ソフトウェア

1. 先着順（FCFS: First Come, First Served）
- 到着順に処理する方式
- 公平だが、1つが長いと後続が待たされる（コンボイ現象）

2. 優先度方式（Priority Scheduling）
- 優先度の高いタスクから処理
- 重要処理を先にできる
- 低優先度タスクがずっと処理されない「飢餓状態」が発生する可能性

3. 最短ジョブ優先（SJF: Shortest Job First）
- 処理時間の短いタスクから先に処理
- 平均待ち時間を短縮できる
- 長いタスクが後回しにされがち

4. ラウンドロビン（RR: Round Robin）
- 一定時間ごとに順番に処理を交代
- 公平性が高い
- インタラクティブシステムに適する

プリエンプション方式とは
👉 **OSが実行中のプロセスを強制的に中断させ、CPUを他のプロセスに割り当てる方式。**

🔷 特徴
- **強制割り込み可能**：OSが「お前はもう少し待て」とCPUを取り上げる
- **マルチタスクの実現**に不可欠
- 優先度の高いタスクがすぐに実行できる
- タイムシェアリング（ラウンドロビン方式）でも利用

ページ置換アルゴリズム（主記憶管理）
1. FIFO（First In, First Out）
- **最も古くから主記憶にあるページを置き換える**方式
- 例：行列の先頭から順番に追い出す
- **メリット**：実装が簡単
- **デメリット**：最近よく使っているページでも古ければ消される（Beladyの例外あり）

2. LRU（Least Recently Used）
- **最も長い間使われていないページを置き換える**方式
- 例：本棚の中で「最後に読んだのが一番古い本」を出す
- **メリット**：現実のアクセスパターンに近い
- **デメリット**：利用履歴を管理する必要があり、実装コストが高い

3. LFU（Least Frequently Used）
- **参照回数が最も少ないページを置き換える**方式
- 例：図書館で「借りられた回数が一番少ない本」を処分

- **メリット**：使われにくいページを排除できる
- **デメリット**：昔一度だけ大量に使われ、その後使われないページが残り続けることがある

4. OPT（Optimal, 最適置換）
- **将来最も長い間使われないページを置き換える**方式
- 例：これから一番長く読まれない本を予知して入れ替える
- **メリット**：理論的にページフォールトが最小
- **デメリット**：将来のアクセスを予測できないので現実的には実装不可（理論比較用）

局所性（Locality）とは
👉 プログラムのアクセスには「近い場所や近い時間のデータを繰り返し使う」傾向があること

時間的局所性（Temporal Locality）
- **最近使ったデータや命令は、近い将来また使われやすい**

空間的局所性（Spatial Locality）
- **あるアドレスにアクセスしたら、その近くのアドレスも使われやすい**

逐次的局所性（Sequential Locality）
- **プログラムの命令は基本的に順番に実行される**

ページイン -> ページを仮想記憶から主記憶に
ページアウト -> ページを主記憶から仮想記憶に

セマフォは、信号機という意味で、並行動作している複数のタスク間で共通して使用する資源へのアクセスを制御するメカニズム

スラッシングは、主記憶の容量が十分でない場合に、多数のプログラムを同時に実行すると、ページ置き換え処理が多発する。これにより、システムのオーバーヘッドが増加し、実行中のプログラムがCPUを使用している割合が極端に少なくなること。