

# Introducció a la Programació de Sistema

## Unitat 0 - Nel Banqué Torné

---

### 1. Introducció

En aquesta secció, s'explorà el desenvolupament de programari de sistema en entorns Linux/UNIX. Utilitzarem el llenguatge de programació **C** i ens basarem en el compilador **GNU GCC** per portar a terme les nostres aplicacions.

#### Comprovar la instal·lació del compilador:

Per assegurar-nos que tenim instal·lat el compilador i les eines necessàries, podem executar el següent script:

```
bash
Copiar código
./check.sh
gcc -v || echo "GCC is not installed!"
ld -v || echo "Please install binutils!"
```

---

### 2. Navegant a les Entranyes del Compilador GCC

El compilador **GCC** (GNU Compiler Collection) realitza diverses operacions en el procés de compilació:

- **Genera un fitxer de codi ensamblador** amb l'extensió **.s**. Per exemple, si el fitxer de codi font és **hola.c**, el fitxer generat seria **hola.s**.
- **Genera l'executable final**, que és el fitxer que es pot executar. Per exemple, **hola**.
- Tots aquests fitxers generats es poden incloure al fitxer **.gitignore** per evitar que es versionin.

---

## 3. Argumentos i Maneig de Fitxers

### Arguments en línia de comandes

- **argv** és una representació tokenitzada dels arguments passats al programa. Mai està buit i sempre conté la ruta completa a l'executable.

### Obrir Fitxers

Per obrir fitxers en C, utilitzem la funció **open**, que permet especificar diversos modes:

- **O\_RDONLY**: Només lectura.
- **O\_WRONLY**: Només escriptura.
- **O\_RDWR**: Lectura i escriptura.
- **O\_CREAT**: Crea el fitxer si no existeix.

---

## 4. Tancament de Fitxers

Per tancar un fitxer, utilitzem la funció **close**, que desassocia el fitxer del procés.

- **fd**: Descriptor de fitxer que volem tancar.

### Llegir i Escriure Fitxers

Per llegir o escriure fitxers, utilitzem la funció **read** i **write**, respectivament.

- **fd**: Descriptor del fitxer que es llegirà/escriurà.
- **nbytes**: Nombre de bytes a llegir/escriure.

Si l'operació és exitosa, la funció retorna el nombre de bytes llegits o escrits.

---

## 5. Gestió de Punter i Desplaçament

### Punter a fitxers

Per controlar la posició de lectura/escriptura en un fitxer, utilitzem la funció **lseek**.

- **fd**: Descriptor de fitxer.
- **offset**: Desplaçament relatiu del punter en bytes.
  - **SEEK\_SET**: Col·loca el punter a **offset** bytes des del començament del fitxer.
  - **SEEK\_CUR**: Mou el punter **offset** bytes des de la seva posició actual.

Si l'operació és exitosa, retorna la nova posició absoluta del punter.

---

## 6. Memòria: Stack vs. Heap

### 6.1 Stack

La **stack** és una regió especial de memòria gestionada automàticament per la CPU. Té les següents característiques:

- **Ordre seqüencial:** Les variables s'empilen i desempilen en ordre seqüencial.
- **Gestió automàtica:** No cal que el programador assigni o alliberi memòria manualment.
- **Limitació de mida:** La pila és limitada, i si es supera el seu límit, es produeix un **desbordament de pila**.

### 6.2 Heap

La **heap** és una àrea de memòria on s'assigna memòria de manera dinàmica durant l'execució del programa.

**Característiques de la Heap:**

- **Gestió manual:** El programador ha de controlar l'assignació i alliberament de memòria.
  - **Flexibilitat:** És útil per a grans blocs de memòria o per a objectes que han de romandre en memòria més temps que una funció específica.
  - **Limitació per memòria física:** La mida de la heap està limitada per la quantitat de memòria física disponible al sistema.
- 

## 7. Errors de Memòria

### Què és un Stack Overflow?

Un **stack overflow** es produeix quan la pila del programa supera la seva capacitat màxima, generalment a causa de recursions excessives o d'una gestió incorrecta de la memòria.

---

## 8. Tipus de Dades Derivades

Una **estructura** és un tipus de dades derivades format per membres que són tipus de dades fonamentals o derivats.

## **Tipedef**

El `typedef` s'utilitza per crear sinònims per a noms de tipus de dades definits prèviament, facilitant la lectura i la gestió del codi.