro uso.
- **Voci Speciali**:
  - `.` (punto): Rappresenta la directory corrente.
  - `..` (punto punto): Rappresenta la directory genitore.
  - Usati per navigare nell'albero dei file
    - **Esempio** `cp ../lib/dictionary .`



Un esempio di albero di directory UNIX.

OPERAZIONI SULLE DIRECTORY

- **Operazioni di Base**:
  - `create`: Creazione di una directory vuota con le voci "." e ".." predefinite.
  - `delete`: Eliminazione di una directory, possibile solo se la directory è vuota.
  - `opendir`: Apertura di una directory per la lettura del suo contenuto.
  - `closedir`: Chiusura di una directory dopo la lettura per liberare risorse.
- **Lettura e Modifica**:
  - `readdir`: Restituisce la prossima voce in una directory aperta senza esporre la struttura interna.
  - `rename`: Rinomina di una directory, simile al rinomino di un file.

GESTIONE DEI LINK E ACCESSI AVANZATI

- **Linking e Unlinking**:
  - `link`: Crea un hard link, collegando un file esistente a un nuovo percorso, condividendone l'i-node.
  - `unlink`: Rimuove una voce di directory, cancellando il file se è l'unico link.
- **Link Simbolici** (*vedi lezioni successive*)
  - Varianti dei hard link che possono puntare a file su dischi o computer diversi.
  - Rappresentano un file tramite un riferimento indiretto che il file system risolve all'uso
    - (meglio nella prossima lezione)
- **Considerazioni Aggiuntive**:
  - Esistono altre chiamate per gestire dettagli come le informazioni di protezione di una directory.
  - I link simbolici offrono flessibilità oltre i limiti dei dischi ma possono essere meno efficienti rispetto agli hard link.

DESCRIZIONE DEL PROGRAMMA

11_SHOW_DIR_CONTENT.C

- **Funzionalità Aggiunta**: Oltre a stampare i nomi, mostra informazioni dettagliate sui file nella directory, simili al comando `ls -lh`.
- **Informazioni Incluse**:
  - Dimensione del file.
  - Permessi di accesso (lettura, scrittura, esecuzione).
  - Proprietario e gruppo del file.
  - Data e ora dell'ultima modifica.
- **Implementazione**:
  - Uso della funzione `stat()` per ottenere i metadati dei file.
  - Formattazione e stampa delle informazioni in modo chiaro e leggibile.

CREAZIONE DI ARCHIVI

COMPRENDERE I FILE TAR.GZ PER LA COMPRESSIONE IN LINUX

- **File TAR: Cosa Sono e Perché si Usano**
  - TAR (Tape Archive) è un formato usato per raccogliere più file e cartelle in un unico archivio, mantenendo la struttura e i permessi originali.
  - Utilizzato comunemente per raggruppare file correlati per backup, trasferimento o archiviazione.
- **Comprimere con GZ**
  - Dopo l'archiviazione con tar, l'archivio viene compresso con gzip per ridurre lo spazio su disco.
  - gzip è un algoritmo di compressione che riduce efficacemente la dimensione del file senza perdita di dati.

UTILIZZO DI TAR E GZ

- **Creazione di un Archivio tar.gz**
  - Comando di Base: `tar -czvf nome-archivio.tar.gz /percorso/della/cartella`
    - `c`: crea un nuovo archivio.
    - `z`: comprime l'archivio usando gzip.
    - `v`: visualizza un output verboso.
    - `f`: specifica il nome del file di archivio.
- **Estrazione di un Archivio tar.gz**
  - Comando di Base: `tar -xzvf nome-archivio.tar.gz`
    - `x`: estrae il contenuto dall'archivio.
    - `z`: decomprime l'archivio usando gzip.
    - `v`: visualizza un output verboso.
    - `f`: specifica il nome del file di archivio.
- In realtà viene creato un file con il comando tar e poi compresso con gzip.
  - Nulla vieta di comprimere con `gzip` un qualsiasi file

CONFRONTO TRA ZIP/UNZIP E TAR.GZ PER LA COMPRESSIONE IN LINUX

- **ZIP e UNZIP: Caratteristiche**
  - ZIP è un formato di compressione che riduce la dimensione dei file singolarmente prima di archiviarli insieme.
  - UNZIP è utilizzato per decomprimere e estrarre i file dagli archivi ZIP.
  - Comandi Comuni:
    - `zip nome-archivio.zip file1 file2,`
    - `unzip nome-archivio.zip.`
  - Vantaggi: Compatibilità ampia con diversi sistemi operativi, compressione individuale dei file.
- **TAR.GZ: Caratteristiche**
  - TAR raccoglie molti file in un unico archivio, poi GZ (gzip) comprime l'intero archivio.
  - **Vantaggi**: Elevata compressione, conservazione della struttura delle directory e dei permessi dei file.
- **Confronto tra ZIP e TAR.GZ**
  - **Efficacia di Compressione**: TAR.GZ tende ad avere un tasso di compressione più alto, specialmente per archivi di grandi dimensioni.
  - **Velocità**: ZIP può essere più veloce nella compressione di file individuali.
  - **Conservazione dei Metadati**: TAR.GZ mantiene meglio la struttura originale e i permessi dei file.
  - **Compatibilità Universale**: ZIP è più comunemente supportato su diverse piattaforme, incluse Windows e macOS.