Přidělování zdrojů (prostředků)

- Proces potřebuje **zdroje** (**prostředky**)
  - **hardware** (I/O zařízení, paměť)
  - **software** (data, programy)

- **Klasifikace zdrojů** (z hlediska multitaskingového režimu)
  - **Násobně použitelné** (= sdílitelné, = **reentrantní**) zdroje lze přidělit jinému procesu před tím, než je proces, kterému byly přiděleny, dokončen. (*základní jednotka, disk jako celek, myš, data pro čtení*)
  - **Opakovaně použitelné** (= **reusabilní**) zdroje lze přidělit jinému procesu až poté, co proces, kterému byly přiděleny dosud, byl ukončen – tvoří kritickou oblast. (*tiskárna, pracovní oblast v paměti a na disku, většina programů, pokud obsahují pracovní položky*)
  - Zdroje **na jedno použití** (= **nereusabilní**) nelze přidělit znova. (papír do tiskárny, ale také třeba špatně vytvořený program – opomenuté nulování střadače)

  - **Práce s reusabilním, ale ne reentrantním zdrojem tvoří kritickou oblast**. Nutno řídit přístup k ní.

Kritická oblast

- **Kritická oblast (sekce)** = část kódu programu, která pracuje se sdílenými prostředky (zdroji) a kterou je potřeba vymezit v každém paralelním procesu a přístup do ní řídit (synchronizovat).
  - V kritické oblasti může být současně nejvýše jeden proces. Je-li kritická oblast obsazena, musí být proces převeden do stavu „čeká na prostředek“, do doby než oblast stávající proces opustí.
  - Každý proces, který vstoupí do kritické oblasti ji musí v konečném čase opustit.
  - Každý proces, který čeká na vstup do kritické oblasti se v konečném čase musí vstupu do ní dočkat (nesmí dojít k vyhladovění procesu).

Prostředky pro řízení přístupu do kritické oblasti

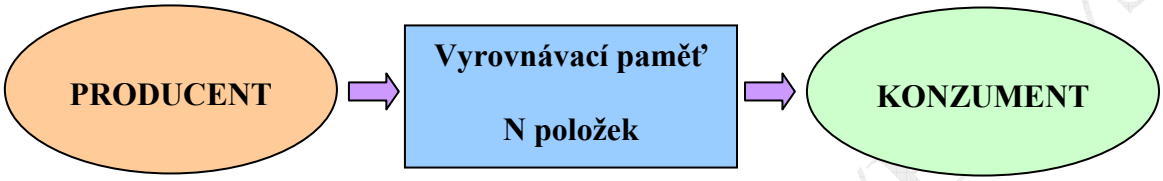
- **Zámek (binární semafor)** je sdílená proměnná (***S***) s dvěmi možnými polohami (1 = otevřen, 0 = uzavřen).
  - Proces před vstupem do kritické oblasti testuje polohu semaforu.
  - Je-li otevřen, pokračuje a semafor ihned uzavře (**potřeba speciální instrukce TST (TSTL)** – „*test and set (lock)*“). Mezi testem a následným uzavřením semaforu nesmí dojít k přerušení, proto je nutné obě akce provést v jediné instrukci.
  - Je-li semafor zavřen, přejde proces do stavu čeká.
  - Po opuštění kritické oblasti proces semafor otevře.

  - **Operace nad binárním semaforem *S***:
    - **wait(*S*)** - je-li *S* otevřen, **ihned** jej zavře a pokračuje, je-li zavřen, zavolá operační systém a ten zařadí proces do stavu ČEKAJÍCÍ
    - **signal(*S*)** - otevře semafor a volá operační systém, který rozhodne o zařazení vhodného procesu (podle zvolené disciplíny obsluhy)

- **Počítaný semafor** je sdílená proměnná typu INTEGER nebo BYTE s více (*N*) možnými hodnotami. ***S* = 1 2, ..., *N*** - semafor otevřen, ***S* = 0** – semafor zavřen.

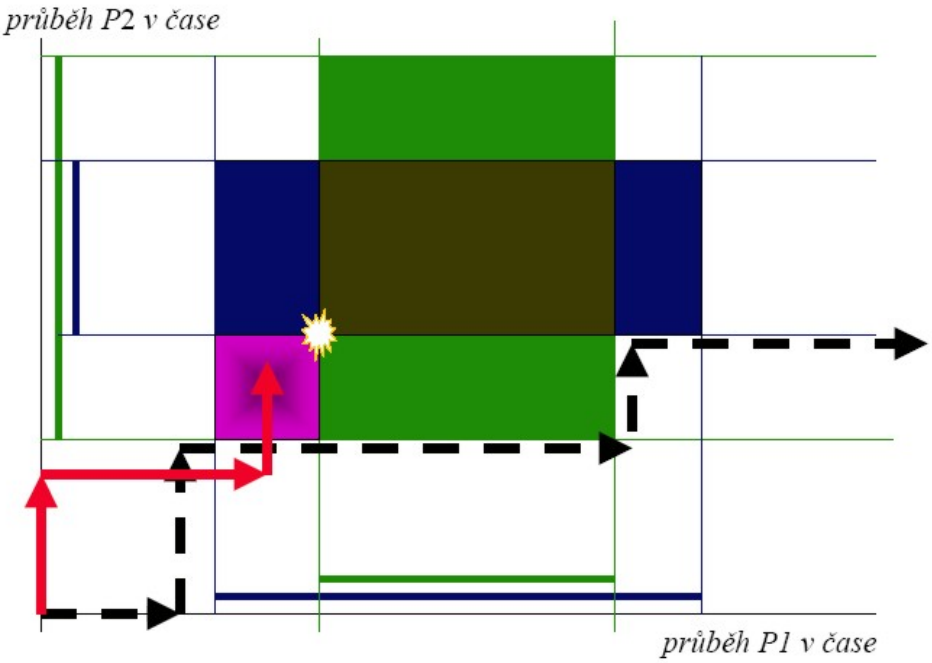
  - **Operace nad počítaným semaforem *S***:
    - **wait(*S*)**: (při vstupu): Je-li ***S* > 0**, pak **ihned *S* = *S* - 1** a pokračuje se, je-li ***S* = 0**, volá se operační systém, který proces odstaví
    - **signal(*S*)**: (při opuštění) ***S* = *S* + 1** a pokud nějaký proces čeká, bude zařazen.

  - **Příklad:** Komunikace procesů **PRODUCENT – KONZUMENT**



    - **Semafory:** PRAZDNYCH na začátku N, PLNYCH na začátku 0
    - Lze realizovat i pomocí počtu vyslaných a počtu přijatých signálů a hlídání rozdílu **0 < PočetVyslaných - PocetPrijatych < N**

- **Uváznutí** (zamrznutí, zaseknutí, smrtelné objetí, smrtelný zámek, *dead lock*) = stav, kdy všechny spuštěné procesy jsou ve stavu „čeká“ na událost (uvolnění zdroje), která nenastane.
  - **Příklad:** Proces **P1** potřebuje zdroj **Z1** a poté **Z2**, proces **P2** potřebuje **Z2** a poté **Z1**



Nebezpečná oblast. Vstup do ní ⇒ budoucí uváznutí.

„Úspěšný“ průběh: *P*1 obsadí *Z*1, obsadí *Z*2, *P*2 na uvolnění *Z*2 počká, poté uvolní *Z*2, P2 obsadí *Z*2 a počká až P1 *Z*1 uvolní ...

Zpracování nutně uvázne. P1 obsadí *Z*1 a drží jej. *P*2 obsadí *Z*2 a drží jej. *P*1 čeká na *Z*2, *P*2 čeká na *Z*1 ⇒ systém vždy uvázne.

Jak zabránit uváznutí

- Následné řešení : **DETEKCE A ZOTAVENÍ**
  - V určitých časových intervalech je (na základě přerušení od hodin) spouštěn systémový program („démon“), který kontroluje, zda nejsou všechny spuštěné uživatelské procesy ve stavu „čeká“. Pokud ano, některý zruší (nebo vyzve obsluhu ke zrušení).
  - **Nevýhoda**: Důsledek je porucha zpracování. Zrušený proces neproběhne.

- **PREVENCE** *(vždy lepší než následná terapie)*

  - **Úplné přidělování:** Procesu jsou přiděleny všechny reusabilní prostředky ihned po spuštění, i když je bude potřebovat až později.
    - HODNOCENÍ: **Plýtvá se prostředky**.

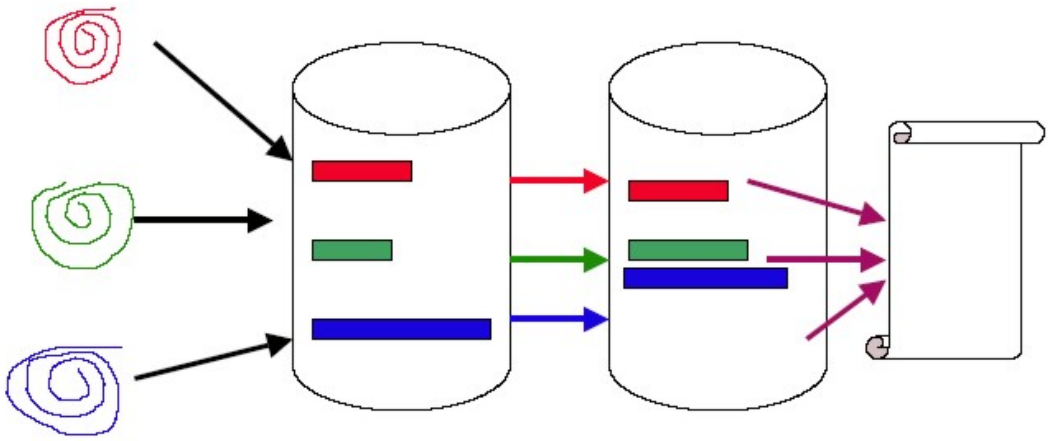
  - **Uspořádané přidělování:** Zdroje seřazeny. Současně s přidělením prostředku se přidělí i všechny, které jsou v tomto pořadí před ním.
    - HODNOCENÍ: **Rovněž plýtvání**, i když někdy o něco (málo) menší.

  - **Řízené přidělování – Algoritmus bankéře:** Při každé žádosti o přidělení prostředku se zkoumá nejen, zda je tento prostředek k dispozici, ale také, **zda existuje další bezpečná strategie dalšího přidělování, která umožní všechny procesy dokončit.**
    - Bankéř má základní kapitál a několik zákazníků. Jeho kapitál postačí pokrýt potřeby každého ze zákazníků jednotlivě, ne však všech současně. Zákazníci žádají o uvolnění prostředků na své investice **postupně**. Půjčené prostředky vrátí až po dokončení **celé** investice.
    - **Příklad:** Základní kapitál banky 20mil. 3 zákazníci. Každý potřebuje na celou investici 10mil.
    - **Průběh:** Prvý žádá na stavbu 6mil. LZE VYHOVĚT. Bance zbude 14mil. Druhý žádá na stavbu též 6mil. LZE VYHOVĚT. Bance zbude 8. Třetí žádá rovněž 6mil. Kdyby je banka půjčila, zbyly by jí jen 2mil. a to nestačí k dokončení žádné s investic, které je nutné pro zahájení splátek. PŮJČIT LZE JEN 2 mil., aby zůstalo 6 k dokončení jedné z investic.
    - HODNOCENÍ: **Dobré využití** zdrojů. Poměrně náročný algoritmus **zdržuje.**

Obvyklé řešení problému nesdílitelných vnějších zařízení

- **JEDEN PROCES NA POPŘEDÍ**
  - pracuje v reálném čase, má přidělen skutečná nesdílitelná zařízení (obrazovku, tiskárnu, …)

- **PROCESY NA POZADÍ**
  - Pracují (zatím) s **virtuálními vnějšími zařízeními** (oblasti v paměti či na disku) a běží s nižší prioritou.



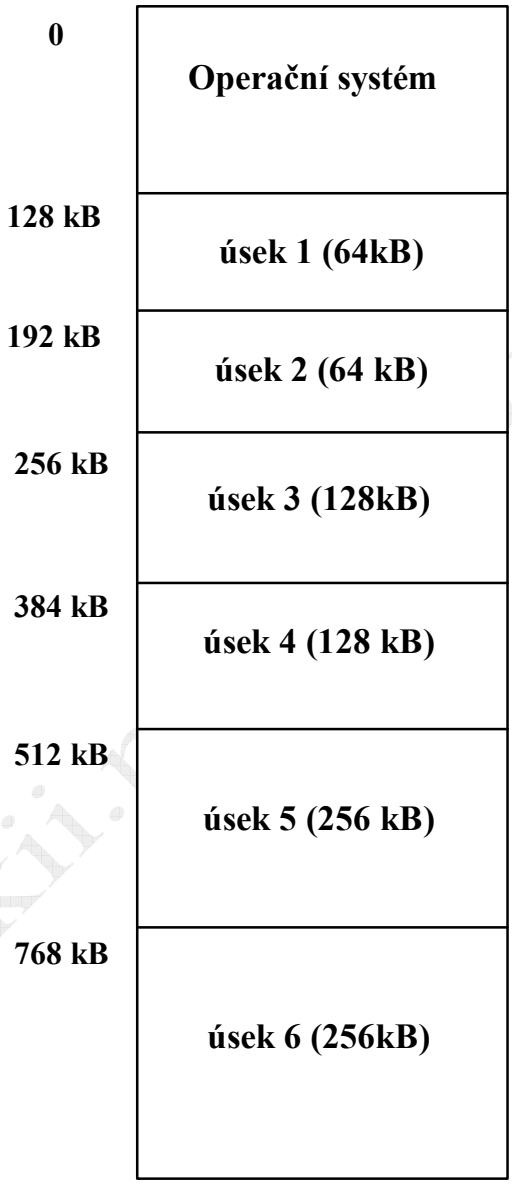
Systémový čteč - připravuje data ze skutečných vstupů na virtuální
P1, P2, P3 - výkonné programy, pracující s virtuálními I/O
Systémový vypisovač – tiskne data z virtuálních tiskáren (diskových oblastí) na skutečné tiskárně Při zpracování *N* uživatelských procesů paralelně je ve skutečnosti spuštěno *N* + 2 uživatelských procesů.
Systémový čteč a vypisovač mají prioritu.

Strategie přidělování paměti

- Pokud je **multitaskingový operační systém** v činnosti, je rezidentní část jádra operačního systému umístěna obvykle **na začátku hlavní paměti**.
- **Zbývající část hlavní paměti** operační systém přiděluje zpracovávaným **procesům**. Jsou možné tyto způsoby jejího přidělování:
  - přidělování **statických souvislých úseků**
  - přidělování **dynamických souvislých úseků**
  - přidělování **virtuálního adresového prostoru**

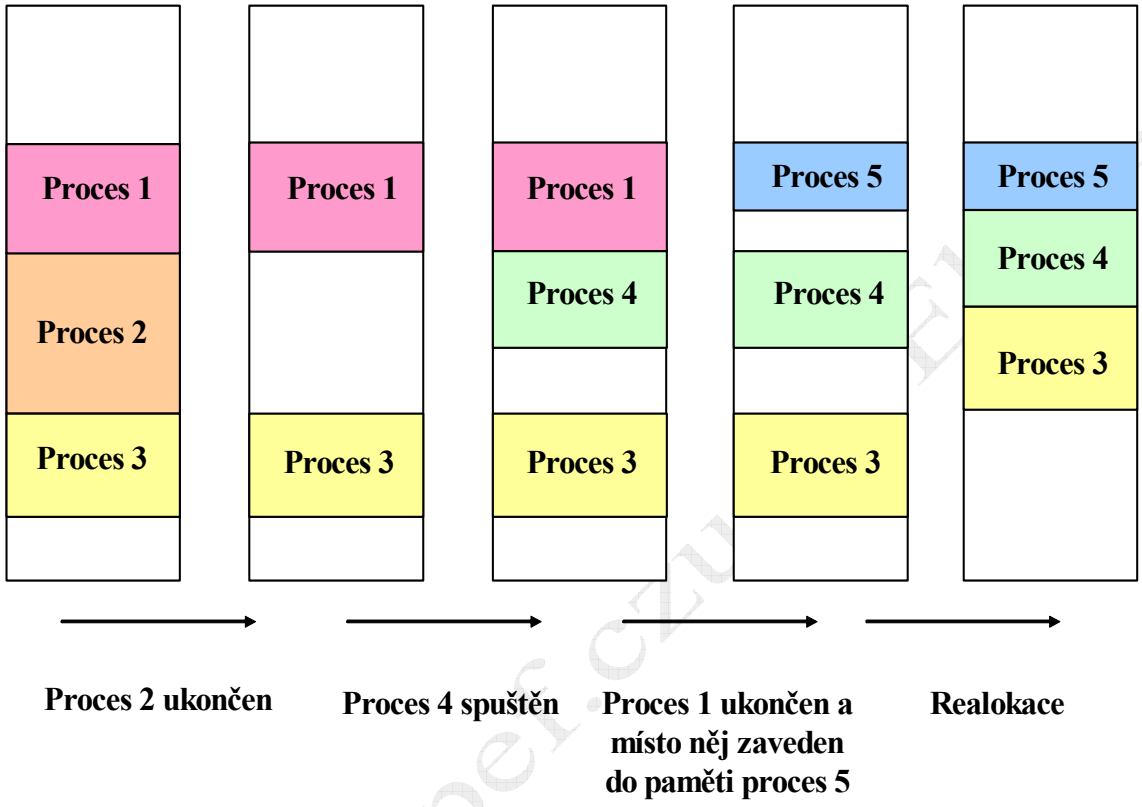
Přidělování statických souvislých úseků

- Paměť je po celou dobu běhu operačního systému rozdělena na souvislé úseky, jejichž umístění se nemění
- Proces se uloží do nejmenšího volného úseku, do kterého ho lze uložit



Přidělování dynamických souvislých úseků

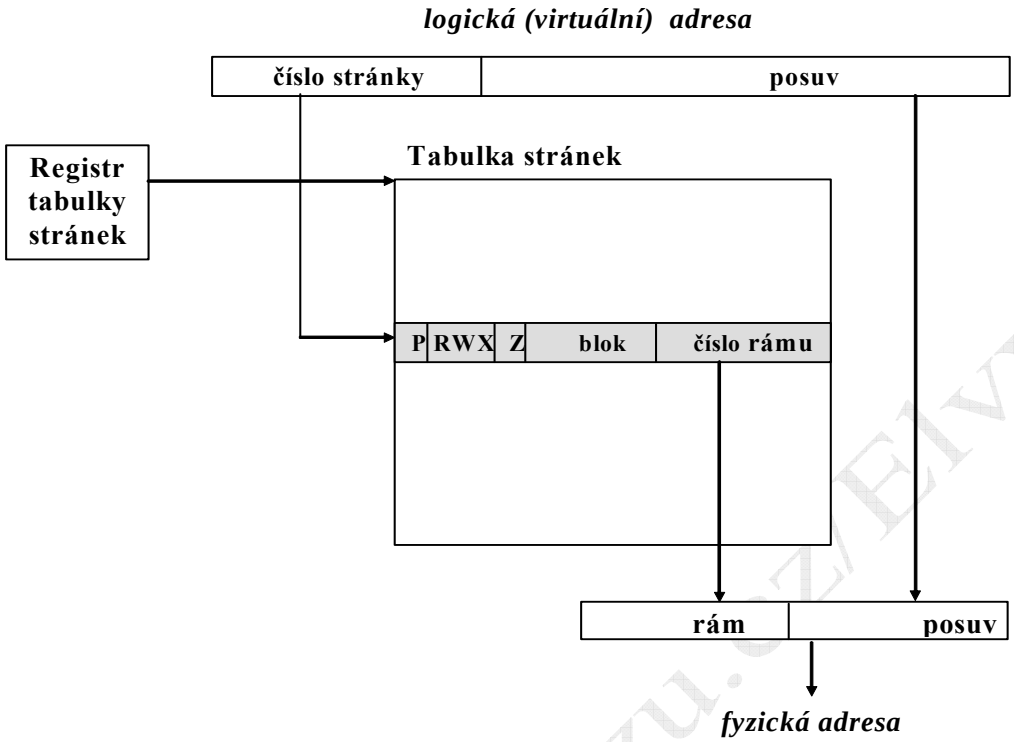
- Operační systém vybere nejmenší souvislou volnou část paměti tak, aby do ní mohl umístit celý proces



- Paměť obsahuje řadu malých, procesy neobsazených částí → **fragmentace paměti**
  - U přidělování dynamických úseků lze odstranit **realokací** obsazených úseků paměti – spotřebovává procesorový čas
- **Proces není chráněn** před ostatními procesy
  - Proces může přepsat oblast paměti vyhrazenou jinému procesu
  - Ochrana procesu pomocí techniky **klíč-zámek**

Virtuální paměť

- Je-li **proces větší než fyzická paměť počítače**, nelze jej přímo uložit a spustit
  - Proces je nutno rozdělit na menší části (tzv. *overlays*) a postupně nahrávat do paměti a zpracovávat
- Je-li postupné umísťování do paměti prováděno automaticky, mapuje jednotka **DAT** (*Dynamic Adress Translation* – logický obvod, který je součástí procesoru nebo samostatnou jednotkou) logický (virtuální) adresový prostor procesu na fyzický tak, jak stanovuje operační systém.
  - Procesor může pracovat v celém svém logickém prostoru, jakoby se fyzická paměť rozšířila na velikost logické paměti – **virtuální paměť**
- **Stránkování paměti**
  - **virtuální adresový prostor** rozdělen na **stránky** (1-8kB)
  - **fyzický adresový prostor** (fyzická operační paměť) rozdělen na stejně velké úseky – **rámy** nebo **rámce** (*frames*)
- **Logická (virtuální) adresa** rozdělena na
  - **číslo stránky**
  - **posuv** (*offset*) – vyjadřuje umístění (lokální adresu) v rámci stránky
- **Každý proces** má svoji **tabulku stránek**
- **Každý řádek tabulky** stránek odpovídá jedné stránce, **v řádce je uvedeno**:
  - **číslo rámu**, který operační systém stránce přidělil
  - **bit platnosti** stránky – **P**
  - **kód autorizace** –
    - **R** (povolení číst)
    - **W** (povolení zapisovat)
    - **X** (použít stránku pro řízení procesu)
  - **bit změny** obsahu stránky – **Z**
  - **číslo bloku**, ze kterého lze stránku získat (systém souborů, swappovací oblast)



- Operační systém procesu namapuje jen určitý počet stránek – použije-li proces virtuální adresu mimo namapovaný prostor, dojde k **výpadku stránky** → je generováno přerušení → OS požadovanou stránku namapuje

- Operační systém často mapuje procesu stránky až, když dojde k výpadku stránky – **stránkování na žádost**

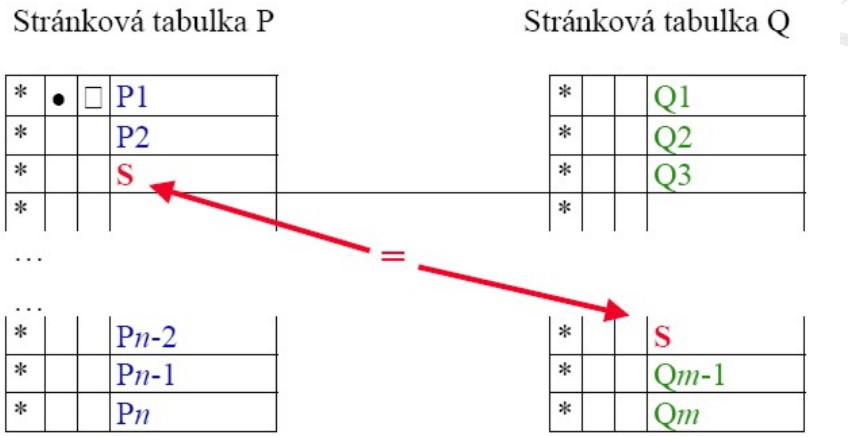
- Nastane-li situace, kdy nejsou žádné volné rámy, OS musí některé stránky **odmapovat** – **různé strategie výběru stránek** - nejlepší **LRU** (poslední použitá), dále **FIFO** (nejdéle použitá) nebo **náhodná**.

- Přenosy při řešení výpadku stránky velmi zdržují. Nesmí být požadovány příliš často. Nebezpečí **ZAHLCENÍ** – „trashing“.

- Proto je rozumné určit pro každý proces tak zvanou **PRACOVNÍ MNOŽINU** (= počet stránek, které by měly být přítomny v paměti trvale). Zda je volné místo pro pracovní množinu kontrolovat před zahájením procesu programem řízení úloh (spolu s tím, zda jsou volné nesdílitelné zdroje).

Zajištění determinismu a spolupráce procesů

- **Různé procesy** mají každý svoji stránkovou tabulku. Začátky těchto tabulek jsou uvedeny **ve společné tabulce segmentů**.
  - Zcela **nezávislé procesy** mají **obsahy tabulek disjunktní**
  - **Procesy**, které **spolupracují** mají ve svých stránkových tabulkách odkazy na některé **společné stránky**. Tyto stránky obsahují společná data obou procesů a jsou jim přiřazeny stejné stránkové rámy v odkládacím prostotu na disku. Přístup ke sdíleným datům je nutné řídit **kritickou sekcí** ve všech spolupracujících procesech.



P*i* ≠ Q*j* pro všechna *i* a *j*.
S = stránka se sdílenými daty