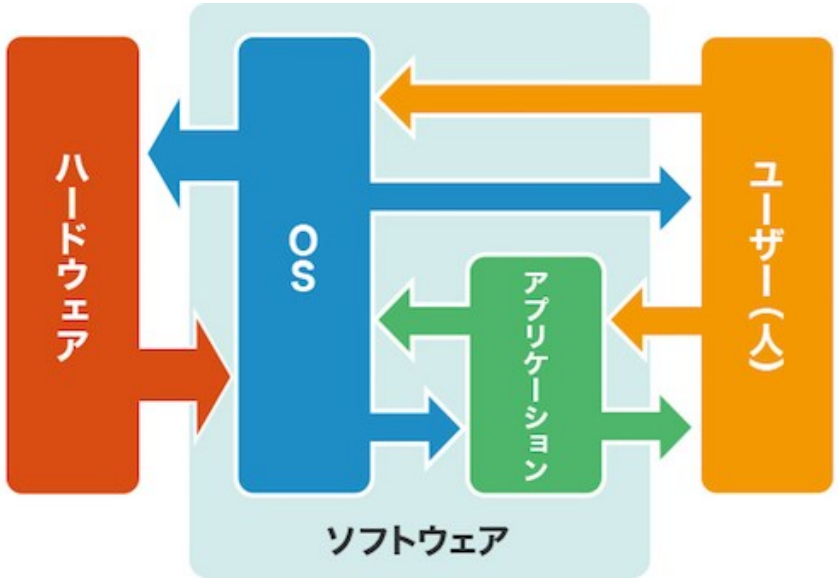
ソフトウェア

http://cobayasi.com/koza/kihon/2_softwear.pdf

1. ソフトウェアとOS ★
2. タスク管理 ★★★
3. 記憶管理 ★★★
4. ファイル管理 ★★

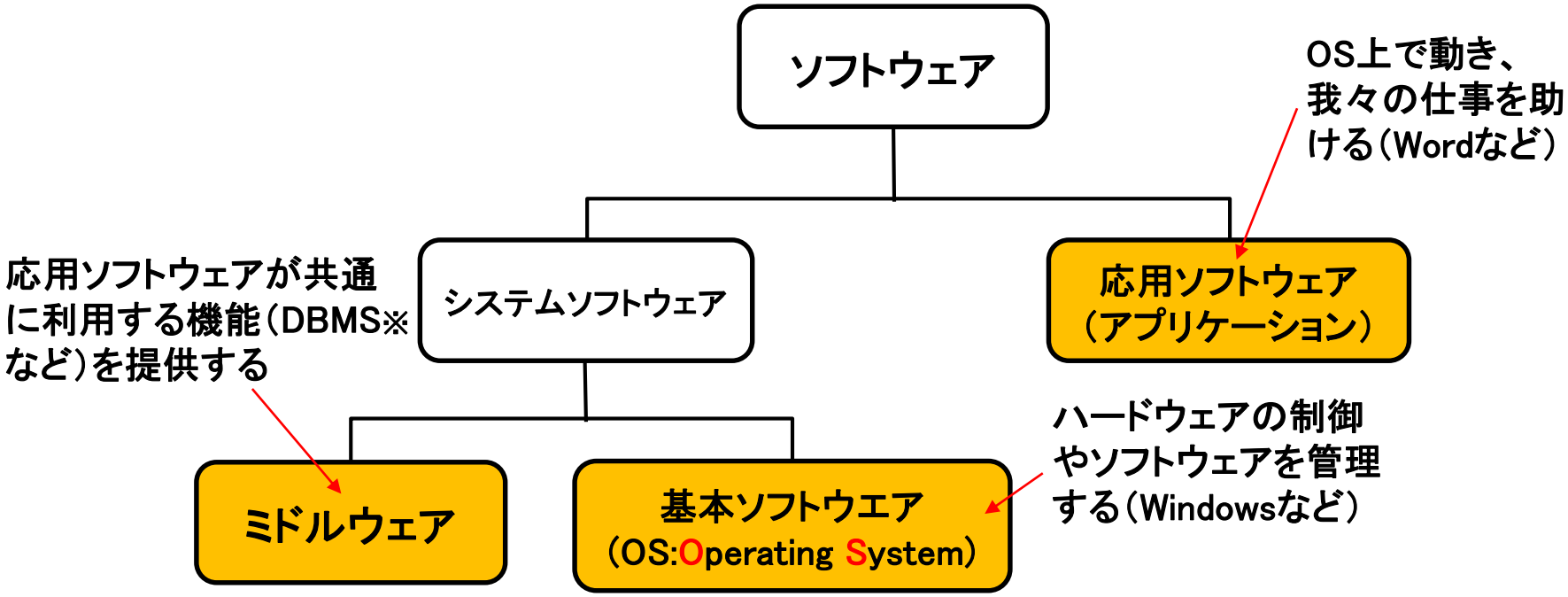
1. ソフトウェアとOS

- ソフトウェアの分類
- OSの役割



● ソフトウェアの分類

ソフトウェアの役割によって3つに分類される



※DBMS：DataBase Management System
　データベースを管理し、外部のソフトウェアからの要求に応えてデータベースの操作を行う専門のソフトウェアデータベース管理システム

● OSの役割

■ ハードウェア管理（タスク管理、記憶管理に関連）

ハードウェア資源（CPUやメモリなど）を、効率よく使用できるように仕事（タスク）を割り振る

■ ファイル管理

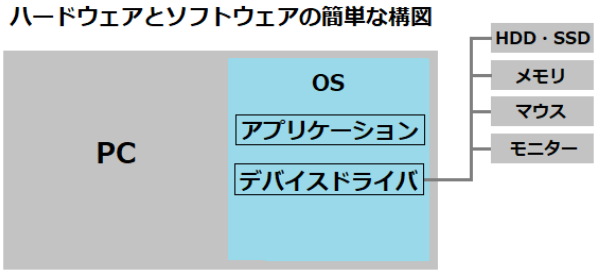
ファイルやフォルダの作り方や使い方が適切であるかを管理する

■ 周辺機器管理

周辺機器（マウスやプリンタなど）を、ユーザがスムーズに利用できるように支援する。また、新しい周辺機器を接続したときは、適切なソフトウェア（デバイスドライバ）をインストールする。

これも
知っとこ

デバイスドライバ(Device Driver)



コンピュータの内部や外部に、接続した機器や装置などのハードウェアを制御・操作するためのソフトウェア

【過去問題】
デバイスドライバの役割として、適切なものはどれか。

ア　アプリケーションプログラムの要求に従って、ハードウェアを直接制御する。
イ　実行を待っているタスクの中から、次に実行するタスクを決定する。
ウ　複数のウインドウの画面上で、表示状態を管理する。
エ　利用者が入力するコマンド文字列を解釈して、対応するプログラムを起動する。

これも
知っとこ

API(Application Program Interface)

- 応用ソフトウェアからOSの各種機能を利用するための仕組み
- OSの機能を利用するための関数として提供されている
- APIが出来るまでは、OSの機能を利用するためのプログラムを、そのつど開発していた。APIが出来てからは、それらの関数を呼び出すだけで、OSの機能を利用できるようになった

【過去問題】
パソコンのOSが提供する機能を利用するためのAPIに関する記述のうち、適切なものはどれか。

a. API で呼び出される OS の処理モジュールは，あらかじめそれを利用するプログラムに 静的にリンクしておく必要がある。

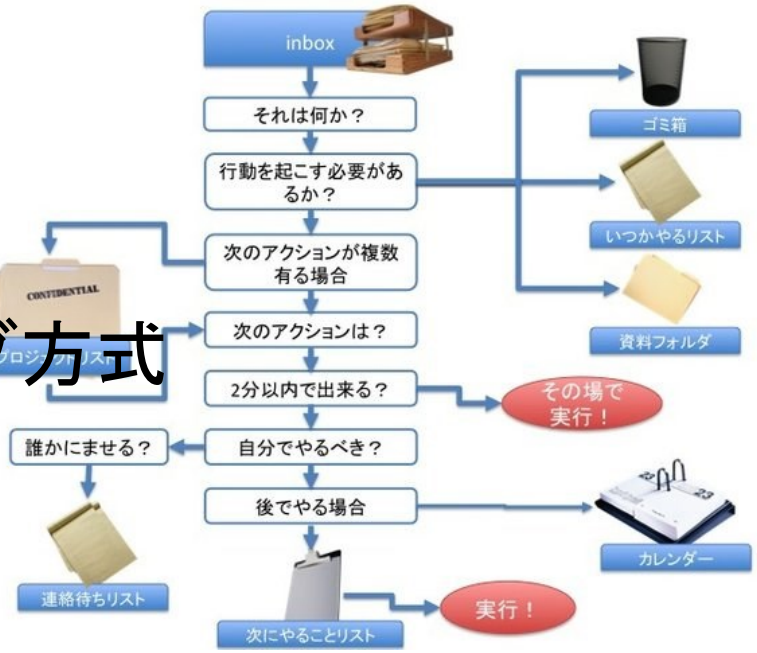
b. OS のAPIが提供されていない周辺機器は，ユーザプログラムから利用又は 制御することはできない。

c. アーキテクチャの異なる CPU 間でも，同じ OS とその API を使用することによって，プログラムの互換性を高め，移植時の工数を削減することが可能である。

d. 異なる OS 間でも API は共通であり，API だけを使用したプログラムであれば， 再コンパイルだけでほかの OS への移植が可能である。

2. タスク管理

- タスク管理の役割
- タスクの状態推移
- タスクのスケジューリング方式
- マルチプログラミング
- 割込み処理

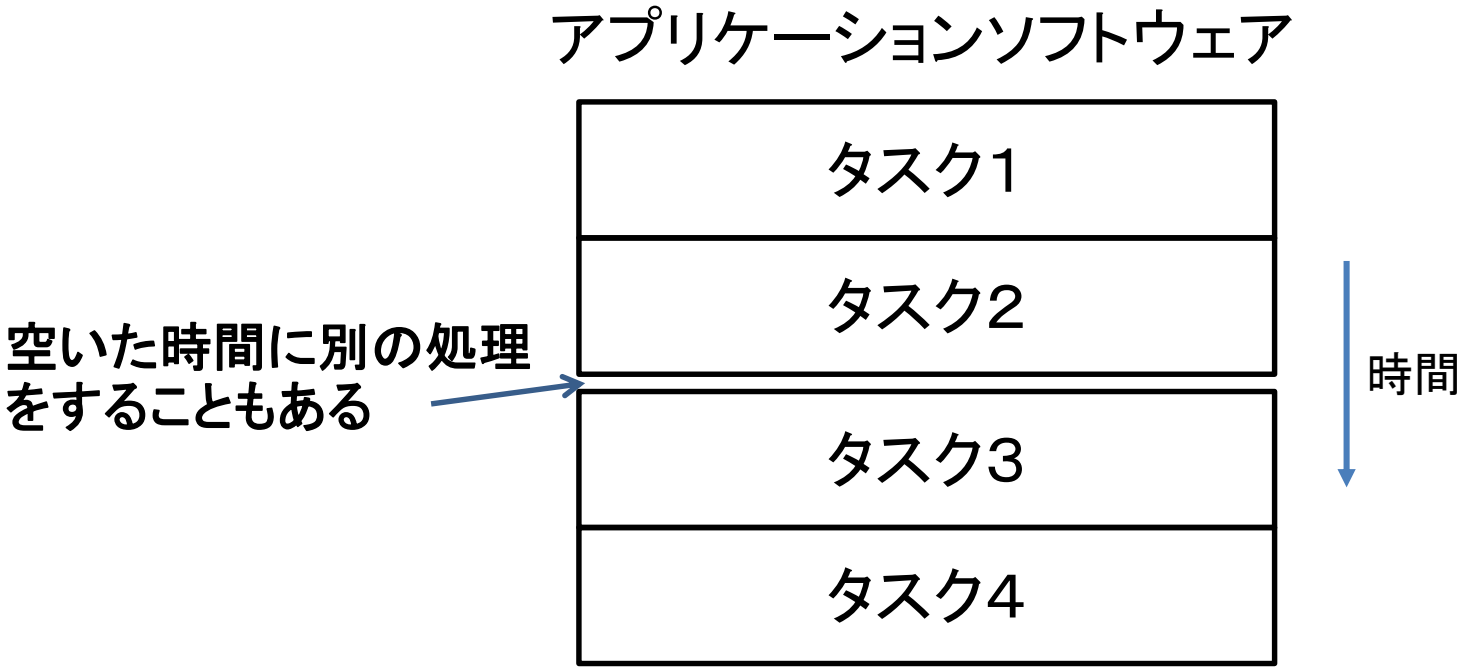


● タスク管理の役割

コンピュータから
見た仕事の単位

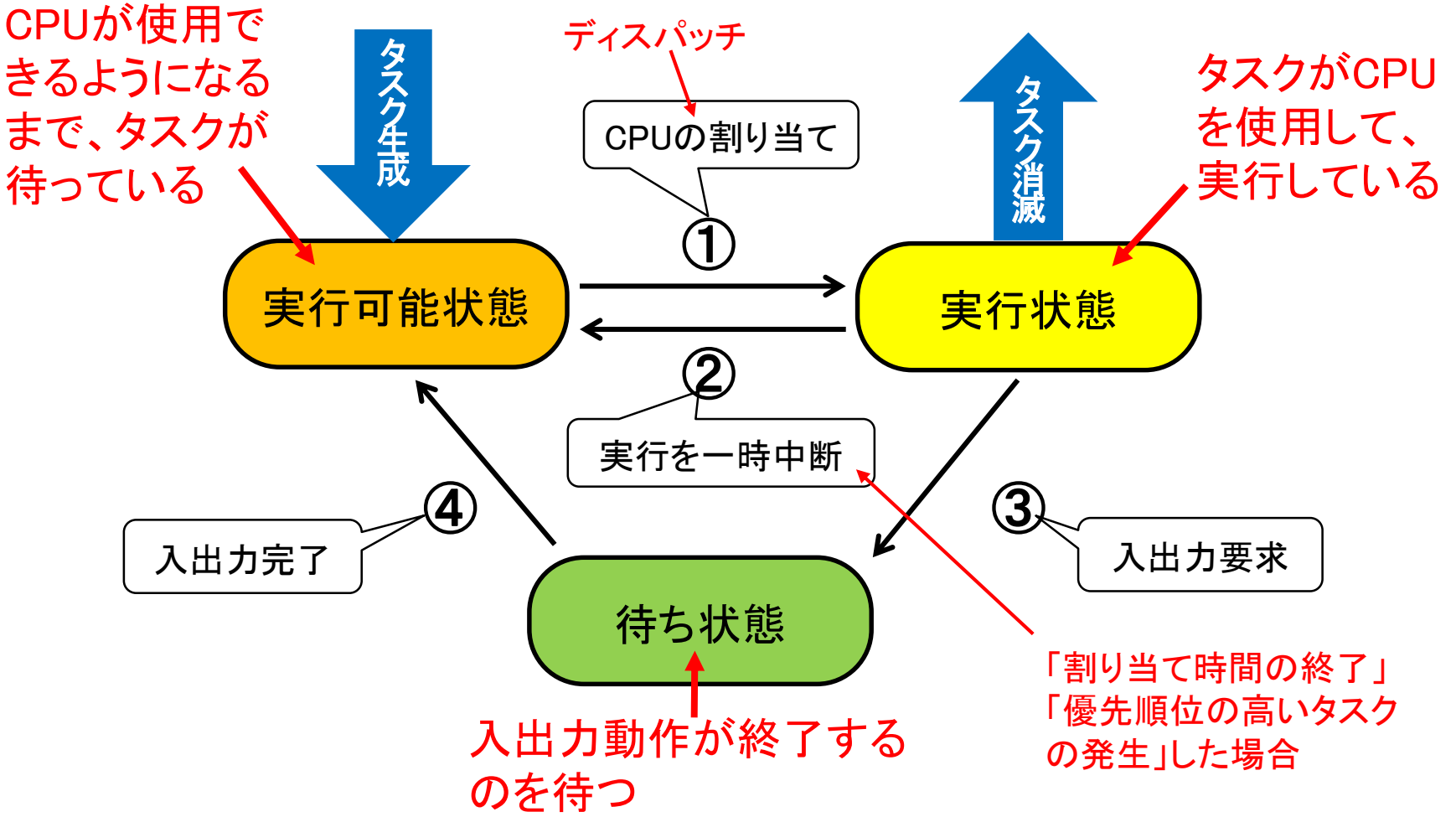
人間から見た仕事
の単位は、ジョブ

１つのアプリケーションソフトウェアを、いくつかのタスク（または、プロセス）に分けて効率よく動かす



● タスクの状態推移

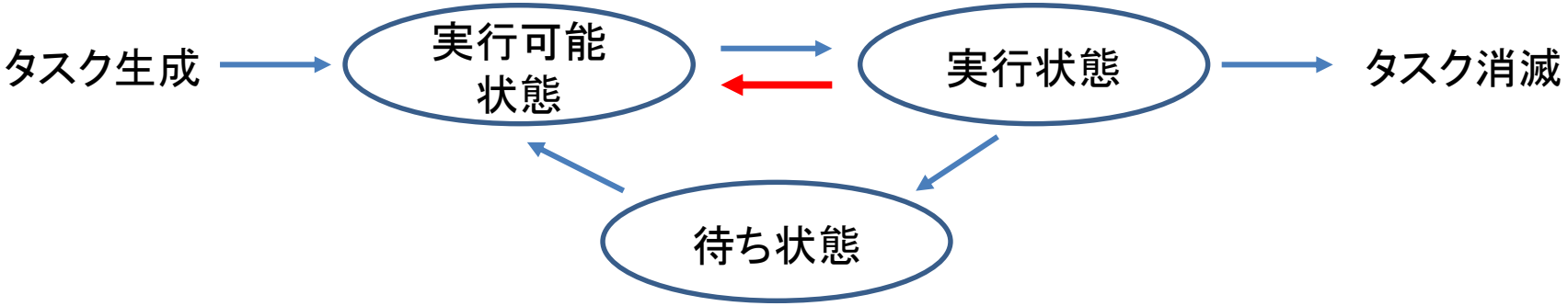
OSがCPUの使用権（CPUを使うための権利）を適切に割り当てて、タスクを効率よく実行させるための状態管理



【過去問題】

図は、マルチタスクで動作するコンピュータにおけるタスクの状態遷移を表したものである。**実行状態のタスクが、実行可能状態に遷移する**のは、どの場合か。

ア 自分より優先度の高いタスクが実行可能状態になった
イ タスクが生成された
ウ 入出力要求による処理が完了した
エ 入出力要求を行った



優先度の高いタスクにCPU使用権が移るため、CPU使用権を失ったタスクは実行可能状態に遷移します

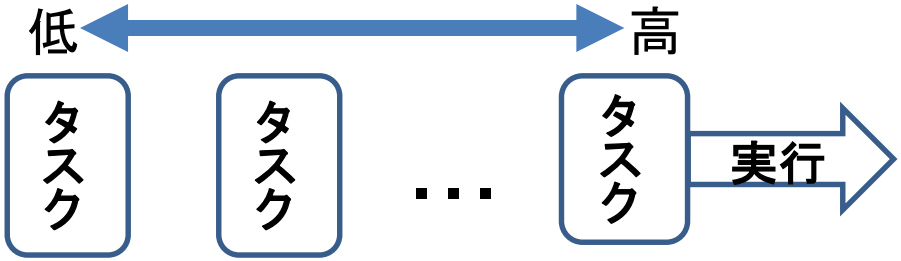
● タスクのスケージューリング方式
（タスクの実行する順番を決める方法）

- プリエンプティブ方式

  OSが強制的にタスクのスケジュールを決める

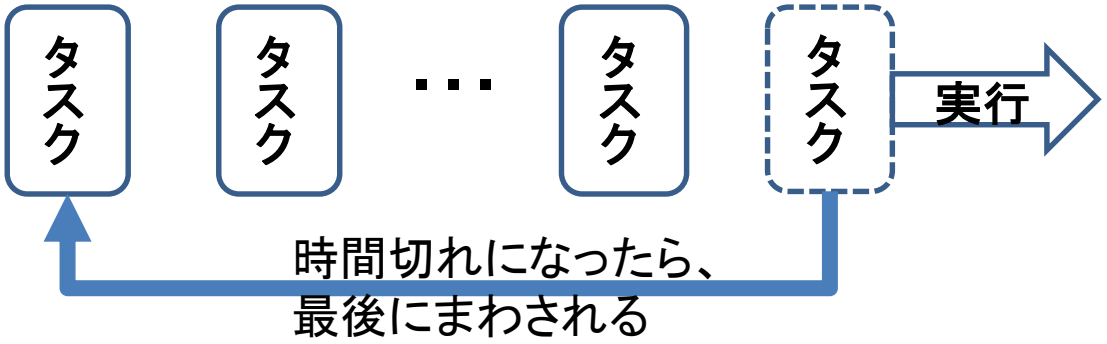
  ■ プライオリティ（優先度）方式

  優先順位の高いタスクから実行する



  ■ ラウンドロビン（タイムスライス）方式

  均等に時間を割り当てて、順番に実行する

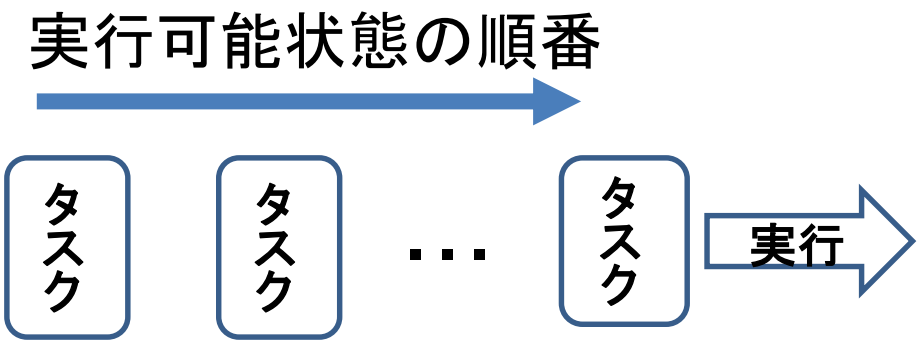


- ノンプリエンプティブ方式

  プログラムやタスクがスケジュールを決める

■ 到着順方式

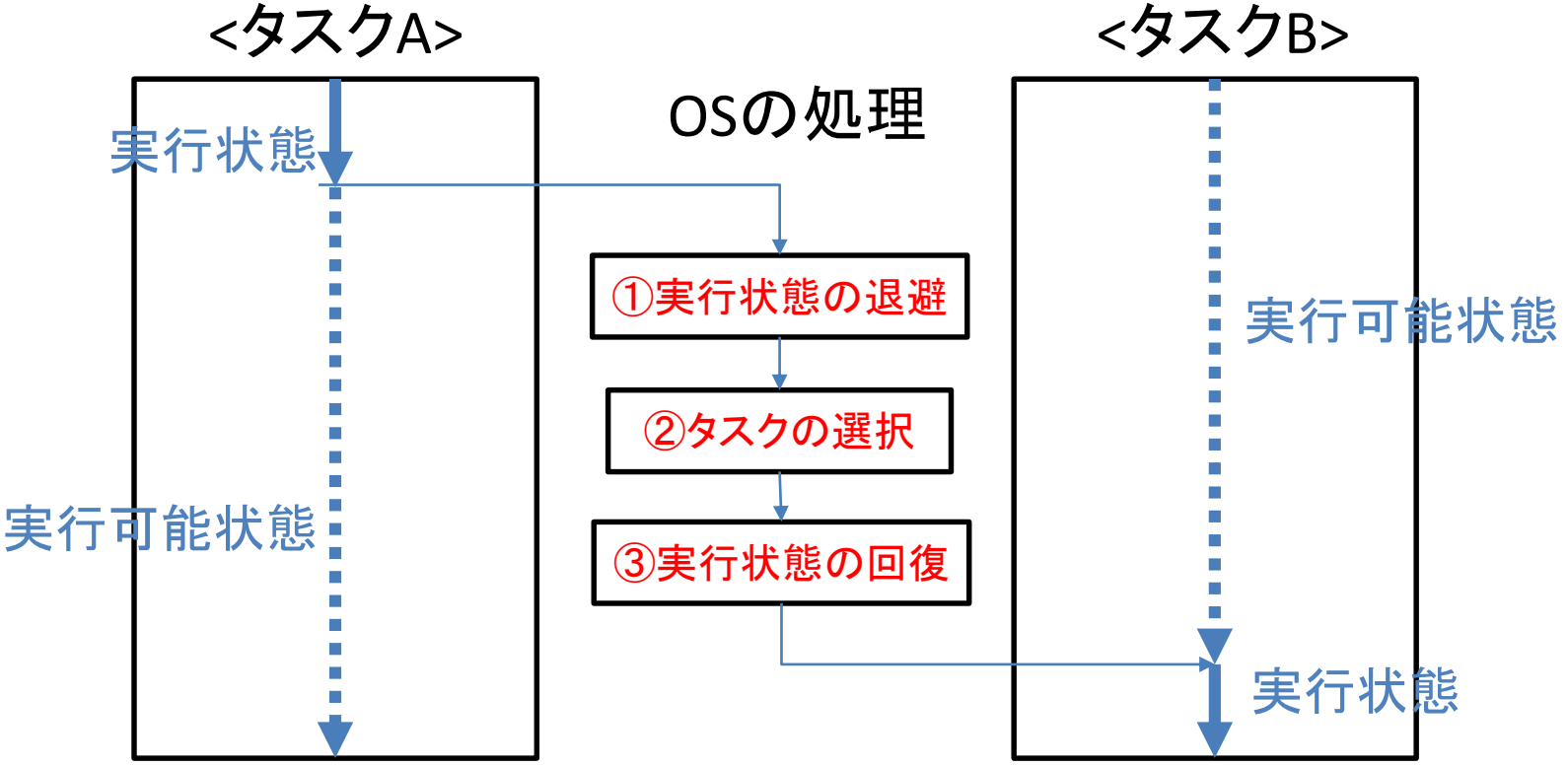
実行状態に移行した順番でタスクを実行する。実行終了（タスク消滅）までは、他のタスクは実行状態に移行しない。



● マルチプログラミング

- マルチタスク
- 多重プログラミング

複数のタスクを見かけ上、同時に実行するためには、OSが<タスクA>に対して①実行状態の退避をし、②タスクの選択を行い、<タスクB>に対して③実行の回復を実施する



● 割り込み処理

内部割込みと外部割込みを分類できるように

（実行中のタスクを中断して、別のタスクを実行する）

■ 内部割込み処理：コンピュータ内部の処理が原因で起こる

- プログラム割り込み
  プログラムのエラー（桁あふれ、ゼロ除算など）による割込み処理
- SVC（ Super Visor Call ）割込み
  アプリケーションがOSに対して、入出力要求することによる割込み処理

■ 外部割込み処理：外部要因による割込み処理

- マシンチェック割込み
  ハードウェアの誤動作や故障による割込み処理
- 入出力割込み
  入出力動作（キーボートやマウス、プリンタ）の終了や入出力動作が変化（紙切れや故障など）したときにおこる割込み処理
- タイマー割込み
  プログラムの実行割り当て時間を超えたことによる割込み処理

【過去問題】
プログラム割込みの原因となり得るものはどれか

ア　入出力動作が終了した

イ　ハードウェアが故障した

ウ　プログラムで演算結果があふれた（オーバフローした）

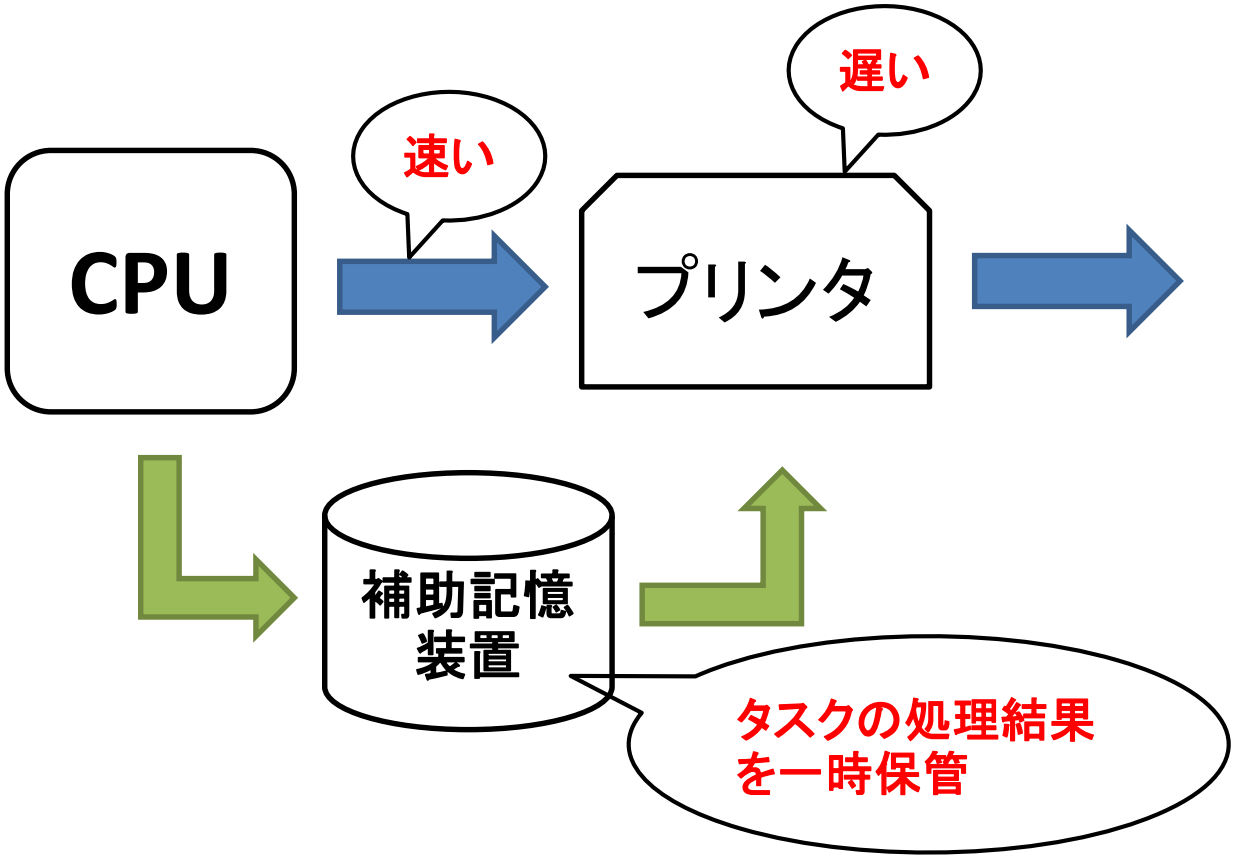
エ　プログラムの実行時間が設定時間を超過した。

アは、入出力割込み
イは、マシンチェック割込み
エは、タイマ割込み

これも
知っとこ

スプーリング

CPUがタスクを処理する速度に比べて、入出力装置（プリンタなど）の処理速度が遅いので、タスクの処理結果を一時的に補助記憶装置に保管する



【過去問題】

組込みリアルタイムOSで用いられる、優先度に基づくプリエンプティブなスケジューリングの利用方法として、適切なものはどれか

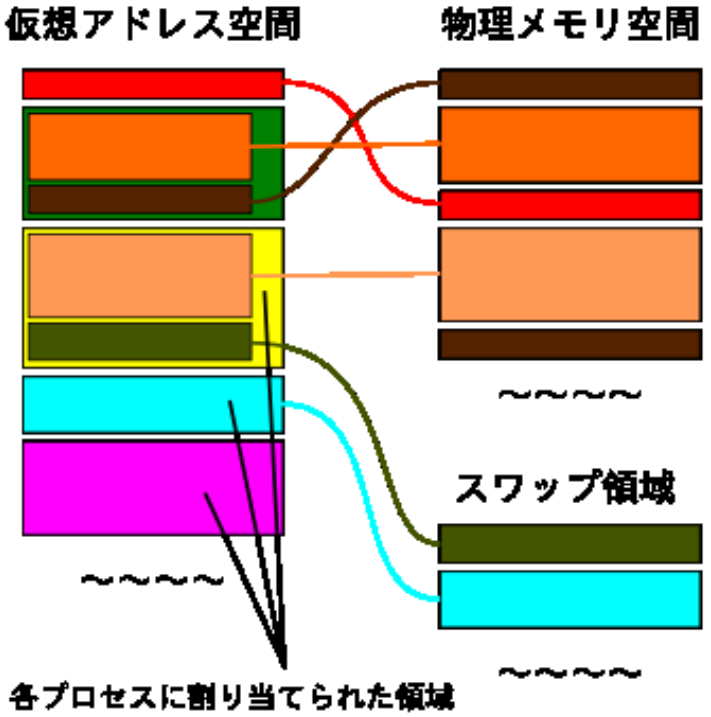
ア　各タスクの実行時間を均等配分する場合に利用される
イ　起動が早いタスクから順番に処理を行う場合に利用される
ウ　重要度及び緊急度に応じて処理を行う場合に利用される
エ　処理時間が短いタスクから順番に処理を行う場合に利用される

ア　ラウンドロビン方式
イ　到着順方式
エ　残余処理時間順方式

リアルタイム制御が必要とされる組込みリアルタイムOSでは、リアルタイム処理において目標時間内に処理を完了させることが最重要。重要度が高い、あるいはデッドラインが近いタスクに、より高い優先度を与えることで、実行するタスクの順序を強制的に入れ替えることができる優先度制御方式が適している。

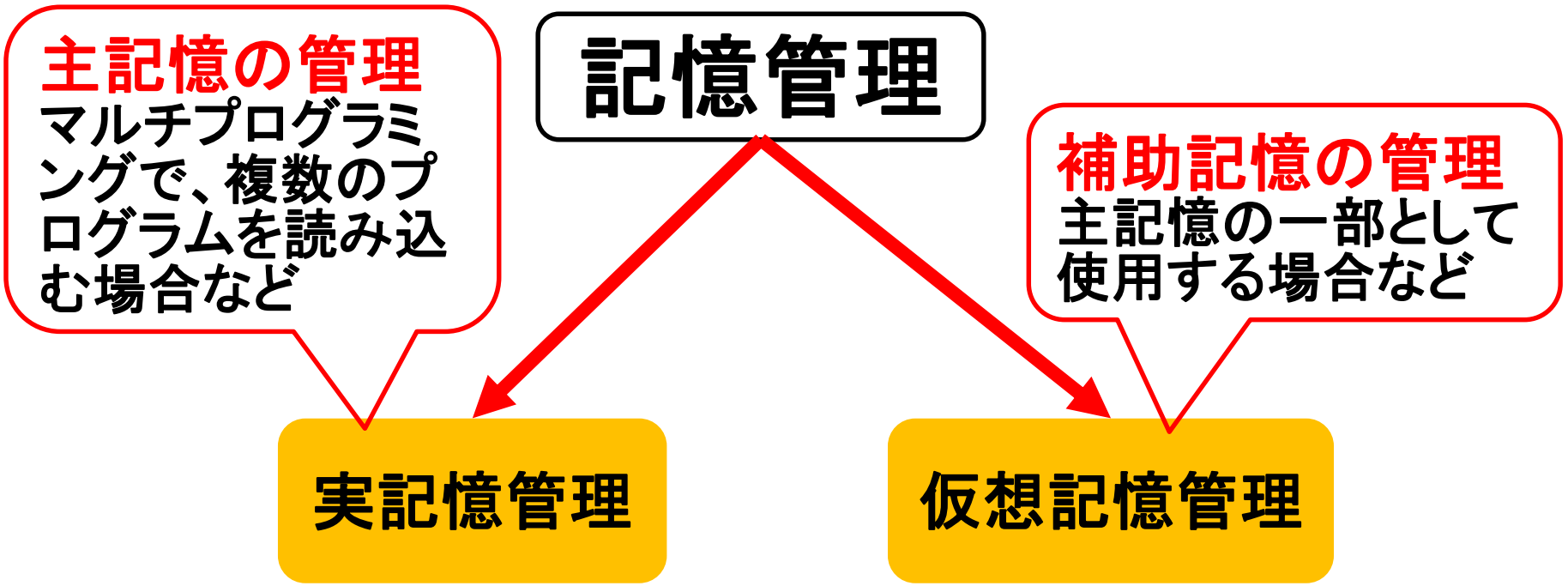
3. 記憶管理

- 記憶管理の役割
- 実記憶管理
- 仮想記憶管理



● 記憶管理の役割

限られた記憶（主記憶や補助記憶）装置の容量や機能を、効率的に使用する

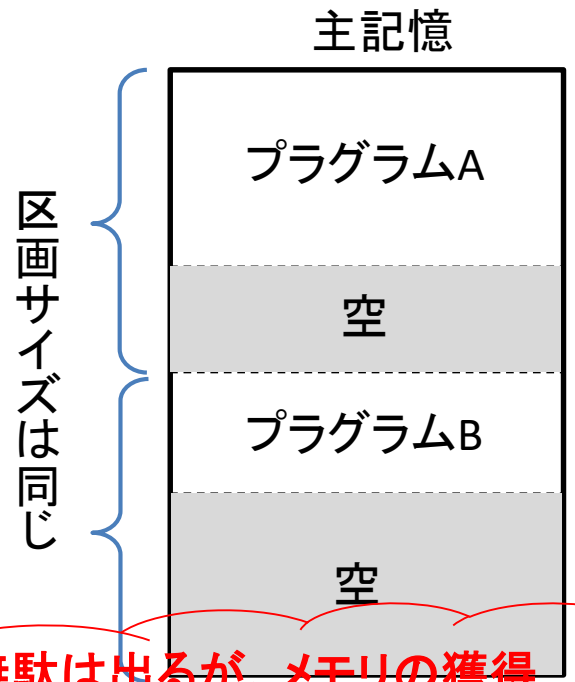


● 実記憶管理

- 区画方式

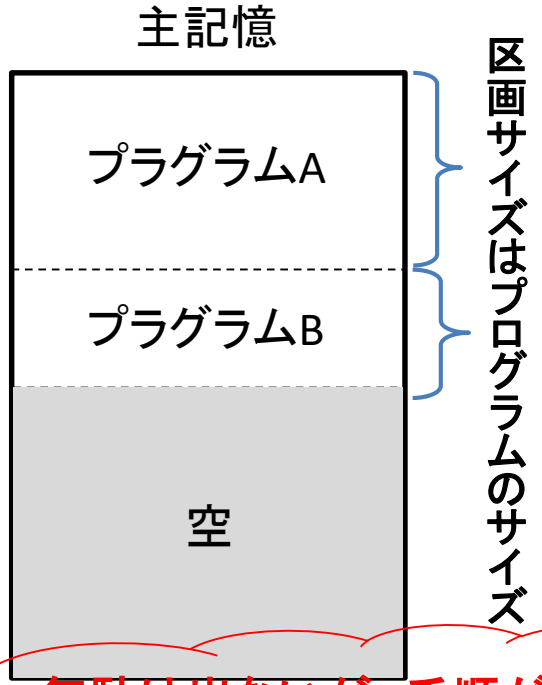
主記憶をいくつかの区画（領域）に区切って、各区画にプログラムを割り当てる

＜固定区画（固定長）方式＞



無駄は出るが、メモリの獲得や返却時間は一定なので、処理時間が速い

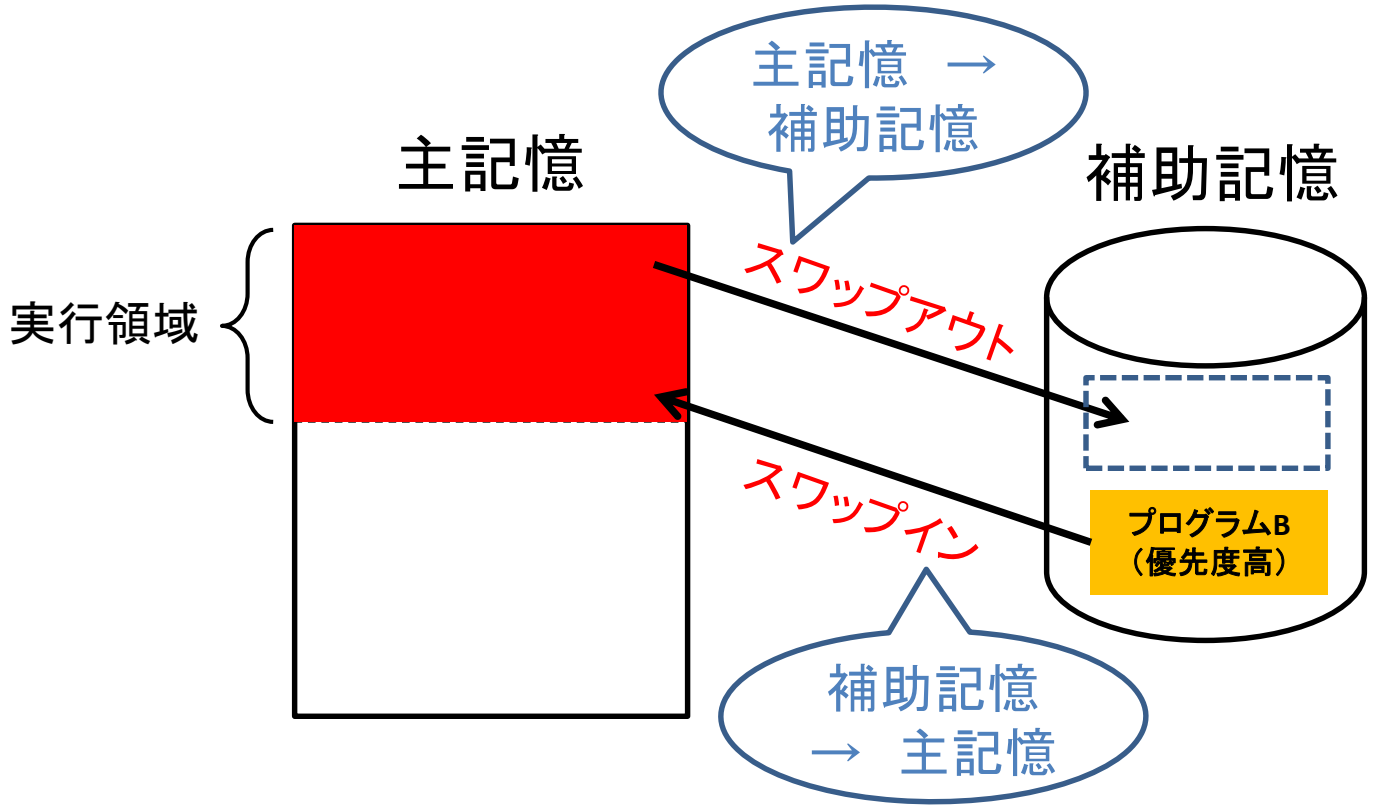
＜可変区画（可変長）方式＞



無駄は出ないが、手順が複雑で処理時間が遅い

- **スワッピング方式**

主記憶の容量が不足しているとき、優先度の低いプログラムAを一時的に中断して補助記憶に退避し、優先度の高いプログラムBを実行する



【過去問題】スワッピングに関する記述として、適切なものはどれか

ページング

ア　仮想記憶の構成単位であるページを、主記憶から補助記憶に書き出したり、補助記憶から主記憶に読み込んだりする

(イ)　システム資源全体の利用率の向上などのために、主記憶と補助記憶の間で、プロセスを単位として領域の内容を交換する

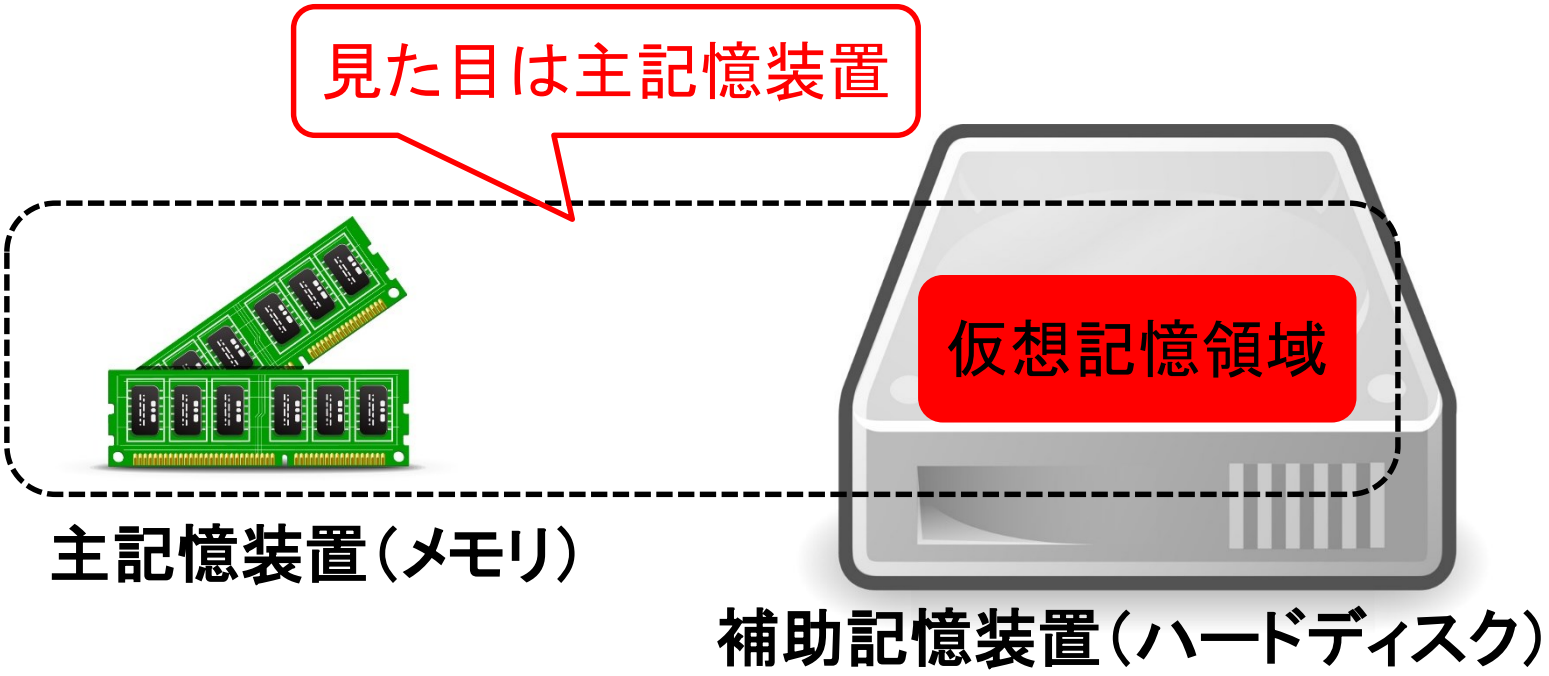
ウ　主記憶上に分散した空き領域を移動して、連続した大きな空き領域を生成する

エ　プログラムを機能ごとにモジュールに分割し、実行時に必要なモジュールだけをロードする

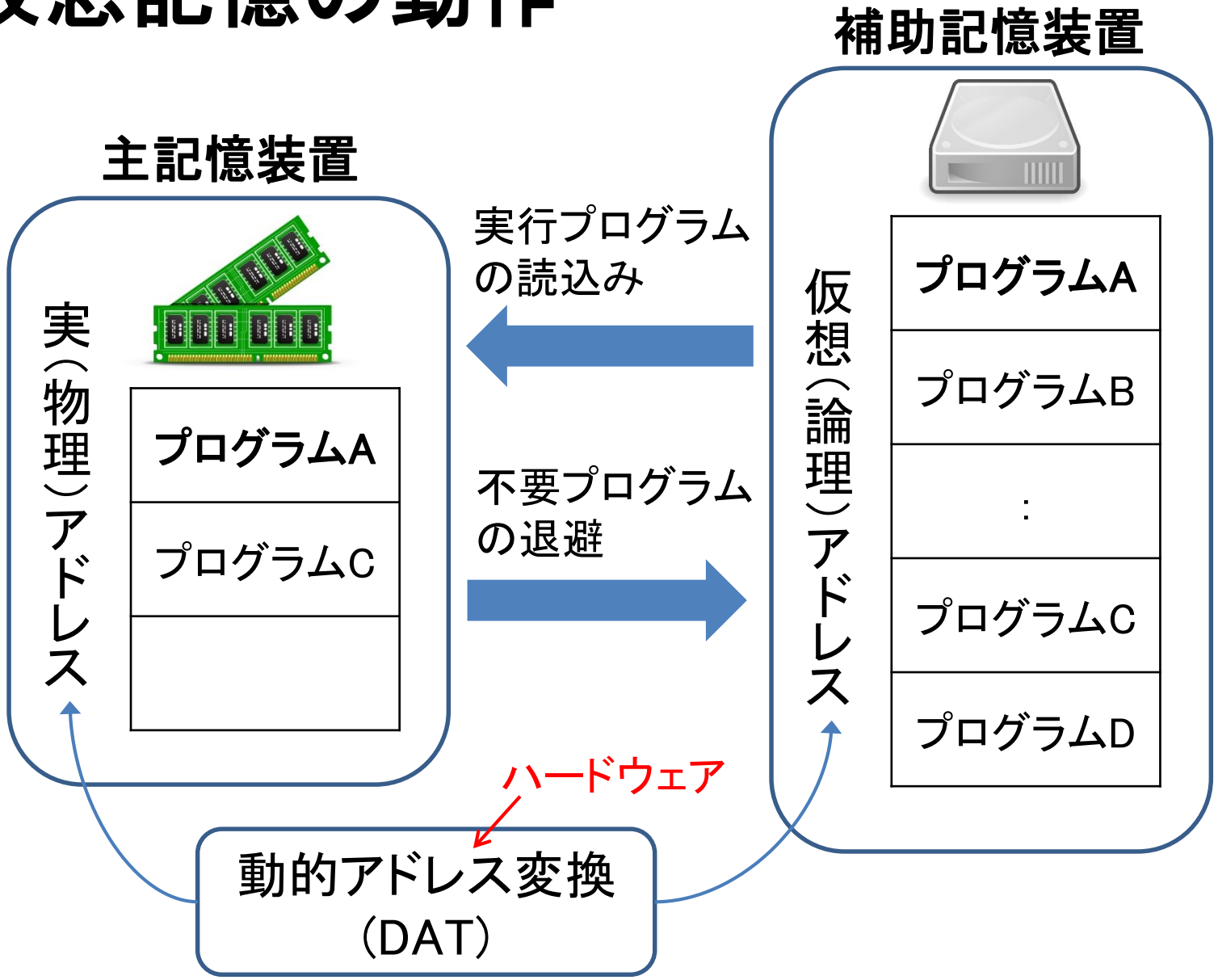
スワッピングとは、メインメモリに入りきらない一部のデータを一時的に外部記憶装置へ退避すること。使用中の主記憶の内容を補助記憶に退避することをスワップアウトといい、再開時に退避した内容を主記憶に再ロードすることをスワップインという。スワップインとスワップアウトを合わせて、スワッピングという。

● 仮想記憶管理

補助記憶装置の一部を、主記憶装置であるかのように見せかけて、実際の主記憶装置の容量よりも大きな記憶領域を仮想的に作り出す



• 仮想記憶の動作


補助記憶装置
主記憶装置
実行プログラム
の読込み
不要プログラム
の退避
実(物理)アドレス
仮想(論理)アドレス
プログラムA
プログラムC
プログラムA
プログラムB
:
プログラムC
プログラムD
ハードウェア
動的アドレス変換
(DAT)

● ページング方式

主記憶（実記憶）と補助記憶（仮想記憶）を、それぞれ決まった大きさのページと呼ばれる領域（通常は2～4kB）に分割して、対応づけて管理する。

ページテーブルを参照した結果、目的のページがなかったときには、ページフォールト(page fault)と呼ぶ割込みが発生して、目的のページを仮想記憶から実記憶へ読込（ページイン）んだり、不要なページを実記憶から仮想記憶へ追い出す（ページアウト）。

仮想記憶

ページ0
ページ1
ページ2
ページ3
:

ページングテーブル

ページ番号	開始アドレス
0	2000
1	2001
2	
3	0000

実記憶

	アドレス
ページ3	0000
:	
ページ0	2000
ページ1	2001
:	

ページフォールト（ページ2を実記憶へ読み込む）

● ページアウトするページを決める方法
（書き換え対象ページの決定方法）

仮想記憶

ページ0	プログラム0
ページ1	プログラム1
ページ2	プログラム2
ページ3	プログラム3
ページ4	プログラム4
	:

実記憶 

ページ0	
ページ1	
ページ2	

仮想記憶のページが、①ページ1 → ②ページ2 → ③ページ3 → ④ページ2 → ⑤ページ4 の順に参照すると、実記憶の状態は？

①<ページ1を参照するとき>

実記憶

ページ0	プログラム1
ページ1	
ページ2	

ページイン

②<ページ2を参照するとき>

実記憶

ページ0	プログラム1
ページ1	プログラム2
ページ2	

ページイン

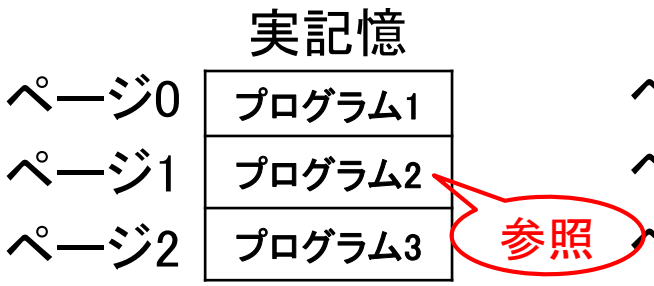
③<ページ3を参照するとき>

実記憶

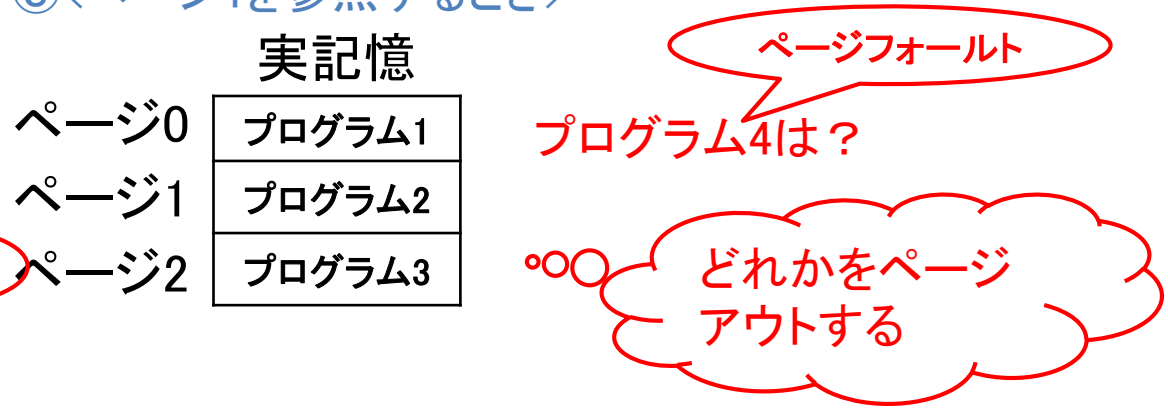
ページ0	プログラム1
ページ1	プログラム2
ページ2	プログラム3

ページイン

④〈ページ2を参照するとき〉

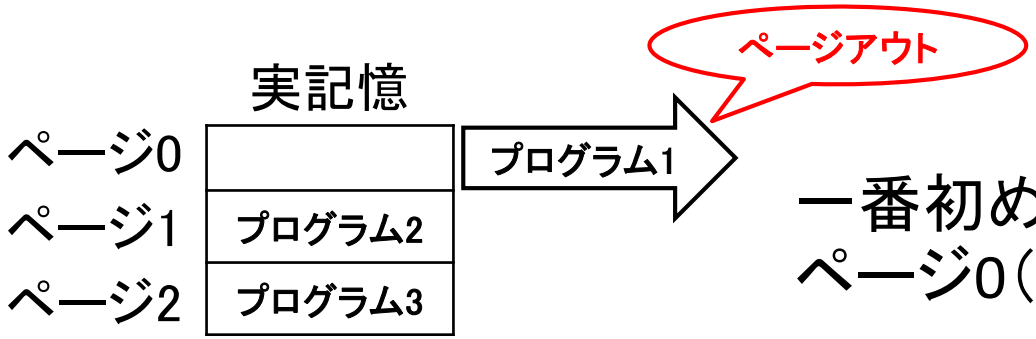


⑤〈ページ4を参照するとき〉



■ **FIFO(First-In First-Out)**

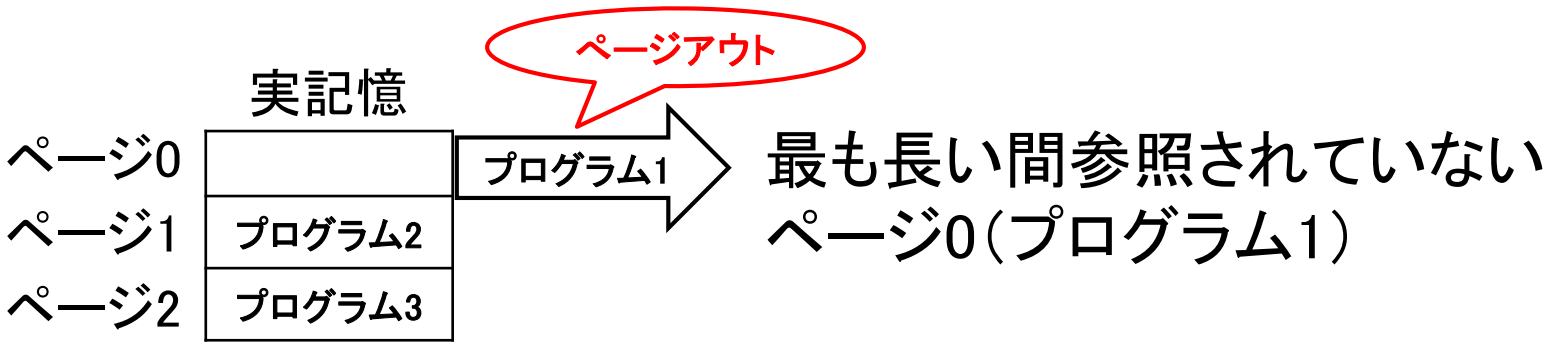
読み込まれた順にページアウトする



一番初めに読み込まれた
ページ0(プログラム1)

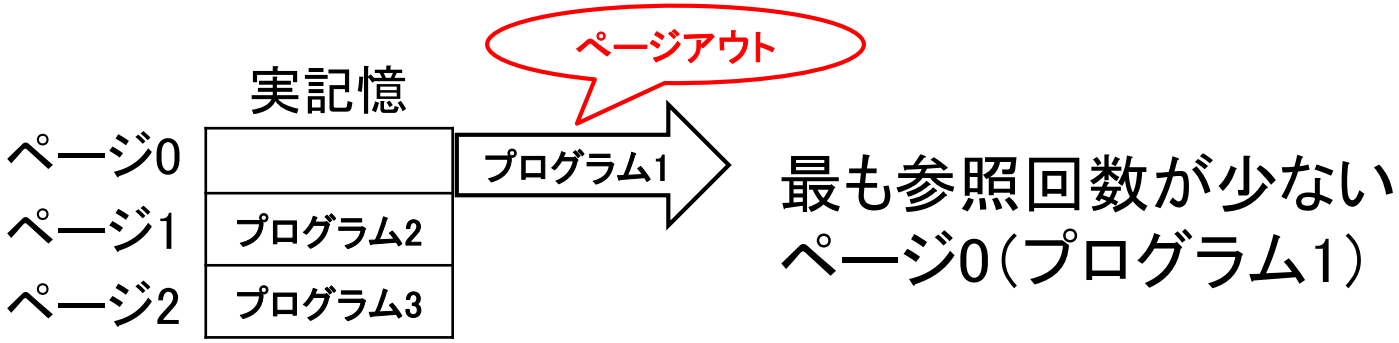
■ LRU(Least Recently Used)

（最後に）参照した時間が古い順にページアウトする



■ LFU(Least Frequently Used)

参照した回数が少ない順にページアウトする



【過去問題】

ページング方式の仮想記憶において，主記憶に存在しないページをアクセスした場合の処理や状態の順番として，適切なものはどれか。ここで，主記憶には現在，空きのページ枠はないものとする。

ア　置換え対象ページの決定→ページイン→ページフォールト→ページアウト

イ　置換え対象ページの決定→ページフォールト→ページアウト→ページイン

ウ　ページフォールト→置換え対象ページの決定→ページアウト→ページイン

エ　ページフォールト→置換え対象ページの決定→ページイン→ページアウト

＜ページング処理の順番＞
①主記憶上に必要なデータが存在しない状態の発生(ページフォールト)
②FIFOやLRUアルゴリズムを用いて置き換え対象のページの決定する
③置き換え対象のページを主記憶から仮想記憶に退避させる(ページアウト)
④実行に必要なデータをページ単位で主記憶に移す(ページイン)

【過去問題】

LRUアルゴリズムで、ページ置き換えの判断基準に用いられる項目はどれか。

ア　最後に参照した時刻

イ　最初に参照した時刻

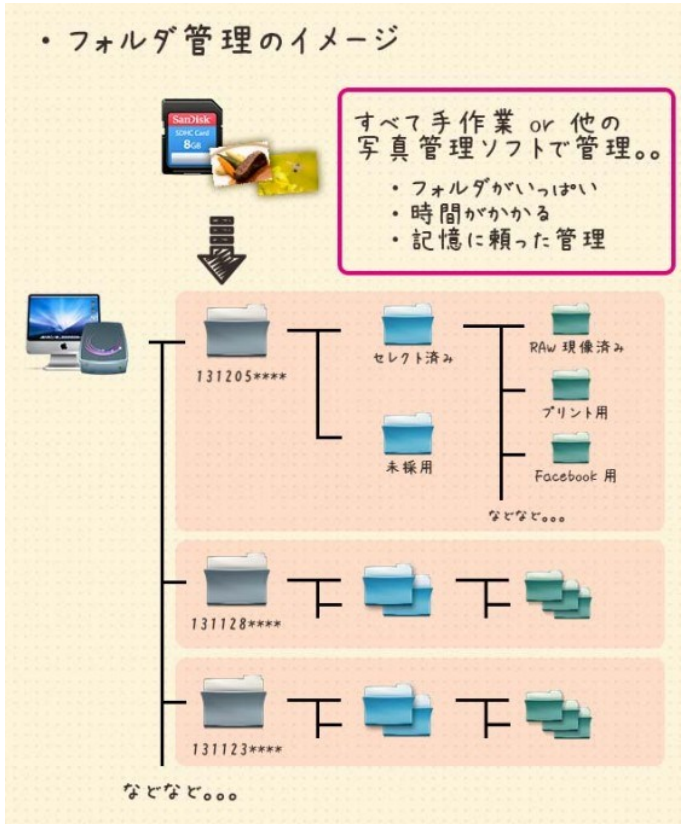
ウ　単位時間当たりの参照頻度

エ　累積の参照回数

LRU(Least Recently Used)は、
（最後に）参照した時間が古い順にページアウトする

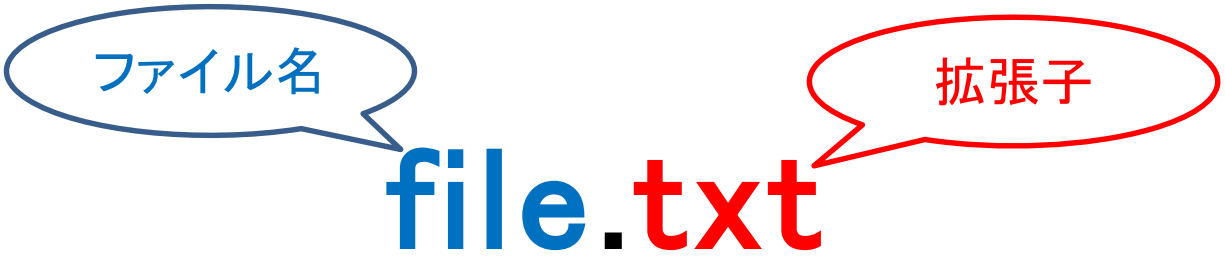
4. ファイル管理

- OSのファイル管理
- ディレクトリ
- パスの指定
- 絶対パス
- 相対パス



● OSのファイル管理

- OSは、文書,音声,画像,動画などのデータをファイルと呼ぶ単位で管理している
- ファイルは、フォルダと呼ぶ入れ物で整理する
- ファイルの種類は、ファイル名の後にドット「.」に続いて書かれた拡張子によって、OSが判断する



＜ファイルの種類と拡張子＞

■文書ファイル

- htmまたはhtml
  ホームページの文章を保存したファイル
- txt
  テキストファイルで、最も単純な文書ファイル
- doc
  Wordのファイル

■音楽ファイル

- MP3　:MPEG Audio Layer-3（圧縮）
  映像データ圧縮方式のMPEG-1で利用される音声圧縮方式の一つ

■静止画像ファイル

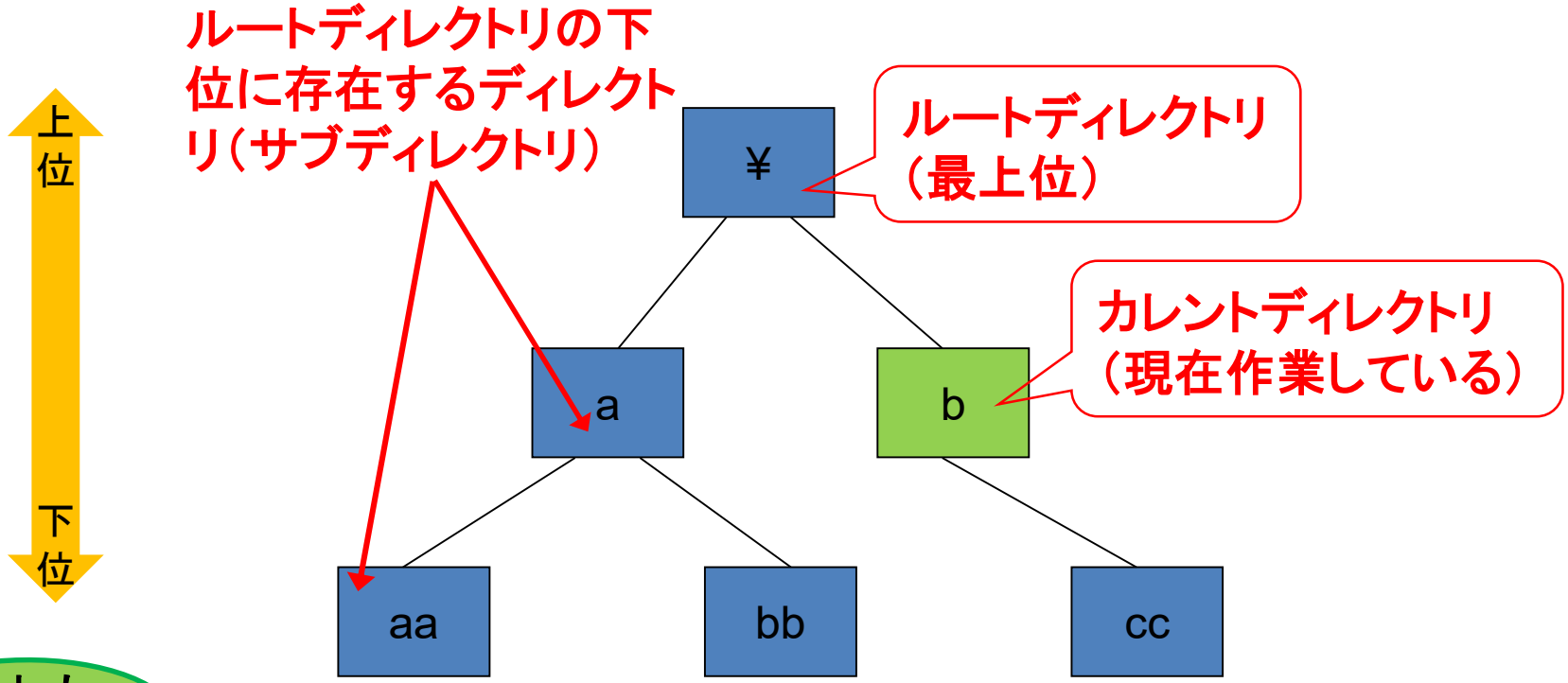
- jpgまたはjpeg　:Joint Photographic Experts Group
  主にインターネットのホームページやデジタルカメラなどで利用される
- gif　:Graphic Interchange format
  主にインターネットのホームページで利用される

■動画像ファイル

- mpgまたはmpeg　:Moving Picture Experts Group
  MPEG-1またはMPEG-2の動画ファイルの形式

● ディレクトリ

ファイルやディレクトリ（またはフォルダ）は、上（位）から下（位）に向かって段階状の構造（**階層構造**）になっている



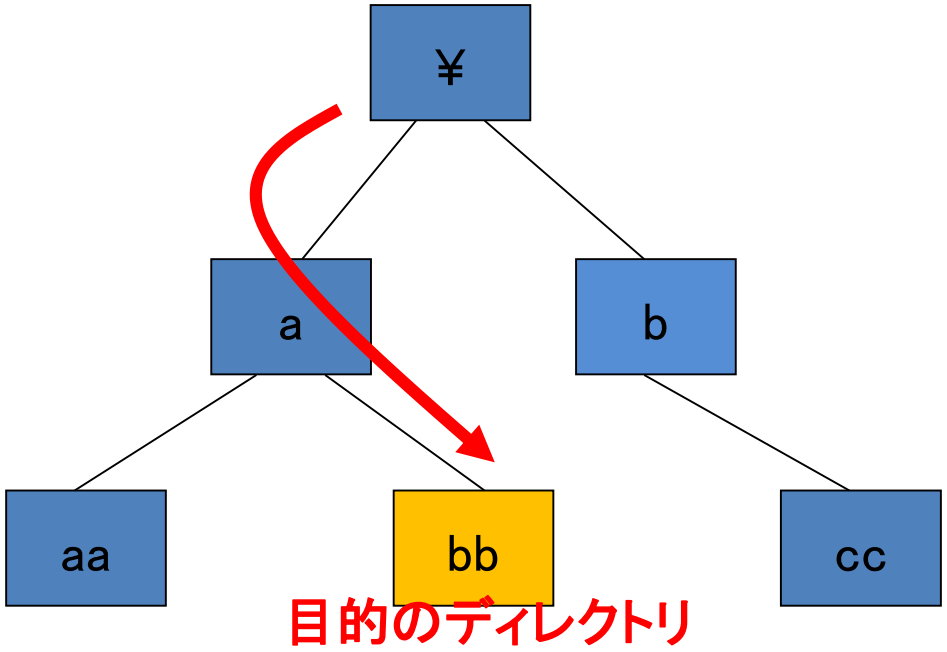
これも知っとこ

● ホームディレクトリ

マルチユーザのOSで、ユーザごとに用意されているユーザ専用ディレクトリ

● パス指定

目的のディレクトリやファイルまでの道順　→　パス（指定）

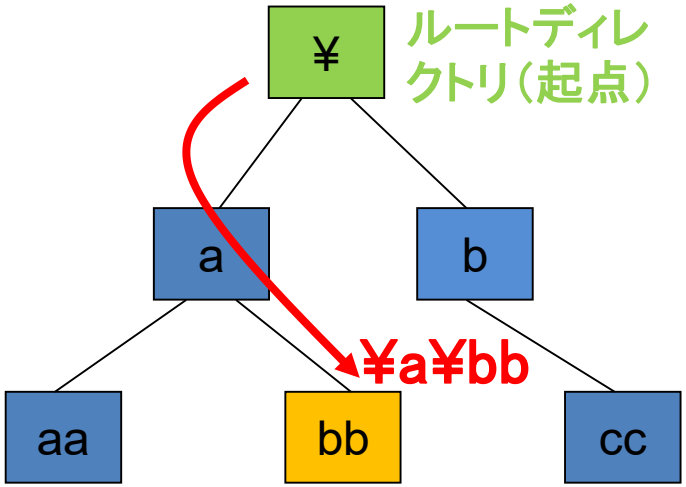


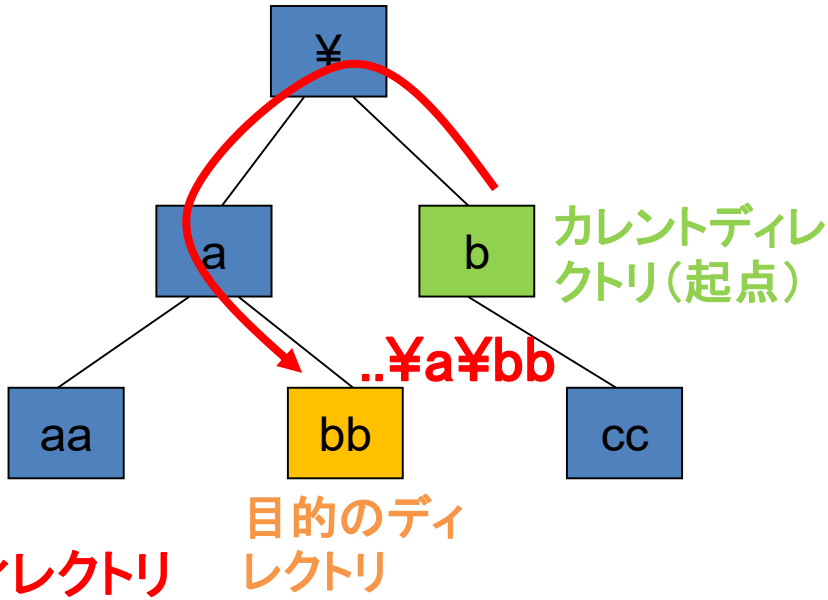
■ パス（道順）の出発点（ディレクトリ）の違いによって、
呼び名（種類）や記述方法が異なる

● 絶対パスと相対パス（パス指定の種類）

■ 絶対パス（指定）
ルートディレクトリ（起点）から目的のディレクトリまでの道順を指定する

■ 相対パス（指定）
カレントディレクトリ（起点）から目的のディレクトリまでの道順を指定する

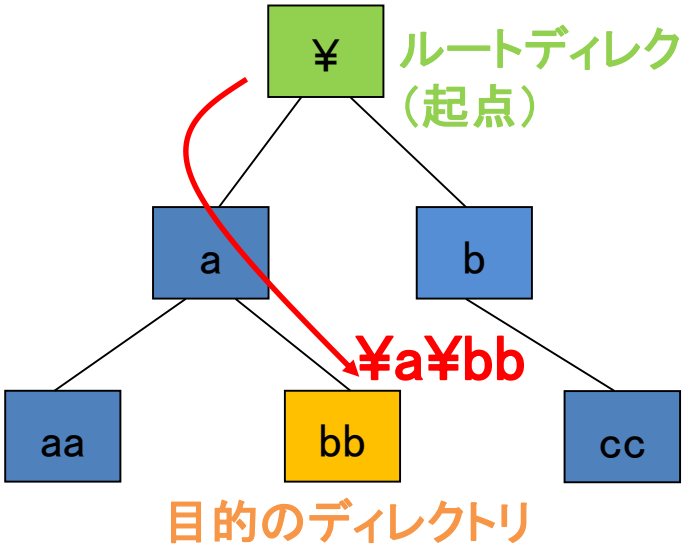


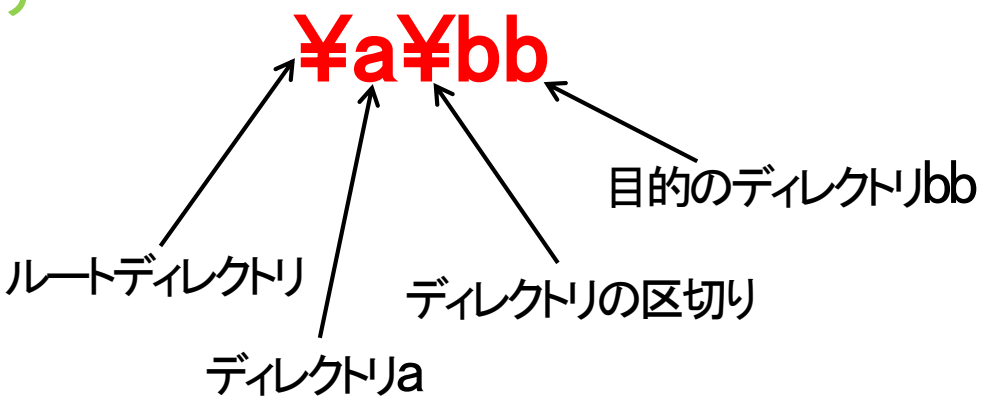


.. ⇒ 1つ上のディレクトリ

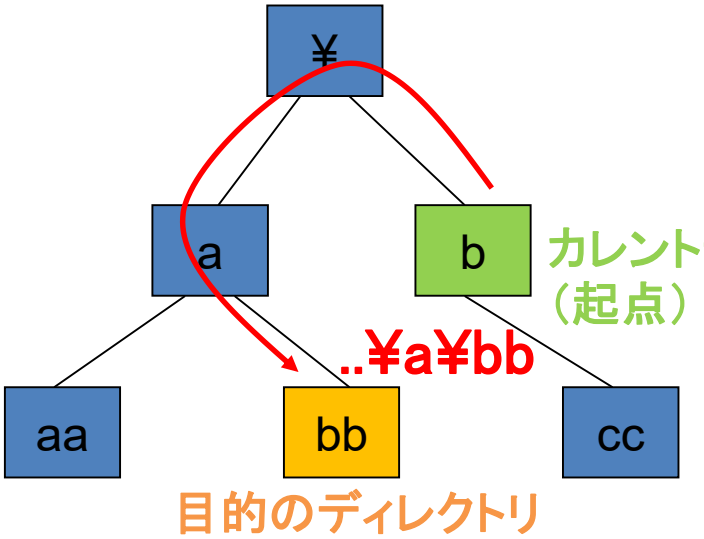
. ⇒ カレントディレクトリ

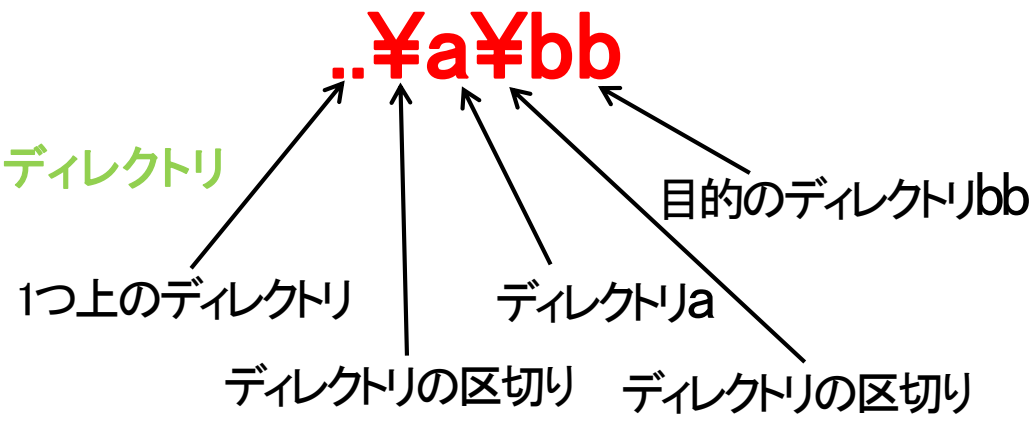
■ 絶対パス


¥
ルートディレクトリ
（起点）
a
b
¥a¥bb
aa
bb
cc
目的のディレクトリ


¥a¥bb
目的のディレクトリbb
ルートディレクトリ
ディレクトリの区切り
ディレクトリa

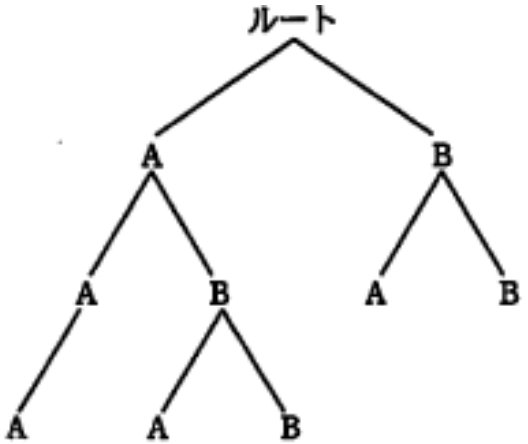
■ 相対パス


¥
a
b
カレントディレクトリ
（起点）
..¥a¥bb
aa
bb
cc
目的のディレクトリ


..¥a¥bb
目的のディレクトリbb
1つ上のディレクトリ
ディレクトリa
ディレクトリの区切り
ディレクトリの区切り

【過去問題】

A，Bというディレクトリ名をもつ複数個のデイレクトリが図の構造で管理されている



カレントディレクトリを　¥A¥B　→　..　→　..¥B　→　.¥A　の順に移動させた場合，最終的なカレントディレクトリはどこか。ここで，ディレクトリの指定方法は次のとおりとする。

〔ディレクトリの指定方法〕

（1）ディレクトリは，″ディレクトリ名¥…¥ディレクトリ名″のように，経路上のディレクトリを順に″¥″で区切って並べた後に″¥″とディレクトリ名を指定する。

（2）カレントディレクトリは″.″で表す。

（3）1階層上のデイレクトリは″..″で表する。

（4）始まりが″¥″のときは，左端にルートディレクトリが省略されているものとする。

（5）始まりが″¥″，″.″，″..″のいずれでもないときは，左端にカレントディレクトリ配下であることを表す″.¥″が省略されているものとする。

ア ¥A　　イ ¥A¥A　　ウ ¥A¥B¥A　　エ ¥B¥A

令和元年度 秋期
基本情報処理技術者試験問題・解答（ソフトウェア）

【問18】
　優先度に基づくプリエンプティブなスケジューリングを行うリアルタイムOSで、二つのタスクA,Bをスケジューリングする。Aの方がBより優先度が高い場合にリアルタイムOSが行う動作のうち、適切なものはどれか。

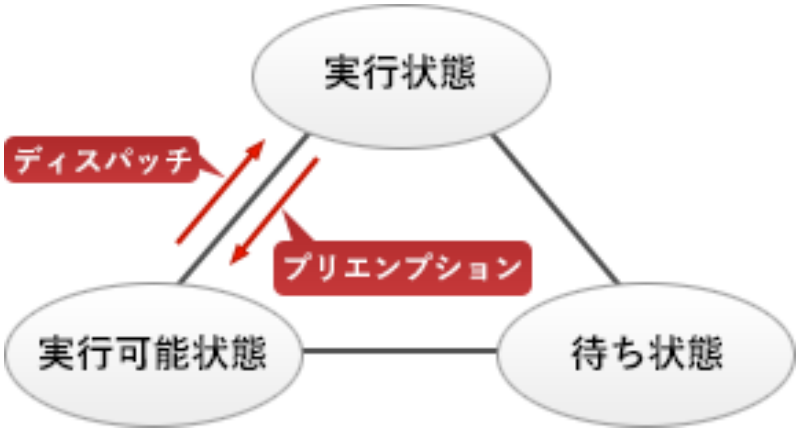
ア　Aの実行中にBに起動がかかると、Aを実行可能状態にしてBを実行する。
Aの実行が継続する

イ　Aの実行中にBに起動がかかると、Aを待ち状態にしてBを実行する。
Aの実行が継続する

(ウ) Bの実行中にAに起動がかかると、Bを実行可能状態にしてAを実行する。

エ　Bの実行中にAに起動がかかると、Bを待ち状態にしてAを実行する。
Bは実行可能状態に移される

プリエンプティブなタスクスケジューリングでは、CPUの使用をOSが管理し、タスクを動的に切り替えながら実行する。
OSの動作は、A・Bの優先度及びA・Bの状態によって変わる。



＜Aの実行中にBに起動がかかる＞
タスクの優先度は「A＞B」なので、Bは実行可能状態となり、Aの実行が継続する。待ち状態は、入出力待ちなどですぐにCPU処理に移れないタスクが遷移する状態なので、Bは待ち状態ではなく実行可能状態に移される。

＜Bの実行中にAに起動がかかる＞
タスクの優先度は「A＞B」なので、Bは実行可能状態に移され、AにCPU使用権が与えられる。

平成31年度 春期
基本情報処理技術者試験問題・解答（ソフトウェア）

問16　タスクのディスパッチの説明として，適切なものはどれか。

ア　各タスクの実行順序を決定すること

イ　実行可能なタスクに対してプロセッサの使用権を割り当てること

ウ　タスクの実行に必要な情報であるコンテキストのこと

エ　一つのプロセッサで複数のタスクを同時に実行しているかのように見せかける機能のこと

ディスパッチとは、OSによるタスク管理の制御機能の1つで、実行可能状態のタスクの中から優先度順などによって次に実行すべきタスクを選択して、CPUの使用権を割り当てること。

ディスパッチを行うプログラムをディスパッチャ(Dispatcher)と呼ぶ。

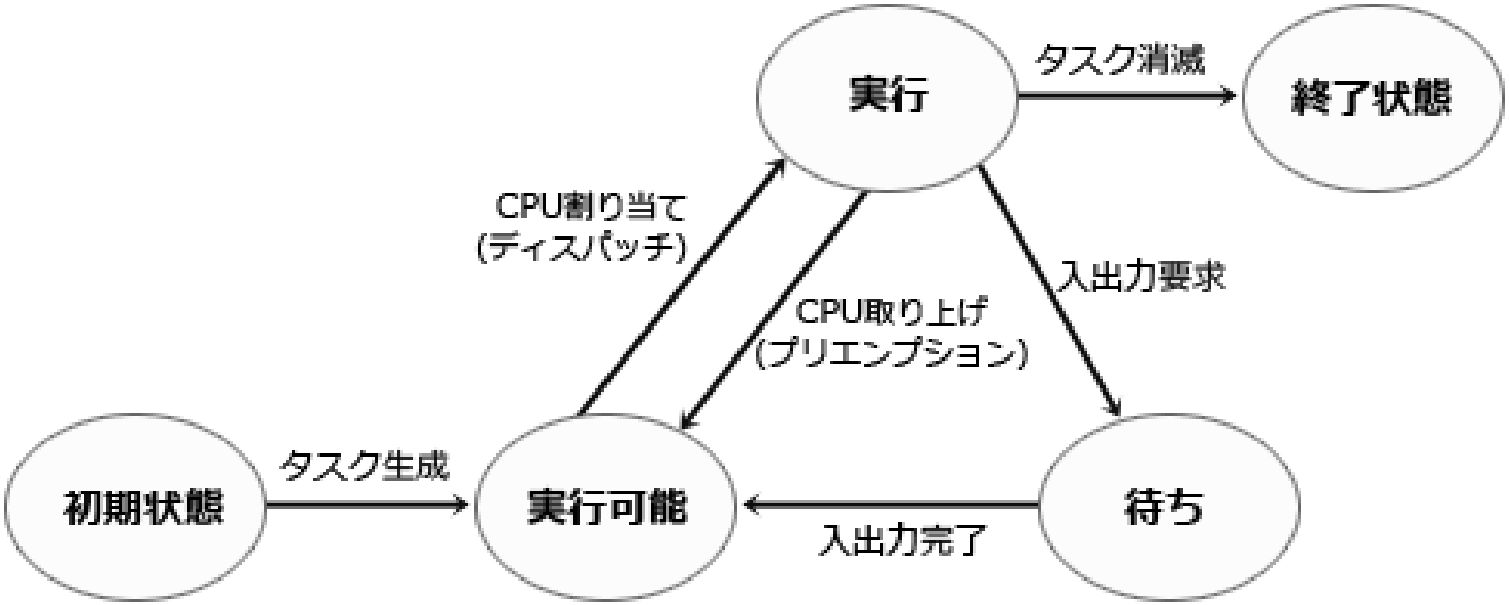


図　状態遷移図で表したタスクの遷移