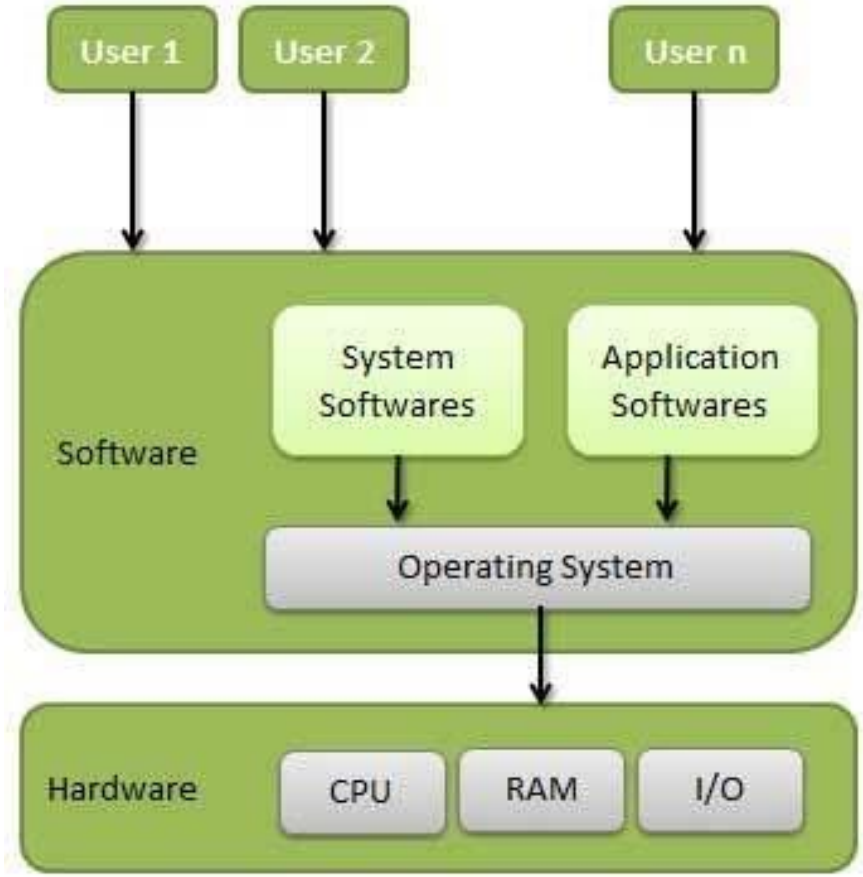
Systemy operacyjne

Procesy, synchronizacja, przetwarzanie

System Operacyjny



https://www.tutorialspoint.com/operating_system/images/conceptual_view.jpg

Ogólne zadania systemów operacyjnych

Zarządzanie pamięcią

Zarządzanie czasem procesora

Zarządzanie urządzeniami

Zarządzanie plikami

Zabezpieczenie

Zarządzanie wydajnością systemową

Nadzorowanie pracy

Wspomaganie wykrywania błędów

Nawigacja pomiędzy wszelkim oprogramowaniem i użytkownikami

Typy systemów operacyjnych

- wsadowe
- podziału czasowego
- rozproszony
- sieciowy
- czasu rzeczywistego

Wsadowy system operacyjny

- brak bezpośredniej interakcji użytkownika ze sprzętem
- każde zadanie wykonywane jest oddzielnie dla użytkownika na zapisanym nośniku
- zadania mogą być grupowane
- osprzęt jest często w stanie jałowym
- ciężko jest zaplanować priorytetyzację zadań

System operacyjny z podziałem czasu

- działa jako pojedyczy system z wieloma terminalami
- nastawione na minimalizację czasu odpowiedzi
- każdy proces każdego użytkownika otrzymuje określony czas przydziału do procesora (kwant czasu)
- zmniejsza stan jałowy procesora
- problemy z bezpieczeństwem przetwarzanych danych
- komunikacja pomiędzy danymi poszczególnych procesów
- niezawodność rozwiązania

Rozproszony system operacyjny

- rozwiązania wieloprocesorowe wielomaszynowe
- obsługa wielu aplikacji czasu rzeczywistego wielu użytkowników
- komunikacja następuje poprzez różnego rodzaju sprzęgi łączące
- redukcja przestojów pomiędzy przepływem danych (także ładowania ich na urządzenia)
- nawet uszkodzenie pojedynczych połączonych jednostek nie eliminują działania systemu
- możliwość załadowania informacji do systemu z różnych terminali i aplikacji

Sieciowy system operacyjny

- uruchamiany głównie na serwerze

- udostępnia szeroko pojęte usługi, takie jak udostępnianie dokumentów, drukarek czy dostęp do innych systemów i sieci

- możliwość łączenia się z dowolnego węzła sieciowego

- łatwa aktualizacja zasobów zarówno oprogramowania jak i sprzętu

- duża centralizacja usług

- bezpieczeństwo usług sieciowych i związane z tym bezpieczeństwo użytkowników terminalowych

System operacyjny czasu rzeczywistego

- systemy tego typu kładą nacisk na obsługę żądań użytkownika

- najczęściej są to systemy reagujące na dowolne przerwanie od użytkownika

- systemy te nie należą do grupy ogólnego przeznaczenia; są najczęściej projektowane do określonego zadania

- są silnie nakierowane na stałe czasy odpowiedzi, niejednokrotnie zapisywane są na pamięci typu ROM

- w mniej restrykcyjnej formie pozwalają na priorytetyzowanie zadań, posiadają także pewne rozwiązania narzędziowe

- rozróżnia się systemy ściśle rzeczywiste oraz luźne rzeczywiste

Usługi systemu operacyjnego

Wykonywanie programów

Operacje wyjścia/wejścia

Obsługa systemu plików

Komunikacja

Wykrywanie błędów

Alokacja zasobów

Ochrona

Wykonywanie programów

Ładowanie programów do pamięci.

Wykonywanie programów.

Dostarczanie mechanizmu synchronizacji procesów.

Dostarczanie mechanizmu komunikacji pomiędzy procesami.

Dostarczanie mechanizmu eleminacji zakleszczeń.

Operacje wejścia/wyjścia

Operacje I/O pozwalają na obsługę zapisu oraz odczytu danych z dowolnego zasobu na dowolnym nośniku.

System operacyjny zapewnia dostęp do wymaganych urządzeń (poprzez własne mechamizmy lub sterowniki).

Obsługa systemu plików

System operacyjny dostarcza uprawnienia do odczytu i zapisu określonych zasobów w pamięci masowej.

Mogą to być dowolne kombinacje pozwoleń (odczyt, zapis, zabronienie).

System ma za zadanie dostarczy odpowiedni interfejs do obsługi zasobów.

Możliwa jest też obsługa kopii zapasowej zasobów.

Komunikacja

System ma za zadanie zapewniać transfer danych zarówno pomiędzy procesami, jak i innymi urządzeniami.

Transfer może odbywać się pomiędzy jedną maszyną lub pomiędzy różnymi maszynami.

Przemieszczanie danych może być realizowane poprzez np. współdzieloną pamięć lub poprzez wysyłanie komunikatów.

Obsługa błędów

System ma za zadanie wychwytywać wszelkie możliwe błędy, zarówno oprogramowania jak i sprzętu.

System powienien podejmować odpowiednie operacje w reakcji na określone błędy celem zachowania odpowiedniego poziomu operacyjności przetwarzania danych.

Zarządzanie zasobami

System powinien zarządzać wszelkimi zasobami sprzętowymi oraz programowalnymi dostępnymi dla wykonywanych zadań.

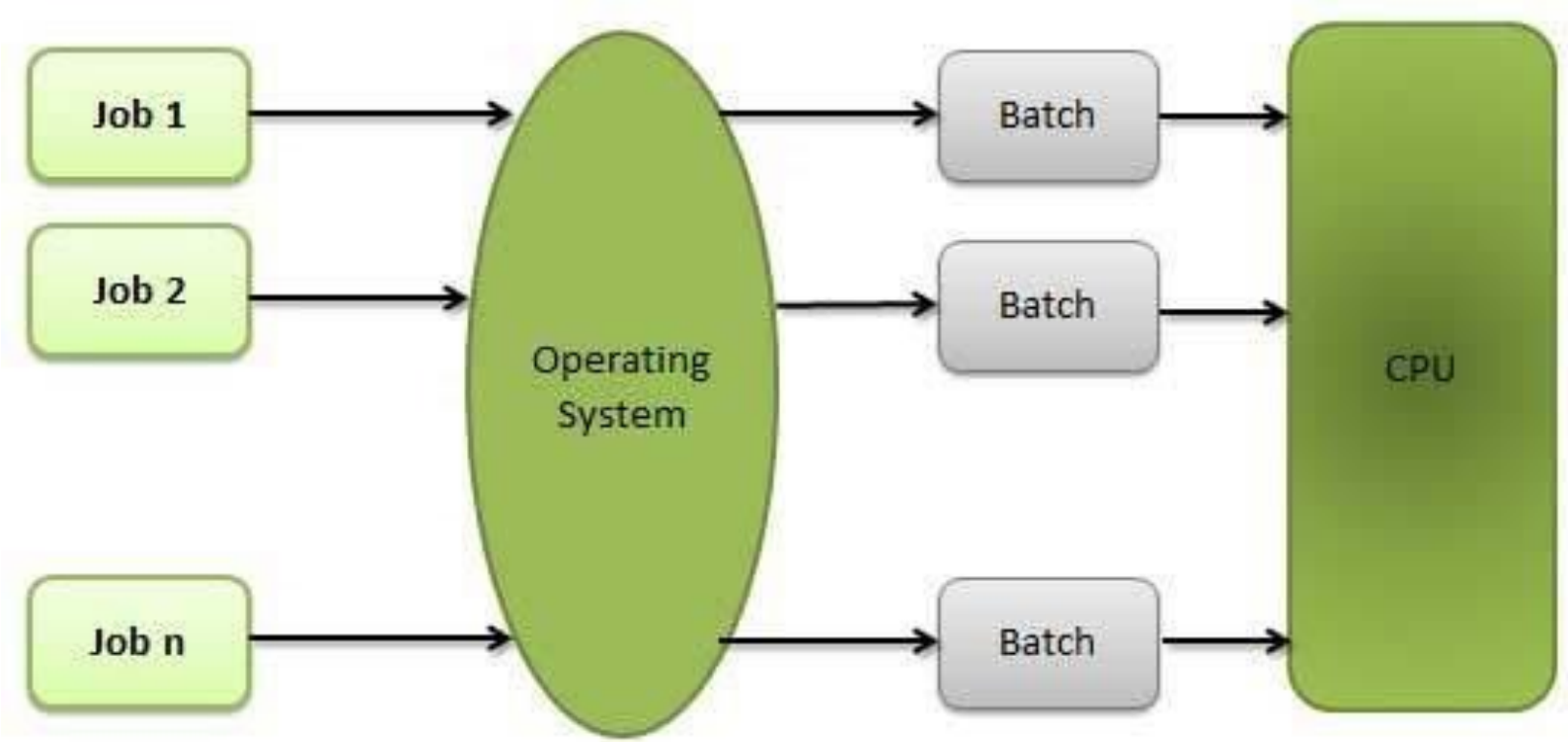
Odpowiednie algorytmy zarządzania przydziałem do zasobów (mikroprocesora) pozwalają na lepsze wykorzystanie dostępnych zasobów.

Ochrona

System powienien zapewniać ochronę zarówno przez wewnętrznymi próbami nieprawidłowego dostępu do określonych danych, jak i przed dostępem z zewnątrz.

Ochrona może być dostarczana poprzez różne narzędzia; najbardziej podstawowym jest system autoryzacji oparty o hasło.

Przetwarzanie wsadowe



https://www.tutorialspoint.com/operating_system/images/batch_processing.jpg

Przetwarzanie wsadowe

Wprowadza automatyzację działania zadań, poprzez predefiniowane sekwencje komend.

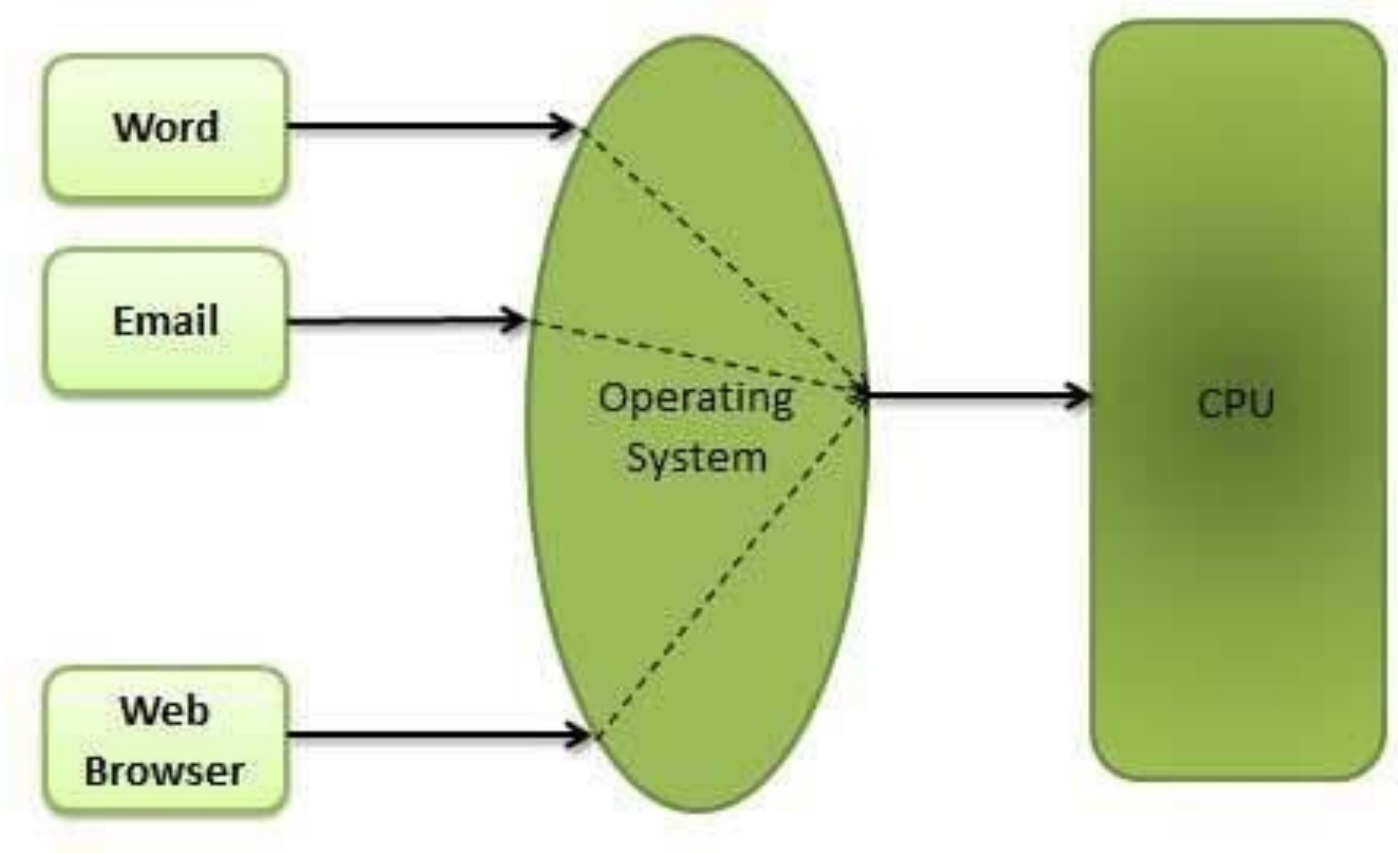
Najczęściej działa na zasadzie kolejkowania.

Po zakończeniu pracy pamięć operacyjna jest automatycznie zwalniania.

Niekiedy działania mogą wykonywać się wadliwie.

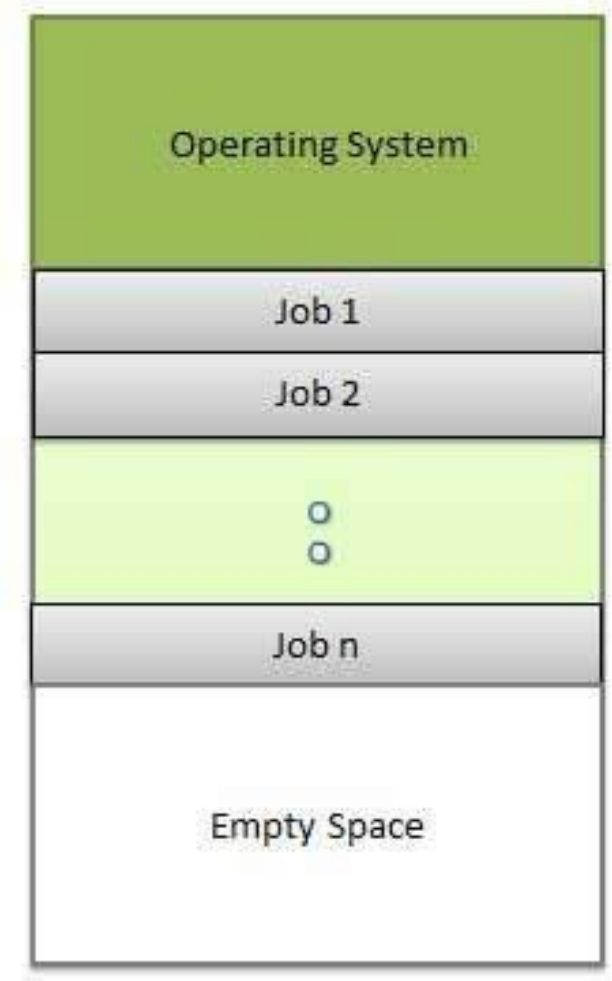
Może następować tzw. Zagłodzenie procesów.

Wielozadaniowość



https://www.tutorialspoint.com/operating_system/images/multitasking.jpg

Wielozadaniowość



https://www.tutorialspoint.com/operating_system/images/memory_layout.jpg

Wielozadaniowość

Użytkownik wprowadza dane bezpośrednio, uzyskując odpowiedzi najszybciej jak to możliwe.

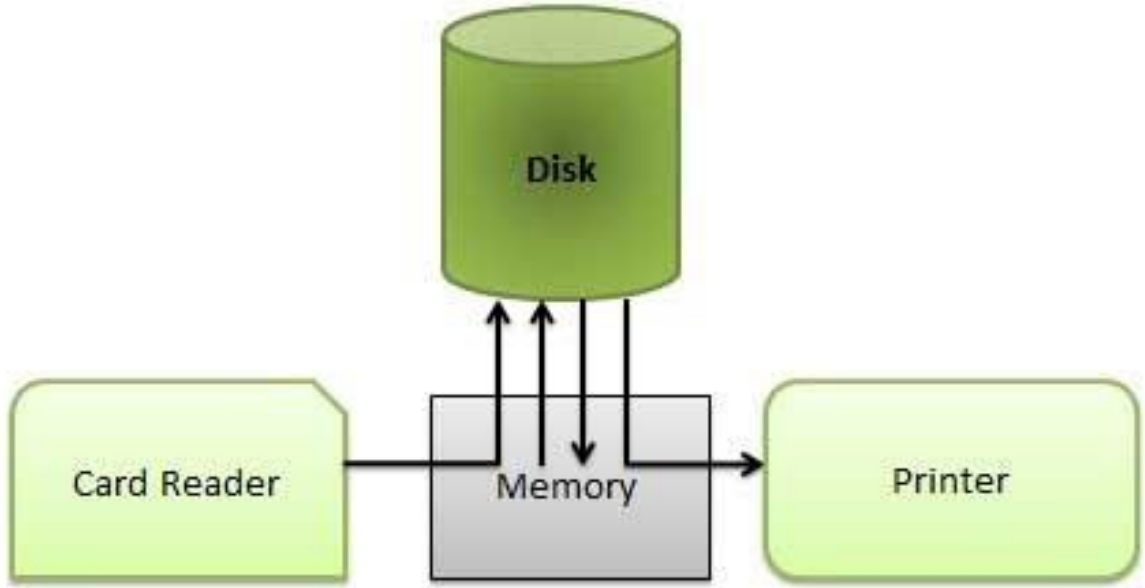
Wielozadaniowość pozwala na osiągnięcie tzw. interaktywności z użytkownikiem.

Każdy aktywny użytkownik posiada przynajmniej jeden niezależny proces w pamięci.

Przeważnie systemy te umożliwiają jednoczesną pracę wielu użytkowników i/lub programów.

Dzięki odpowiednim technikom i rozwiązaniom (np. Planisty) systemy mogą przełączać poszczególne pracoesy (np. Długo treające), by ostatecznie dopasować wykorzystanie CPU dla pozostałych, uruchomionych procesów.

Buforowanie



https://www.tutorialspoint.com/operating_system/images/spooling.jpg

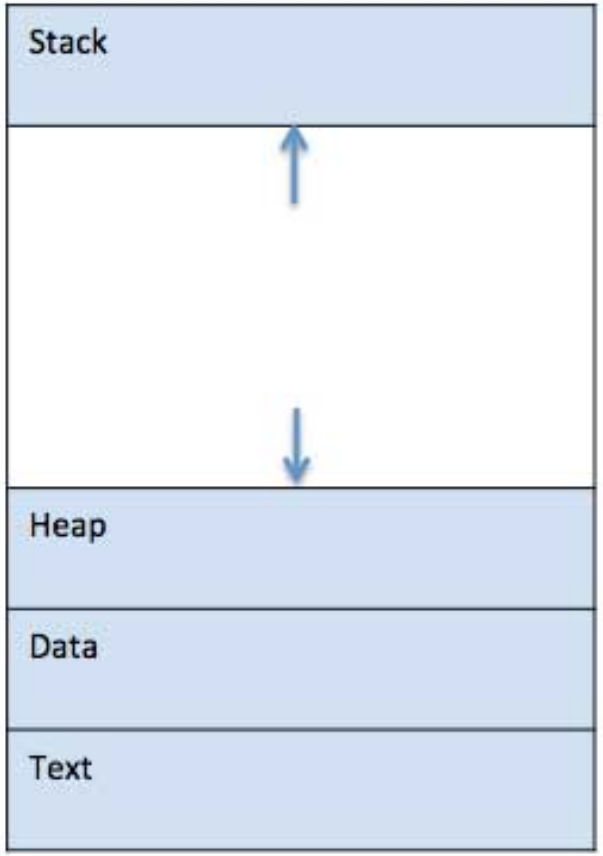
Buforowanie

Obsługa urządzeń posiadających różne czasy dostępu do różnych nośników.

Implementacja umożliwia oczekiwanie (np. w pamięci podręcznej) do czasu przechwycenia danych przez wolniejsze urządzenia.

Pozwala na rórnoległe przetwarzania, np. Wczytywania danych z kilku różnych nośników, zapisywania danych na innym nośniku i jednocześnie drukowaniu wyników na kartce papieru.

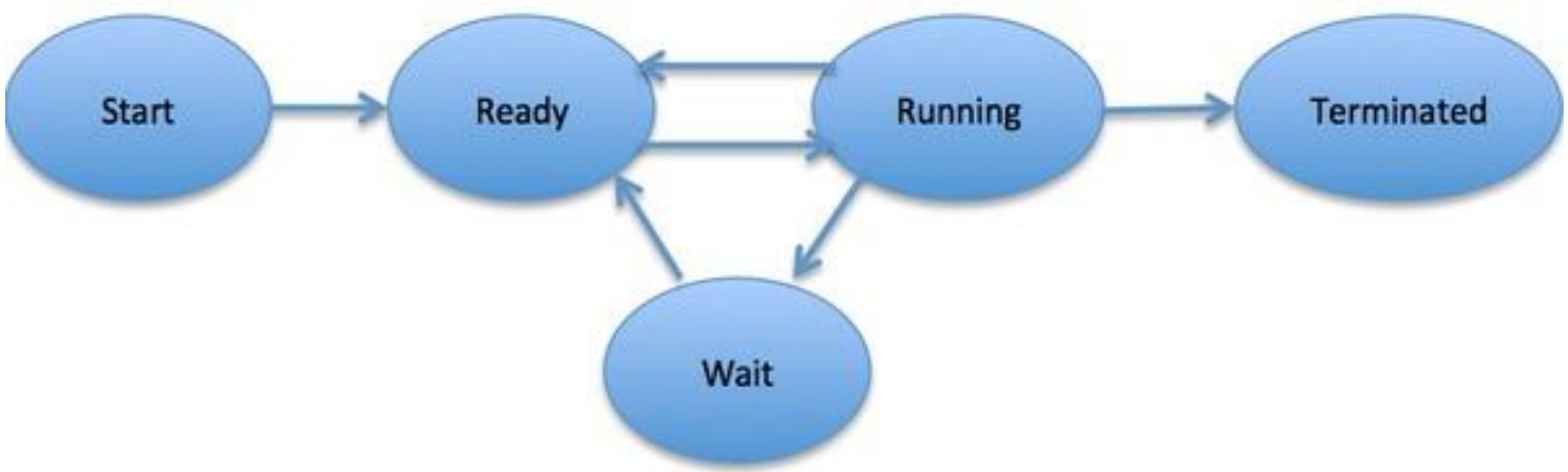
Działanie podprogramów



Stos	Zawiera dane i zmienne lokalne aktualnie wykonywanych funkcji
Sterta	Dynamicznie alokowana pamięć dla procesów
Dane	Zawiera zmienne globalne i statyczne
Tekst	Zawiera informacje o aktualnych działaniach - wartość licznika programowego oraz rejestrów procesora

https://www.tutorialspoint.com/operating_system/images/process_components.jpg

Cykl życia procesu

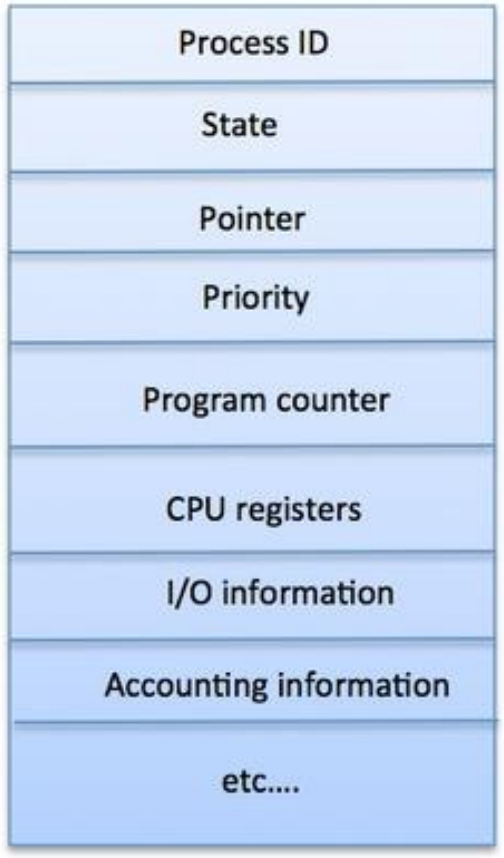


https://www.tutorialspoint.com/operating_system/images/process_state.jpg

Cykl życia procesu

Start	Inicjalizacja procesu (i jego wszelkich wartości)
Gotowość	Oczekiwanie na dostęp do procesora. Ten stan jest osiągalny gdy proces jest po fazie startu lub przerwania działania (znajduje się w fazie oczekiwania) przez planistę systemu.
Działanie	Proces wykonuje przypisane mu instrukcje na procesorze
Oczekiwanie	W tej fazie proces znajduje się np. Gdy oczekuje na dodatkowe zasoby.
Zakończenie/ Wyjście	W tej fazie program oczekuje na usunięcie z pamięci.

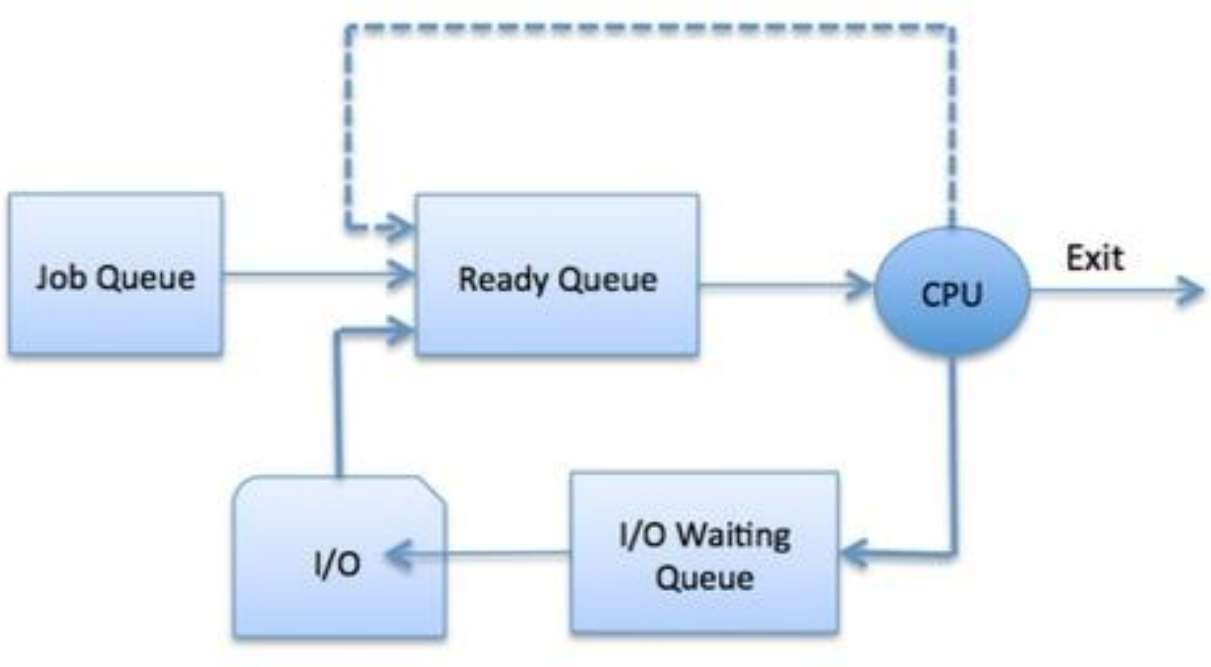
Blok kontroli procesu



ID procesu	Unikatowe wskazanie na każdy proces w systemie operacyjnym
Status procesu	Aktualny stan cyklu procesu
Wskaźnik	Wskaźnik na proces rodzica
Priorytet	Informacje wykorzystywane do planowania przydziału procesu do CPU
Licznik programu	Wskaźnik adresu następnej instrukcji do wykonania w ramach procesu
Rejestry jednostki przetwarzającej	Niezbędne rejestry dla przechowania procesu w czasie wykonania
Informacje I/O	Lista urządzeń wejścia/wyjścia przydzielonych procesowi.
Informacje przydziału	Informacje o użyciu procesora, limitach czasu, ID wykonania
Informacje użycia pamięci	Informacje dotyczące stronicowania, limitów pamięci itp.
Uprawnienia procesu	Niezbędne do działania zasoby

https://www.tutorialspoint.com/operating_system/images/pcb.jpg

Kolejka procesów



https://www.tutorialspoint.com/operating_system/images/queuing_diagram.jpg

Kolejki procesów

Kolejka zadań – kolejka wszystkich zadań obecnie uruchomionych w systemie operacyjnym.

Kolejka gotowości – przechowuje wszystkie procesy rezydujące w pamięci gotowe do wykonania.

Kolejka urządzeń – lista procesów zablokowanych poprzez niedostępność urządzeń wejścia/wyjścia.

Planista

- oprogramowanie systemu operacyjnego

- zarządza dostępem procesów do wykonania na jednostce przetwarzającej

- w systemach mogą występować trzy rodzaje planistów: długoterminowy, krótkoterminowy oraz zrównoważony

Planista długoterminowy (Long Term Scheduler)

- ustala listę procesów mogących korzystać z zasobów

- równoważy ładowanie poszczególnych procesów i ich przydziału do CPU

- średnia ilość tworzonych procesów musi być równa średniej procesów oczekujących na zakończenie

- w praktyce rzadko używany

Planista krótkoterminowy (Short Term Scheduler)

- dobiera zadania na podstawie określonych warunków
- wybiera tylko te procesy, które są w pełni gotowe do wykonania swoich instrukcji
- szybszy niż planista długoterminowy

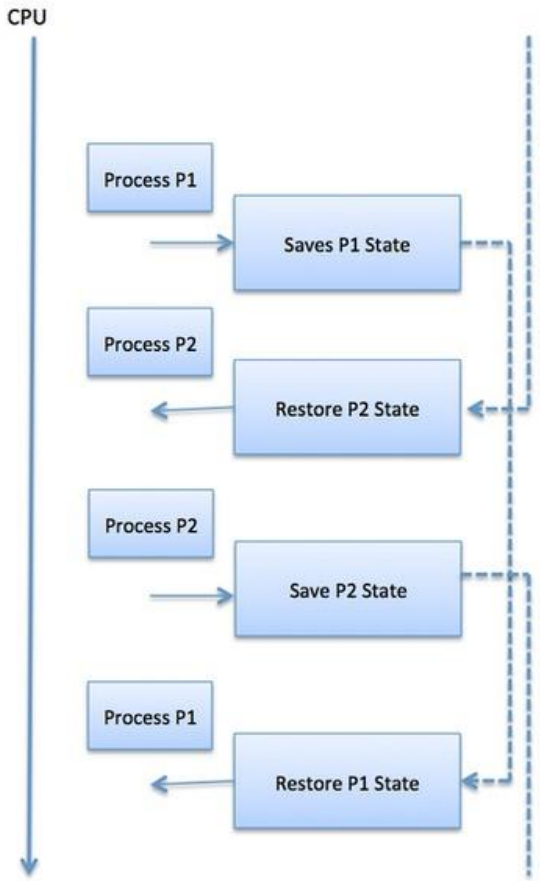
Planista zrównoważony (Medium Term Scheduler)

- często zwany zamieniaczem

- planista ten często "zamraża" (zawiesza) procesy poprzez usuwanie ich z pamięci w chwili, gdy oczekują na dodatkowe informacje z urządzeń wejścia/wyjścia

- to uwalnia pamięć dla innych, nowych procesów, które w danej chwili mogą korzystać z zasobów sprzętu

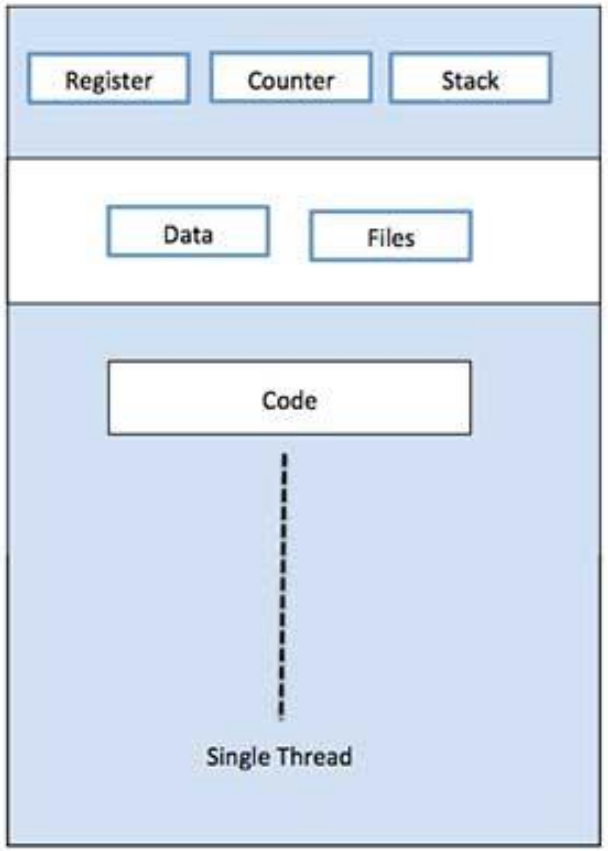
Przełączanie kontekstowe



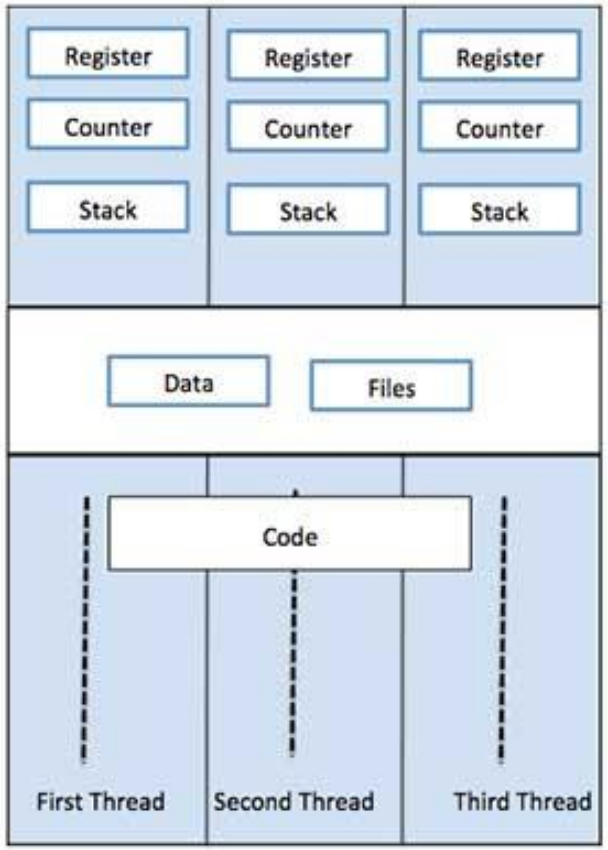
Informacje przechowywane podczas przełączenia procesów:
- Licznik programu
- Planer
- Wartości bazowe I maksymalne rejestów
- Aktualnie użytkowane rejestry
- Stany zmian
- Informacje o stanie I/O
- Wartości liczników

https://www.tutorialspoint.com/operating_system/images/context_switch.jpg

Wielowątkowość



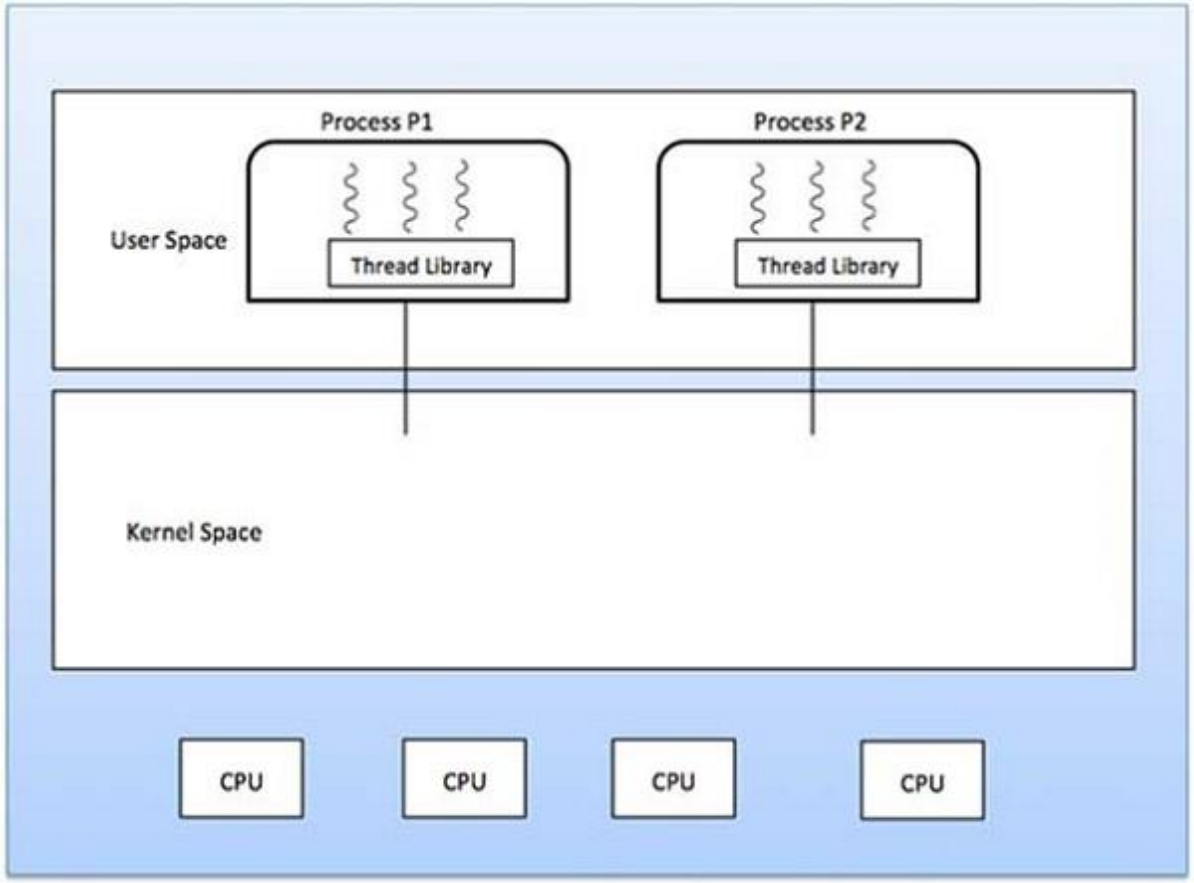
Single Process P with single thread



Single Process P with three threads

https://www.tutorialspoint.com/operating_system/images/thread_processes.jpg

Wątki użytkownika i jądra systemu



https://www.tutorialspoint.com/operating_system/images/user_threads.jpg

Wątki użytkownika

- nie wymagają przywilejów jądra
- mogą być uruchamiane (niemal) w każdym systemie operacyjnym
- szybkie przy tworzeniu i zarządzaniu
- utrudnione odwołania do elementów systemowych

Wątki jądra

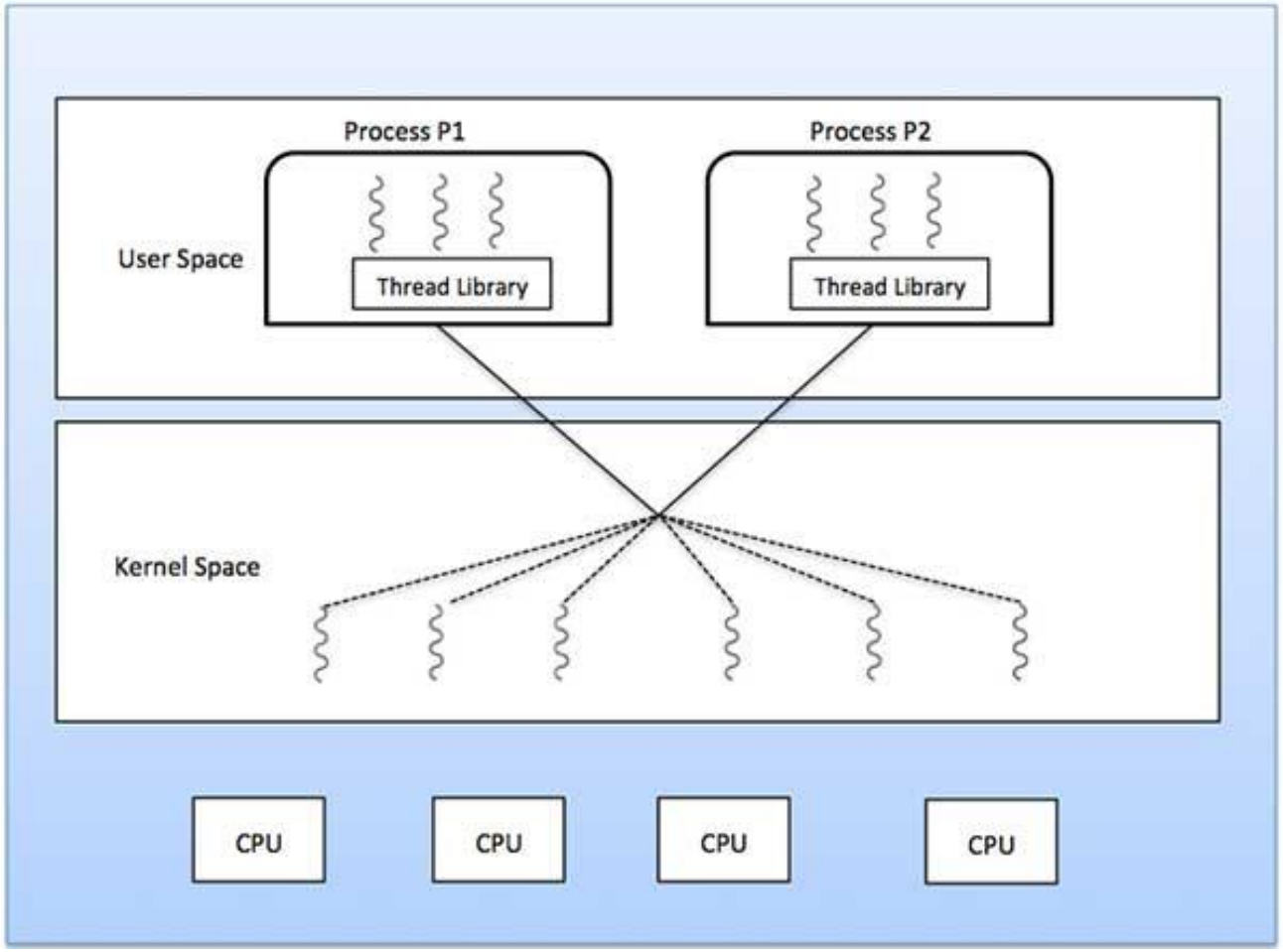
- jądro może planować przełączanie wątków pomiędzy poszczególnymi procesami, jednak wymaga to zawsze odwołania do jądra (mode switch)

- planista ma możliwość zarządzania każdym wątkiem wewnątrz procesu

- algorytmy jądra mogą być wielowątkowe

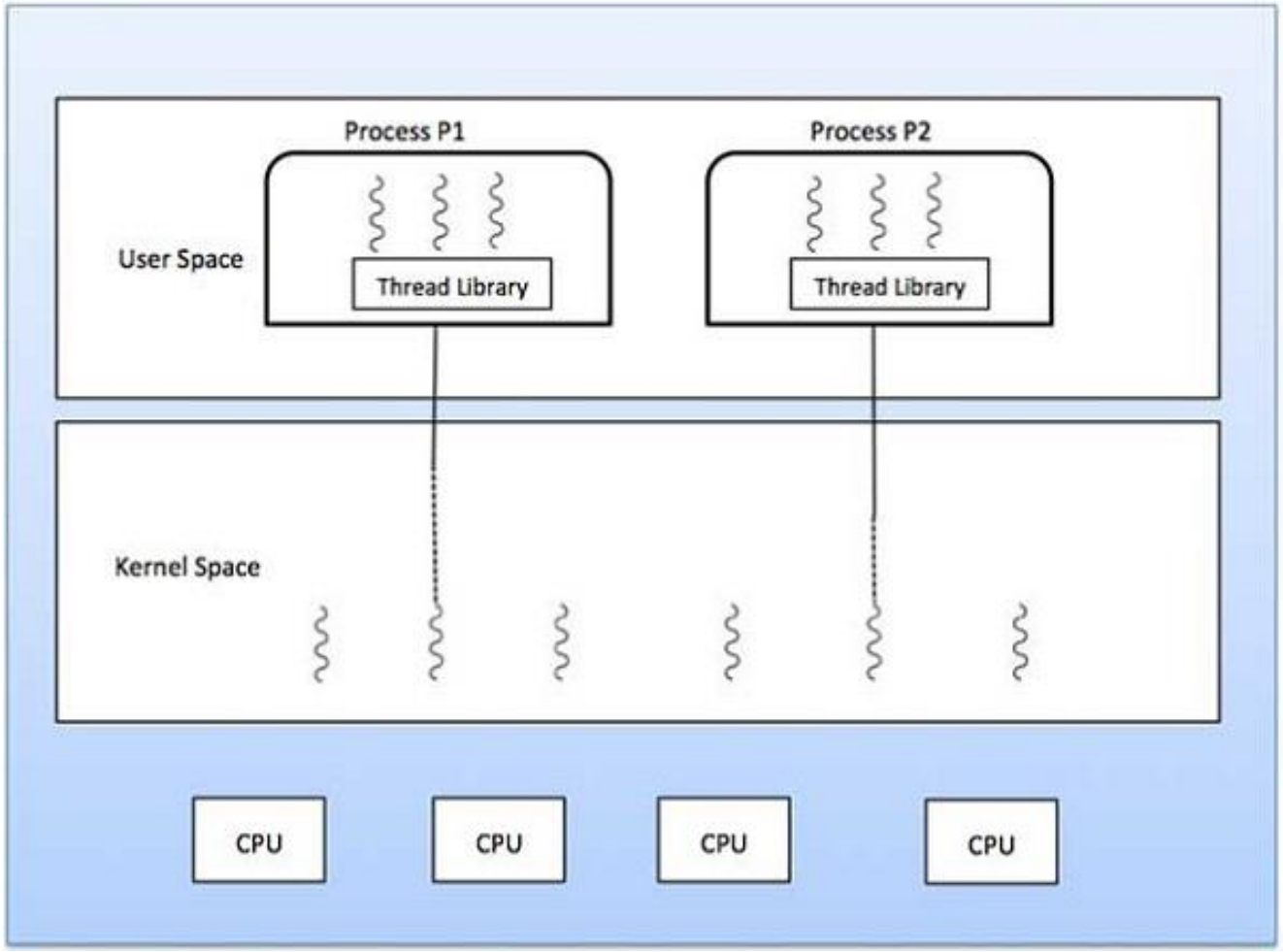
- niestety są one mniej wydajne (efektywne) przy tworzeniu i użytkowaniu niż wątki użytkownika

Model wiele-do-wielu



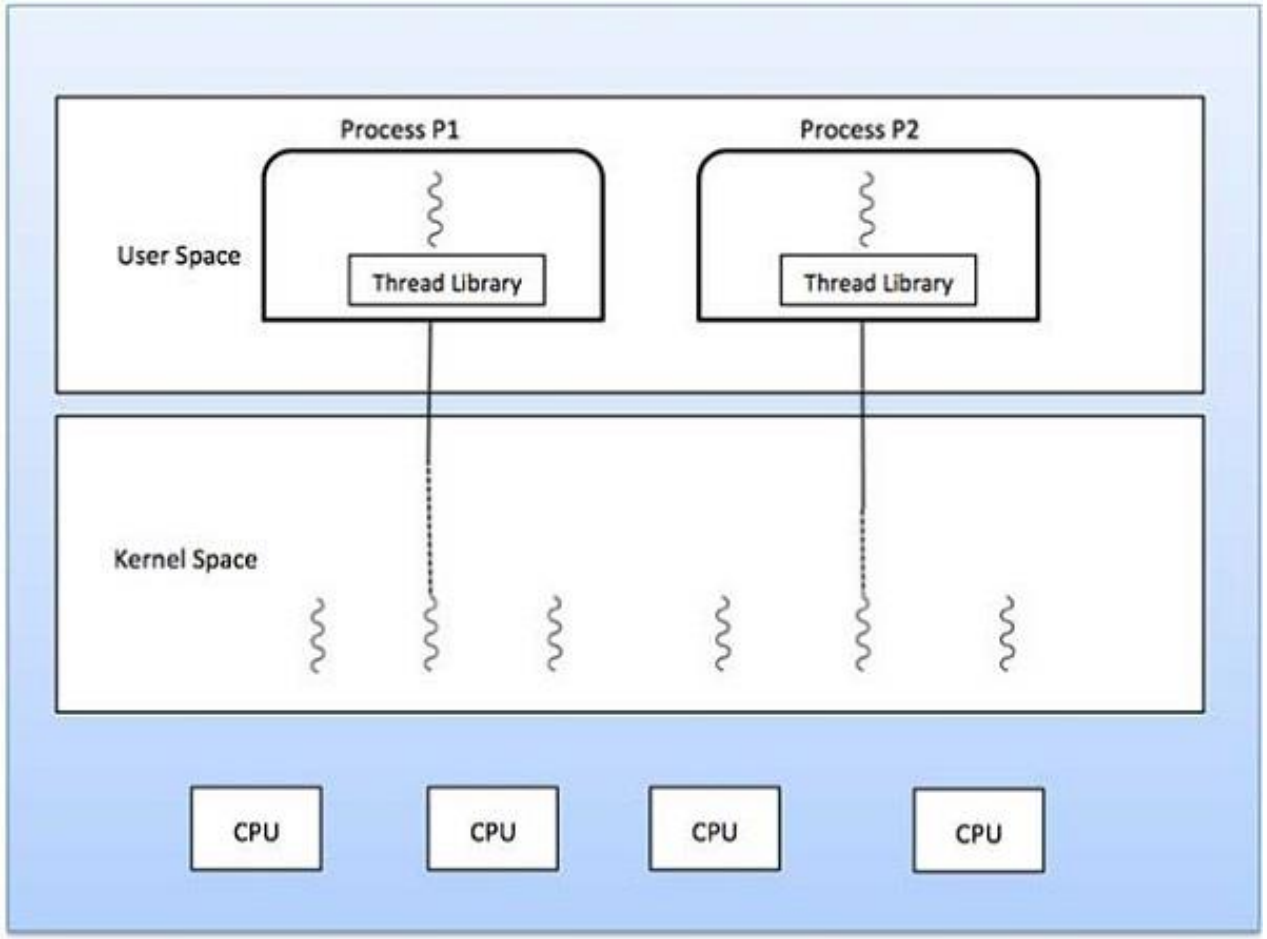
https://www.tutorialspoint.com/operating_system/images/many_to_many.jpg

Model wiele-do-jednego



https://www.tutorialspoint.com/operating_system/images/many_to_one.jpg

Model jeden-do-jednego



https://www.tutorialspoint.com/operating_system/images/one_to_one.jpg

Zarządzanie pamięcią

- system operacyjny zarządza przesyłaniem danych pomiędzy pamięcią główną a pamięcią magazynową

- przydziela i zwalnia pamięć używaną przez procesy

- to system zarządza, który proces dostanie przydział pamięci i w jakiej wysokości

- zarządzanie przestrzenią pamięci jest ściśle związane z architekturą sprzętową oraz systemową; 32-bitowe systemy mogą zarządzać teoretycznie maksymalnie 32-bitową przestrzenią adresową.

- ponieważ systemy operacyjne zarządzają pamięcią logicznie, możliwe są odstępstwa od tej reguły.

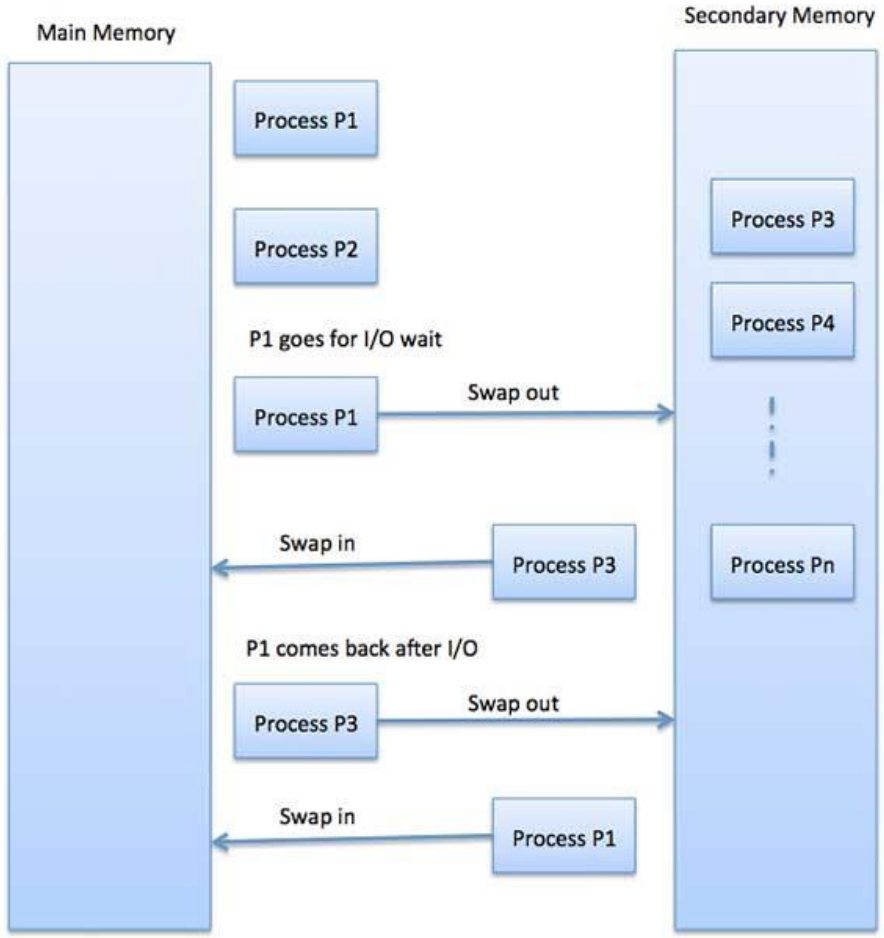
Typy adresacji dostępne w systemach operacyjnych

Adresowanie symboliczne - Zmienna, stałe oraz etykiety instrukcji są elementami tejże przestrzeni adresowej (kod źródłowy procesów)

Adresowanie relatywne - W czasie kompilacji adersy symboliczne zamieniane są na adresy relatywne

Adresowanie fizyczna - Adresy generowane z chwilą ładowania programu do pamięci.

Przełączanie pamięci



https://www.tutorialspoint.com/operating_system/images/process_swapping.jpg

Przełączanie pamięci

- mechanizm przerzucania procesu z pamięci głównej do pamięci masowej

- dzięki temu procesy aktualnie zablokowane bądź oczekujące mogą ustąpić miejsca procesom mogącym być obsłużonymi

- mechanizm obciążony dużym spadkiem efektywności procesowej

- niekiedy nazywany kompaktowaniem pamięci.

Organizacja pamięci operacyjnej (alokacja)

Alokacja pojedynczej partycji

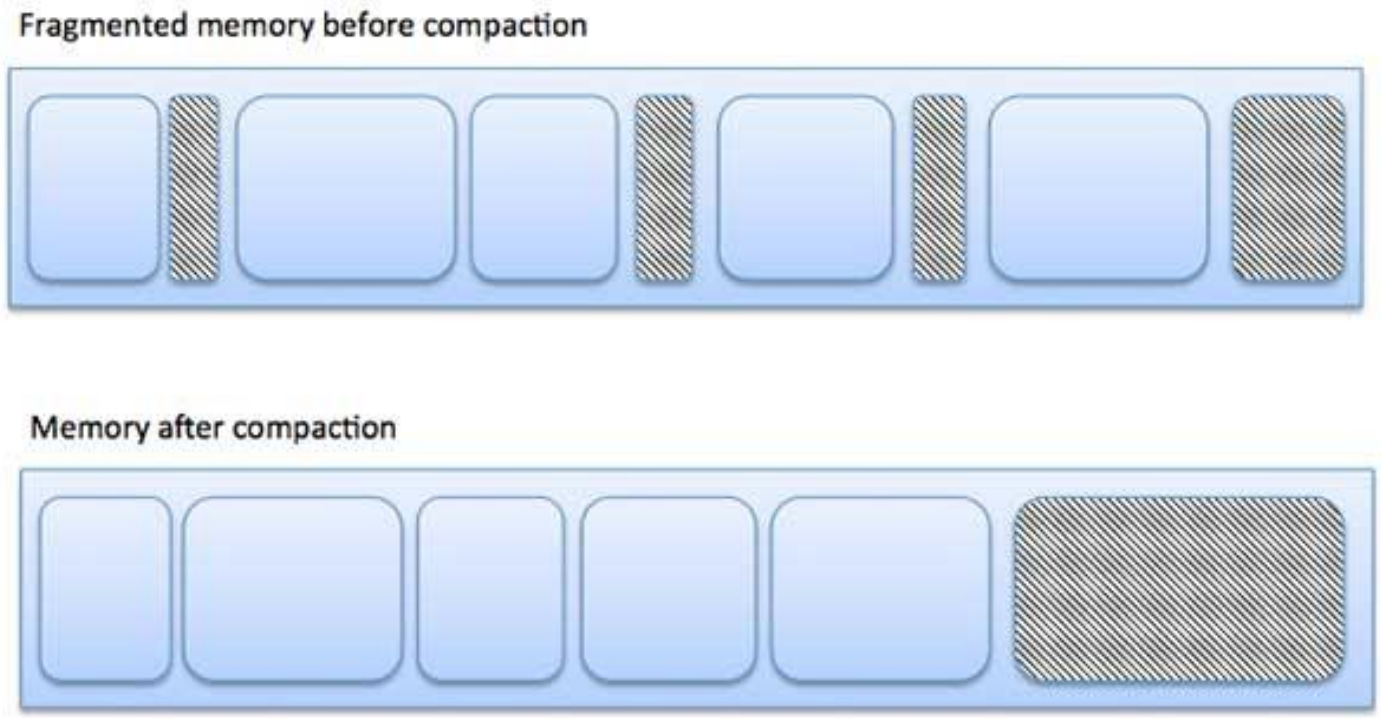
Rejestr relokacji chroni procesy użytkownika przed ich wspólnym złączeniem pamięci, w tym zmian kodu i danych.

Alokacja wielu partycji

Pamięć podzielona na wiele partycji o stałej wielkości - dla każdego procesu jedna.

Dodatkowo pamięć operacyjna podzielona jest na pamięć niską (low memory) oraz pamięć wysoką (hight memory). Pierwsza z nich zarezerwowana jest na główny kod systemu operacyjnego, druga zaś do użytku pozostałych procesów.

Fragmentacja pamięci



https://www.tutorialspoint.com/operating_system/images/memory_fragmentation.jpg

Rodzaje fragmentacji

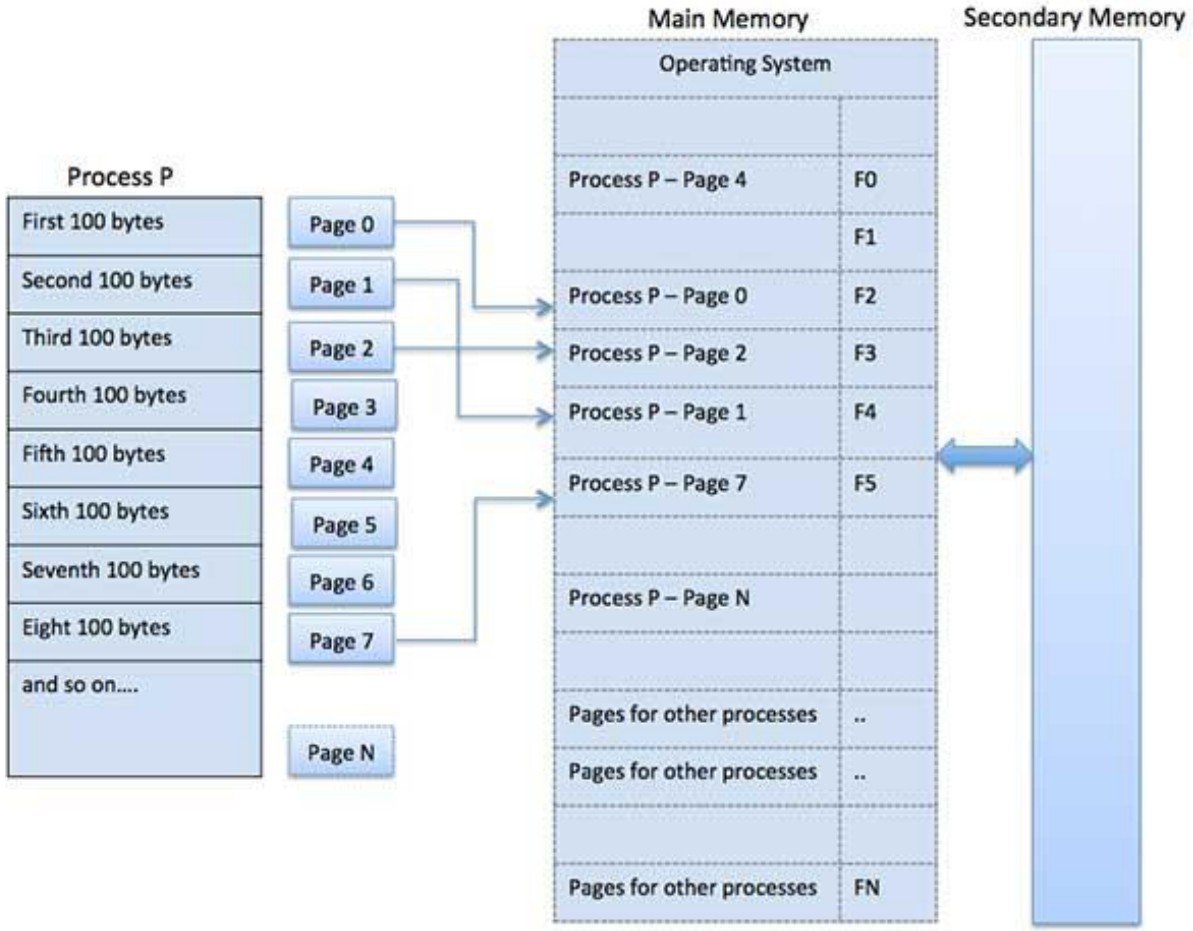
Fragmentacja zewnętrzna

Dostępna pamięć jest wystarczająca dla procesu, jednak jej nadmiar nie może zostać przydzielony do wymagającego jej procesu.

Fragmentacja wewnętrzna

Przydzielona pamięć jest większa niż wymaga proces, przez co pamięć zostaje w pełni użyta.

Stronicowanie



https://www.tutorialspoint.com/operating_system/images/paging.jpg

Stronicowanie

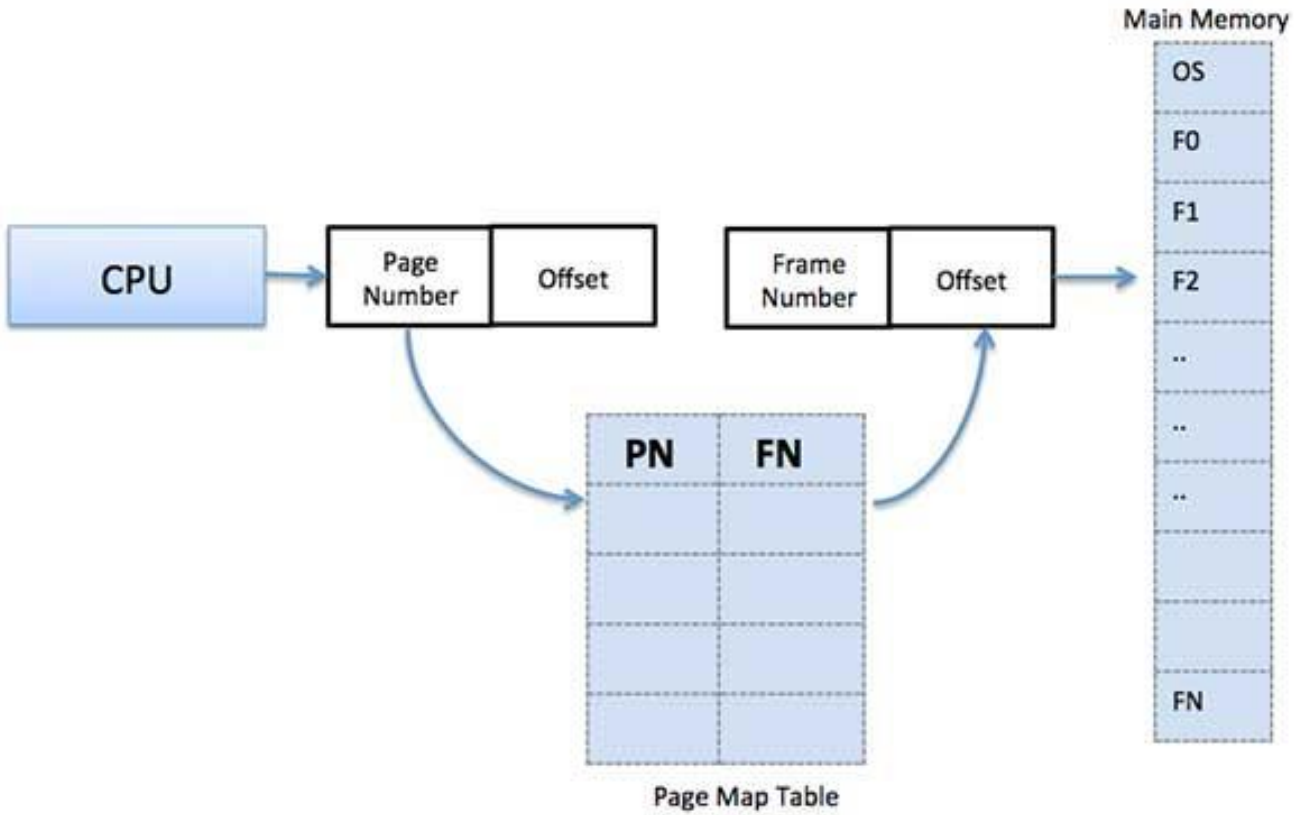
- system może adresować więcej pamięci niż jej fizyczne zasoby

- pamięć ta nazywana jest wirtualną

- stronicowanie polega na dzieleniu przestrzeni adresowej na bloki zwane stronami (potęga 2)

- główna pamięć również podzielona jest na małe bloki fizycznej pamięci (ramki), które są tożsame ze stroną

Tłumaczenie adresów (mapowanie)



https://www.tutorialspoint.com/operating_system/images/page_map_table.jpg

Tłumaczenie adresów

Adres strony nazywany jest adresem logicznym

Logical Address = Page number + page offset

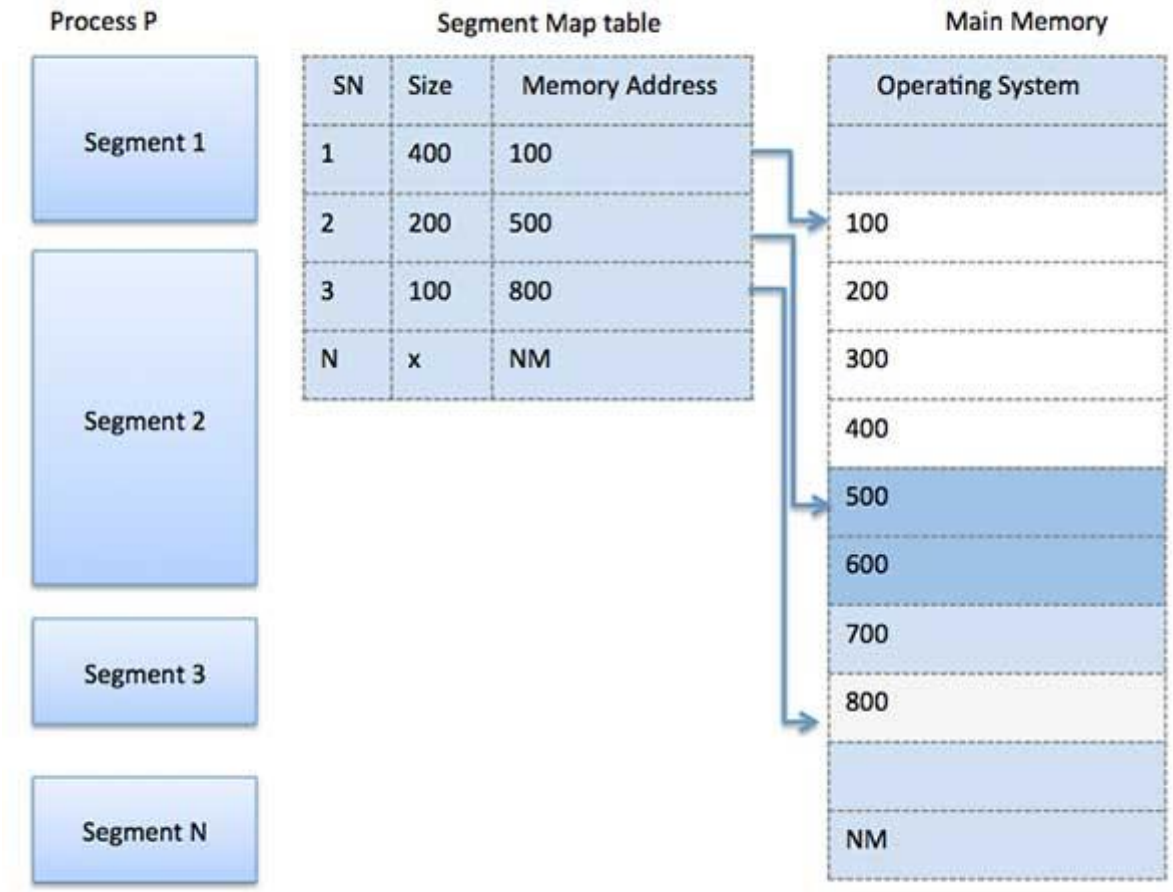
Adres ramki nazywany jest adresem fizycznym

Physical Address = Frame number + page offset

Stronicowanie - podsumowanie

- Zmniejsza zewnętrzną fragmentację, nie eliminuje wewnętrznej
- proste w implementacji i efektywne w zarządzaniu pamięcią
- tablica stronicowania wymaga dodatkowej przestrzeni w pamięci

Segmentacja pamięci



https://www.tutorialspoint.com/operating_system/images/segment_map_table.jpg

Segmentacja pamięci

- zadania otrzymują kilka segmentów różnej wielkości, każdy dla swojego modułu

-poszczególne segmenty nie muszą przylegać do sobie podczas samego wykonania procesu, co powoduje możliwość umieszczenia ich w możliwie optymalnej przestrzeni pamięci.

- działa podobnie do stronicowania

- tablica segmentacji zawiera informacje o starcie segmentu i jego wielkości, a także o aktualnie wolnej pamięci

Pamięć wirtualna

- system posiada możliwość adresacji (wykorzytania) większej ilości pamięci niż wynosi ilość pamięci głównej komputera

- rolę tego typu pamięci może pełnić np. magazyn danych (dysk twardy, SSD)

- tego typu rozwiązanie nazywamy pamięci wirtualną - adresujemy nieistniejącą pamięć główną

- adresowanie to działa dla programów ze zwiększonym zapotrzebowaniem na pamięć (programy graficzne, operacje wejścia/wyjścia)

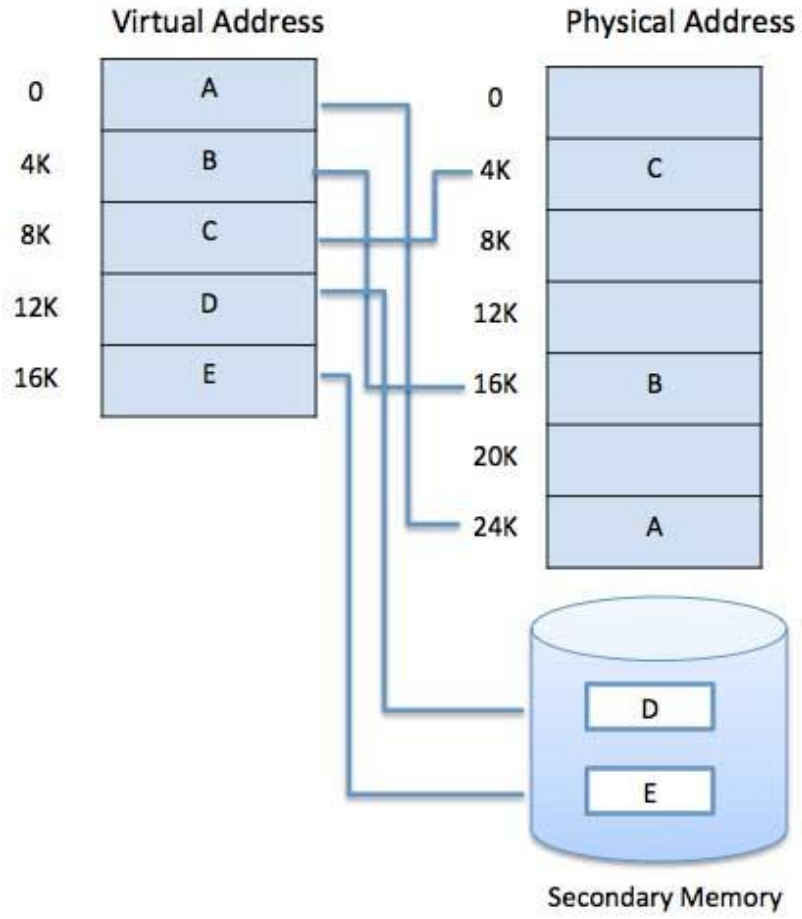
- zapisane dane w pamięci wirtualnej muszą mieć przełożenie na pamięć główną (tzw. mapowanie)

- tryb ten nazywany jest nazywany trybem chronionym (naprzeciw trybu rzeczywistego)

Pamięć wirtualna – kiedy jest używana

- obsługa błędów w przypadku wystąpienia błędów w danych – kod warunkowy
- określone elementy programu nie są często używane
- tablice pamięci programów mają przypisane stałe wielkości przestrzeni adresowej chociaż wykorzystują tylko małą część tejże pamięci
- możliwość wykonania programu, który może być wykonywany na danych tylko częściowo znajdujących się w pamięci
- mniejsza liczba operacji wejścia/wyjścia celem obsługi programów różnych użytkowników
- program nie może być ograniczony wielkością pamięci fizycznej
- można uruchomić więcej programów i procesów, tym samym zwiększając wykorzystanie innych podzespołów komputera

Pamięć wirtualna – koncepcja mapowania



https://www.tutorialspoint.com/operating_system/images/virtual_memory.jpg

Pamięć wirtualna - żądanie stronicowania

- procesy przechowują dane w pamięci dodatkowej i ładują strony (zbiory) pamięci na żądanie

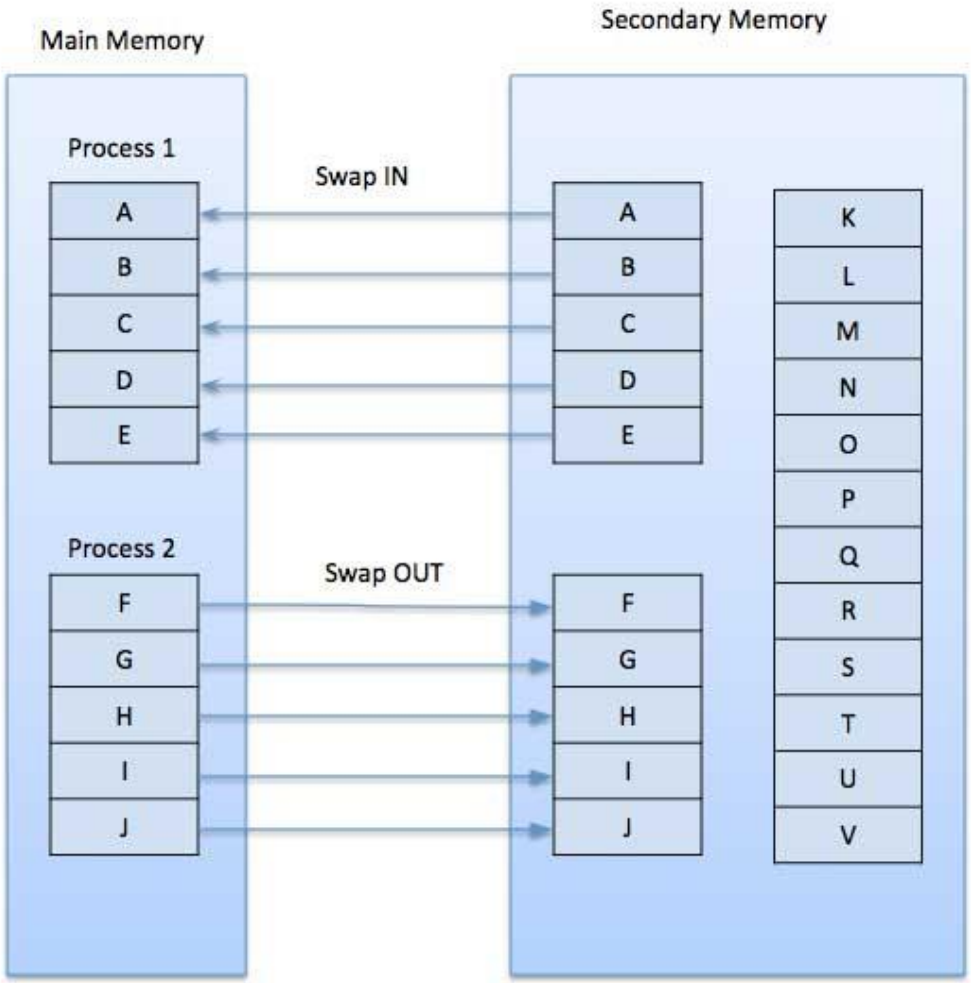
- nie zachodzi tutaj proces kopiowania programu na pamięć dodatkową ani pamięci nowego programu do pamięci głównej

- po załadowaniu pierwszej strony nowy program po prostu jest wykonywany; pozostałe strony są dopasowywane w locie

- jeżeli w danej chwili program odwołuje się do danych, które zostały z pamięci głównej usunięte procesor traktuje to jako wyjątek "błąd stronicowania"

- w tym momencie kontrolę nad pamięcią przejmuje system operacyjny celem zlokalizowania i załadowania brakującej strony pamięci do pamięci głównej

Żądanie stronicowania



https://www.tutorialspoint.com/operating_system/images/demand_paging.jpg

Żądanie stronicowania - podsumowanie

- duża przestrzeń pamięciowa
- zwiększona efektywność użytkowania pamięci operacyjnej
- brak limitu dla jednoczesnej obsługi wielu programów
- duży narzut liczby tabel przy obsłudze przerwań stronicowania (w porównaniu np. do prostych technik stronizowania)

Algorytmy zamiany pamięci

- algorytmy te wykorzystywane są przez system operacyjny do obsługi zamiany stron pamięci

- stronicowanie to obsługa zamiany stron w chwili, gdy następuje wyjątek błędu strony, zaś strona nie może zostać alokowana ze względu na jej brak lub ilość wolnych stron jest mniejsza od wymaganej ilości stron

- algorytm zamiany pamięci musi działać w taki sposób, by wykonać operację podmiany pamięci w jak najkrótszym czasie

- algorytm ma określoną wiedzę na temat dostępności stron

- na podstawie tej wiedzy próbuje wybrać strony do zastąpienia przy jednoczesnej minimalizacji skutków tej podmiany

Wzorcowy ciąg

- ciąg wzorcowy to odniesienie do ciągu w pamięci

- ciąg wzorcowy jest tworzony/generowany syntetycznie lub poprzez kopiowanie zachowania określonego systemu przy zapisie do pamięci

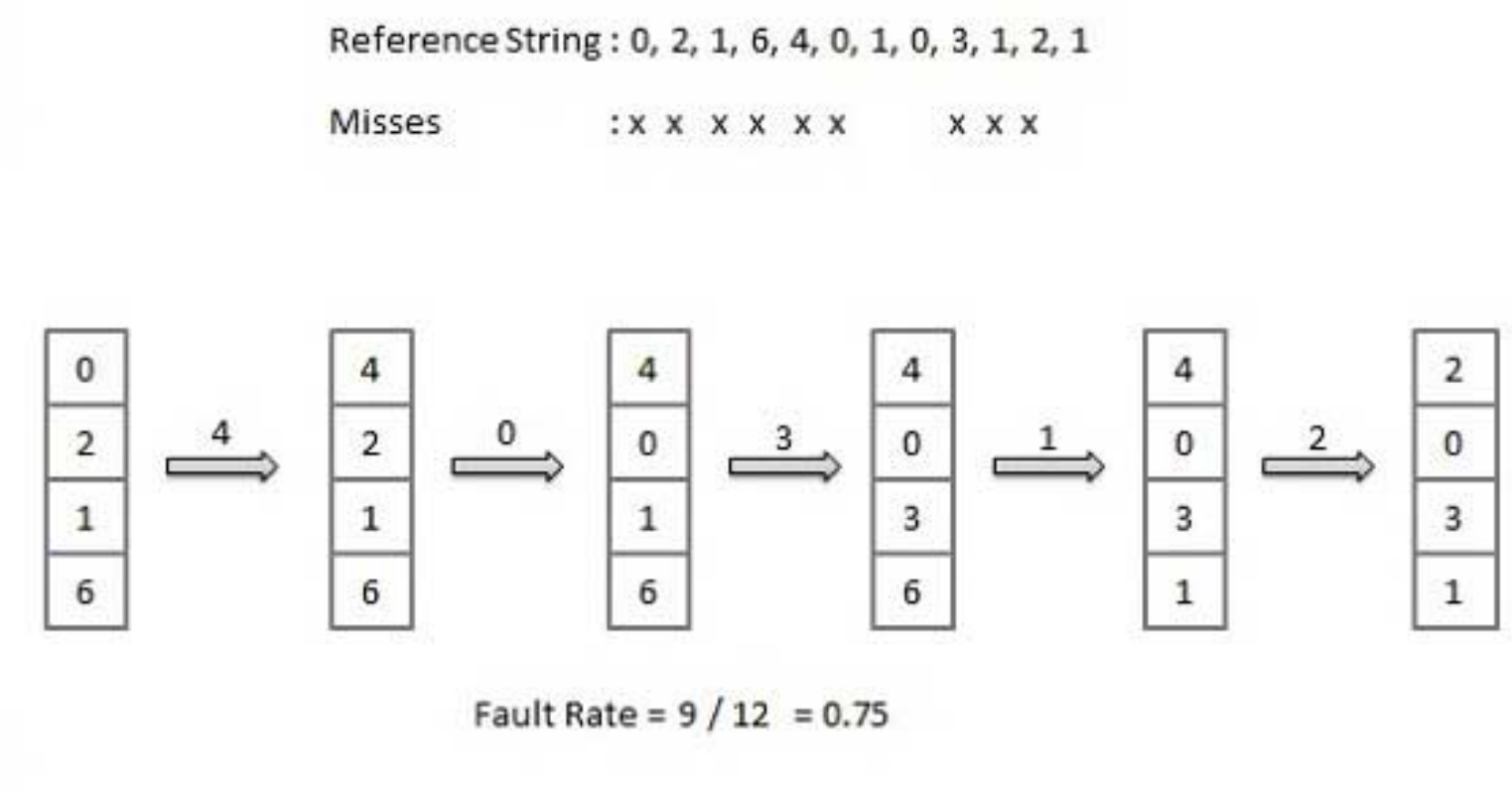
- dla określonego rozmiaru strony musimy znać tylko ilość stron, nie zaś każdy adres

- jeżeli mamy odniesienie do strony p to każde odwołanie do tej strony nie spowoduje błędu stronicowania.

- powoduje to fakt, że po pierwszym użyciu strona pozostanie w pamięci

- jeżeli założyły kolejny ciąg adresów: 1452, 8992, 1001, 98, 18782, 533, 2718, zaś wielkość strony na 1000, to nasz ciąg wzorcowy będzie wynosił: 1,8,1,0,18,0,2

Algorytm FIFO

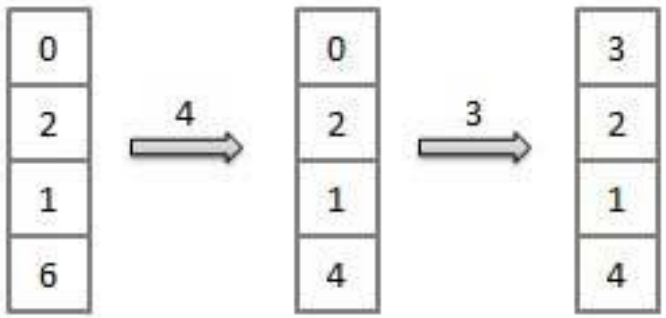


https://www.tutorialspoint.com/operating_system/images/fifo.jpg

Optymalne stronicowanie

Reference String : 0, 2, 1, 6, 4, 0, 1, 0, 3, 1, 2, 1

Misses : x x x x x x



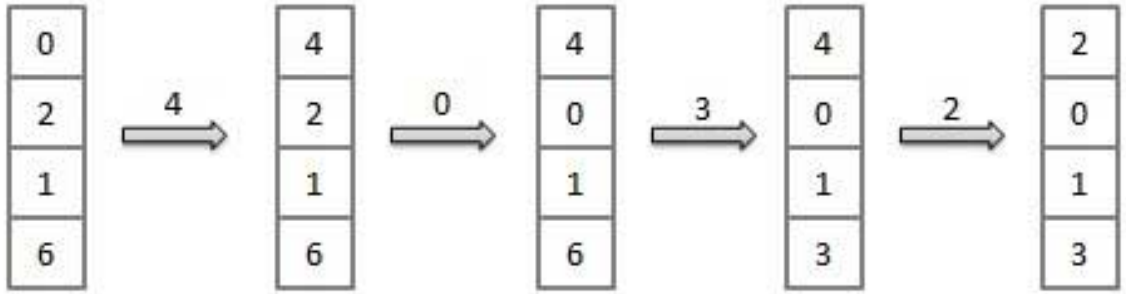
Fault Rate = 6 / 12 = 0.50

https://www.tutorialspoint.com/operating_system/images/opr.jpg

Algorytm LRU (najmniejszego użycia)

Reference String : 0, 2, 1, 6, 4, 0, 1, 0, 3, 1, 2, 1

Misses : x x x x x x x x



Fault Rate = 8 / 12 = 0.67

https://www.tutorialspoint.com/operating_system/images/lru.jpg

Buforowanie stron

- trzymanie puli wolnych ramek

- przy błędzie stronicowania wybieranie strony do zastąpienia

Działanie algorytmu (uproszczone):

- zapis nowej stronu do ramki z puli, zaznaczenie tabeli strony i restart procesu

- zapis szkicu strony poza dysk i umieszczenie jej w ramce z zastąpioną stroną w wolnej puli

Algorytm LFU (najrzadsza częstotliwość używania)

- strona z najmniejszą ilością użycia jest typowana do zastąpienia

- algorytm jest podatny na efekt intensywnego używania strony podczas startu programu i późniejszego nieużywania danych w dalszej części procesu

Algorytm MFU (najczęstszego użycia)

- działa na zasadzie przeciwnej do LFU

- jeżeli strona ma mały licznik użycia to po prostu została tyle co załadowana do pamięci

- usuwane są strony z najwyższym licznikiem

Problemy synchronizacji

- synchronizacja to koordynacja co najmniej dwóch niezależnych procesów/zjawisk

- synchronizacja wykonywana jest w czasie, jednak na różne sposoby (synchronizowanie cząstkowe lub dążenie do synchronizacji wyników)

- w systemach operacyjnych często występuje potrzeba synchronizacji wielu operacji

- przykładowo plik, z którego korzysta wiele aplikacji, wymaga synchronizowania dostępu do jego zawartości (sekwencja nie wchodzi w grę jeżeli programy mogą działać w niezależnych przydziałach czasu)

Problem producenta i konsumenta

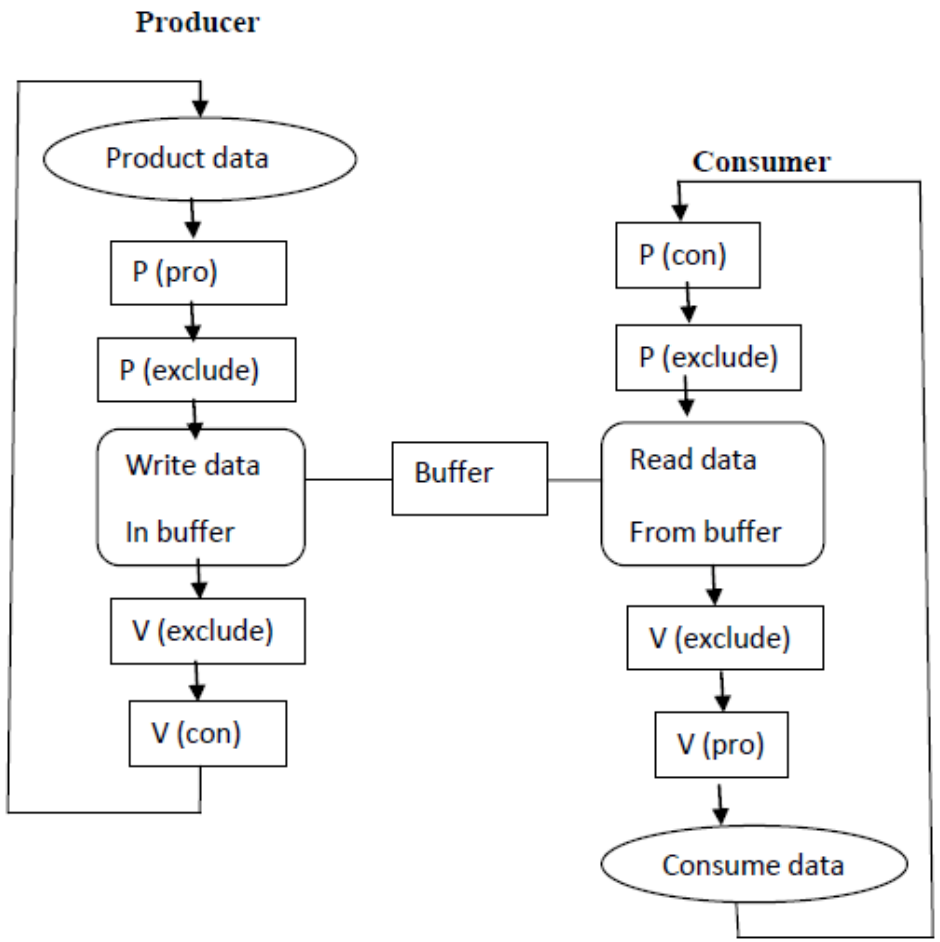
- producent i konsument to programy, które odpowiednio dodają dane (producent) i operują na tych danych (konsument)

- oba programy współdzielą pomiędzy sobą zasoby danych, lokalizowane w przestrzeni pamięci głównej (bufor) lub w pamięci masowej (plik)

- producent, wedle idei, powinien stale dostarczać dane, zaś konsument stale je odbierać

- problem pojawia się w chwili, gdy konsument nie ma jak odebrać danych (przetwarza poprzednie) bądź brakuje mu danych do pobrania (producent nie zdążył wytworzyć nowych danych)

Producent i komsument



https://www.researchgate.net/profile/Bidush-Sahoo/publication/309193487/figure/fig2/AS:418127058882563@1476700671875/Producer-consumer-problem.png

Problem czytelników i pisarzy

- czytelnik to proces nie dokonujący zmian w treści (zawartości) pliku(ów)

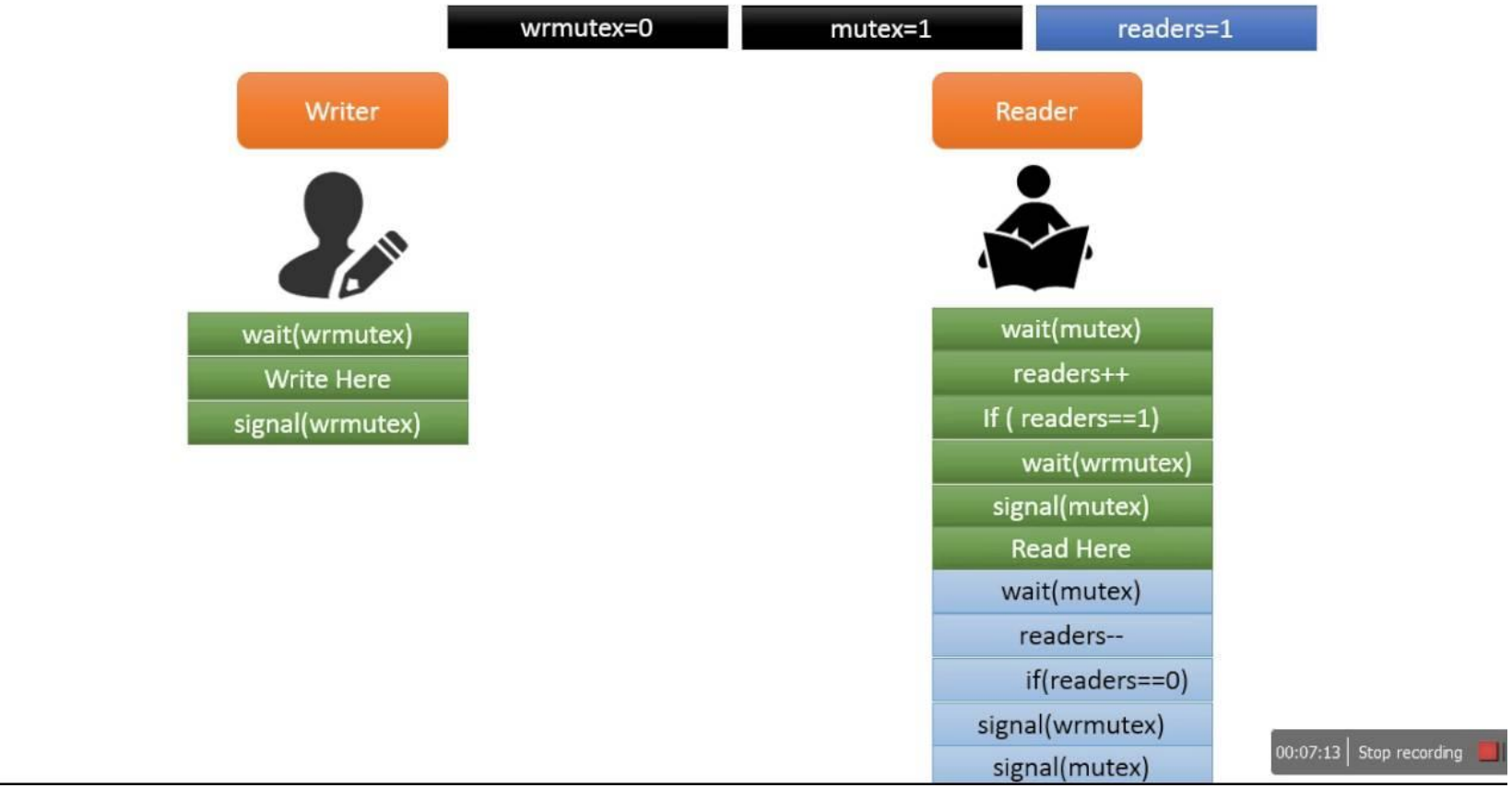
- pisarz to proces tworzący dane

- czytelnicy mogą odczytywać dane niezależnie od siebie, pisarz wymaga dostępu do danych na wyłączność

- gdyby pisarz otrzymał dostęp z czytelnikami lub innym pisarzem(ami), pojawił by się problem integralności danych

-

Czytelnik i pisarz



https://i.ytimg.com/vi/J60NK9F-c7A/maxresdefault.jpg

Materiały i źródła

- https://www.tutorialspoint.com/operating_system

- https://en.wikipedia.org/wiki/Producer%E2%80%93consumer_problem

- https://www.geeksforgeeks.org/readers-writers-problem-set-1-introduction-and-readers-preference-solution/

- Abraham Silberschatz, Greg Gagne, Peter B. Galvin, "Podstawy systemów operacyjnych" Tom I, Warszawa, 10 (1 w PWN), 2021