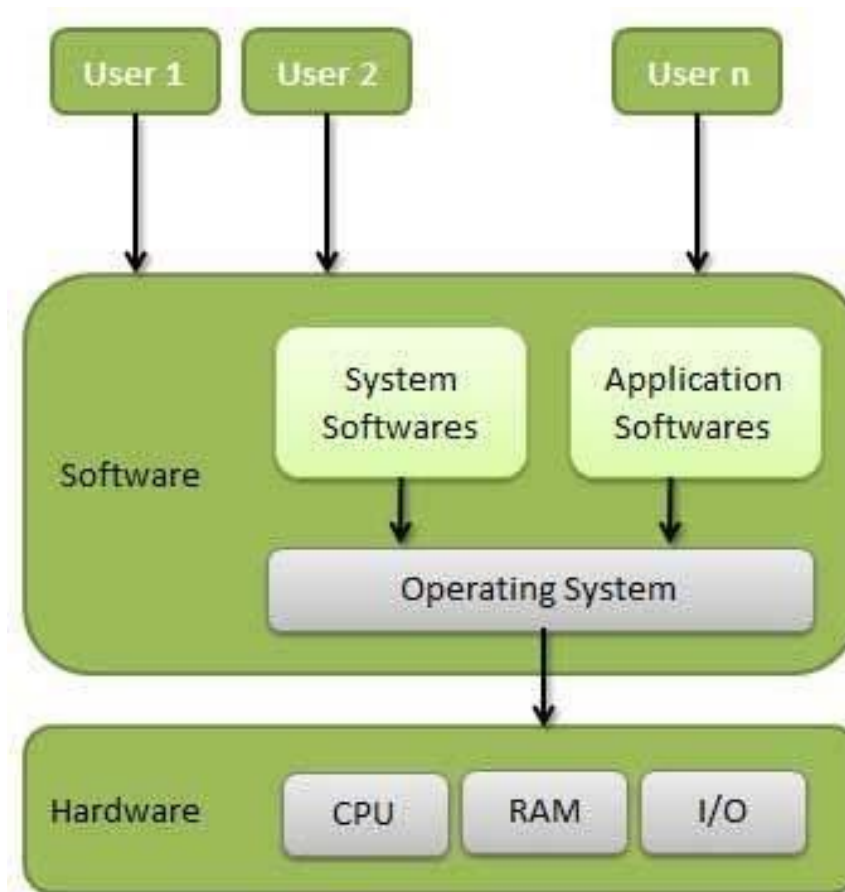


# Systemy operacyjne

Procesy, synchronizacja, przetwarzanie

# System Operacyjny



# 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
- sprzęt jest często w stanie jałowym
- ciężko jest zaplanować priorytetyzację zadań

# System operacyjny z podziałem czasu

- działa jako pojedynczy 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 wieloprocessorowe wielomaszynowe
- obsługa wielu aplikacji czasu rzeczywistego wielu użytkowników
- komunikacja następuje poprzez różnego rodzaju sprzęgi łączące
- redukcja przestoju 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 eliminacji 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 mechanizmy 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 dostarczyć 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 powinien 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.

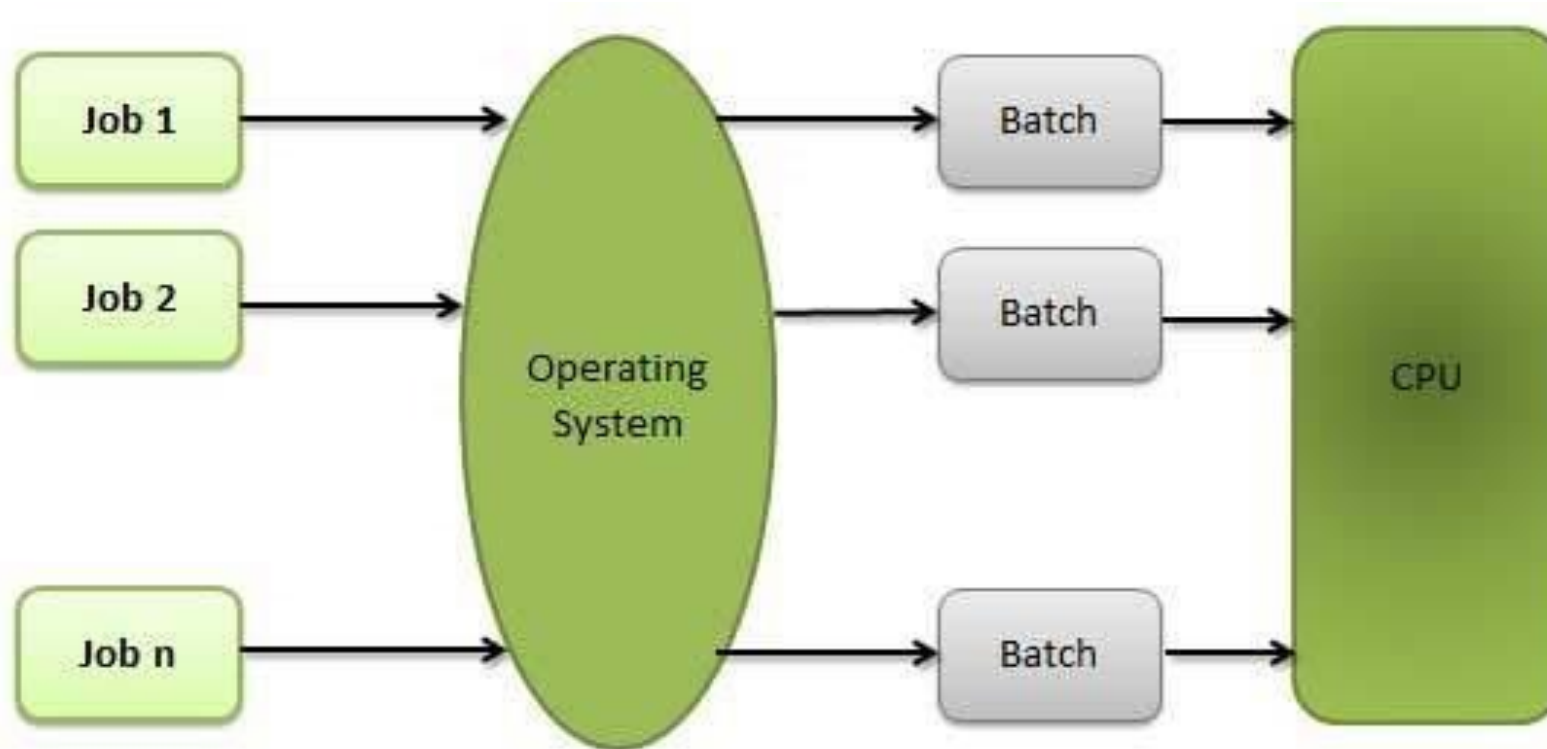
Odpowiednie algorytmy zarządzania przydziałem do zasobów (mikroprocesora) pozwalają na lepsze wykorzystanie dostępnych zasobów.

# Ochrona

System powinien zapewniać ochronę zarówno przed 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](https://www.tutorialspoint.com/operating_system/images/batch_processing.jpg)

# Przetwarzanie wsadowe

Wprowadza automatyzację działania zadań, poprzez predefiniowane sekwencje komend.

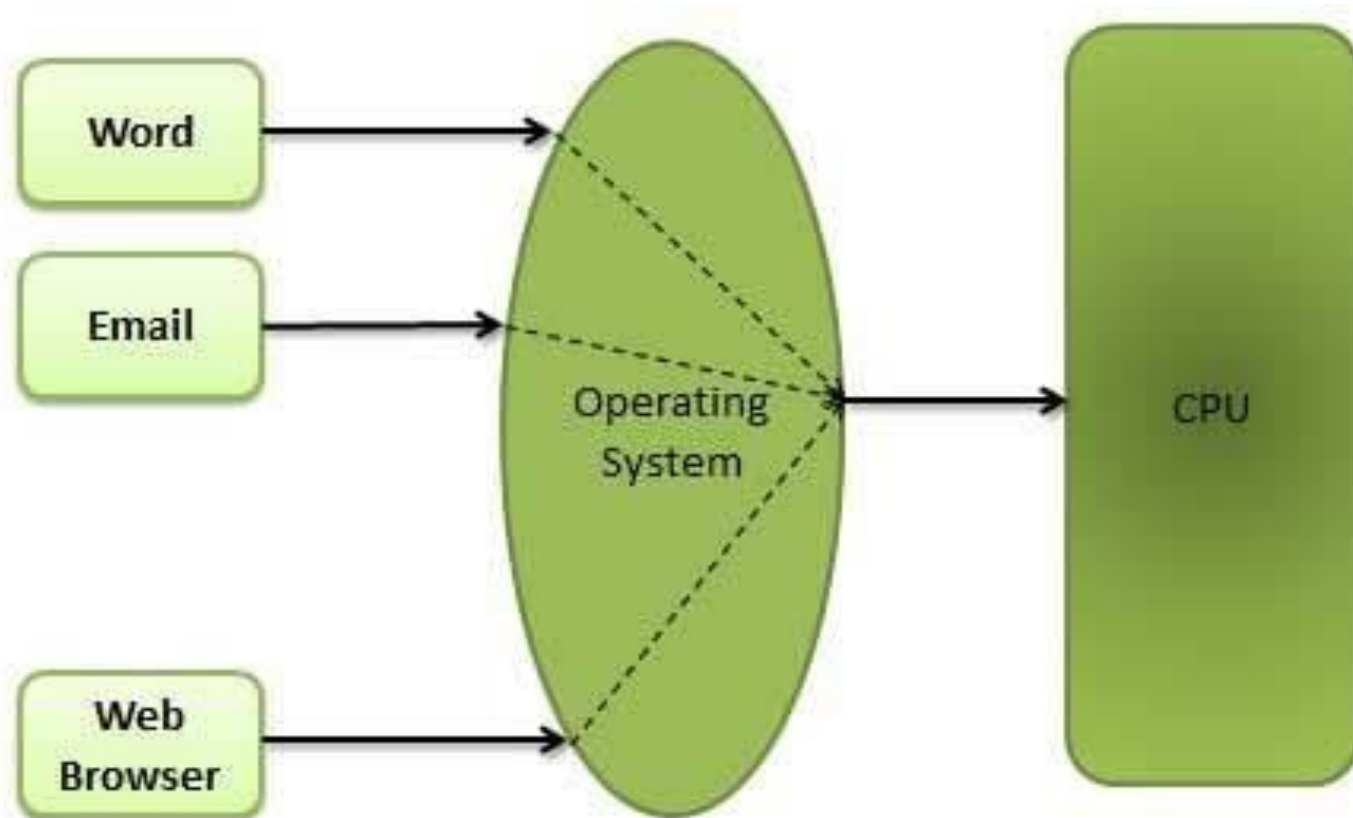
Najczęściej działa na zasadzie kolejkowania.

Po zakończeniu pracy pamięć operacyjna jest automatycznie zwalniana.

Niekiedy działania mogą wykonywać się wadliwie.

Może następować tzw. Zagłodzenie procesów.

# Wielozadaniowość



[https://www.tutorialspoint.com/operating\\_system/images/multitasking.jpg](https://www.tutorialspoint.com/operating_system/images/multitasking.jpg)

# Wielozadaniowość



[https://www.tutorialspoint.com/operating\\_system/images/memory\\_layout.jpg](https://www.tutorialspoint.com/operating_system/images/memory_layout.jpg)

# Wielozadaniowość

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

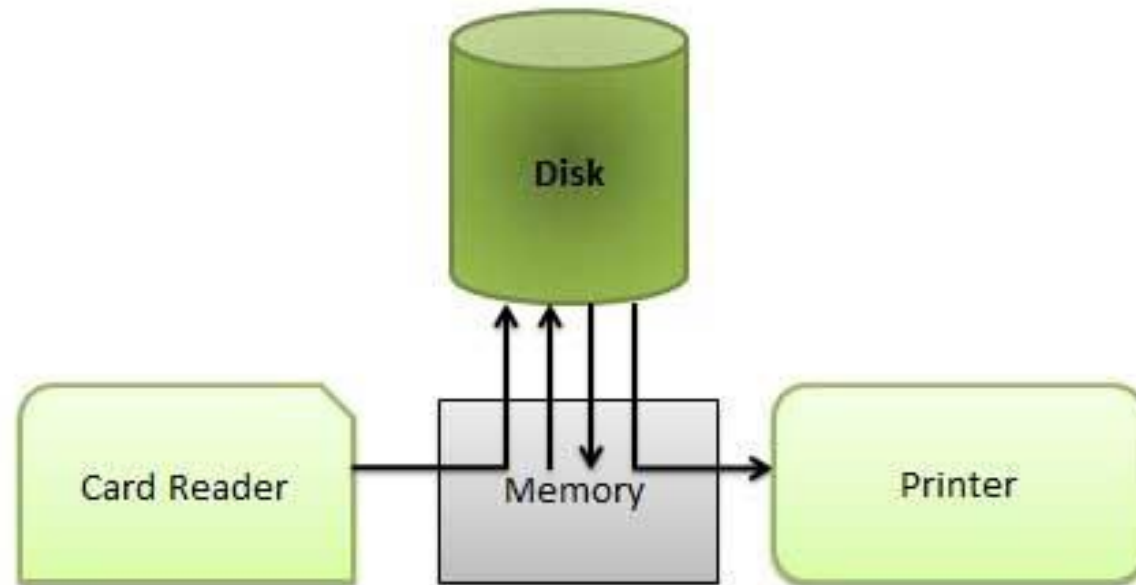
Wielozadaniowość pozwala na osiągnięcie tzw. interaktywności z użytkownikiem.

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 procesy (np. Długo trwające), by ostatecznie dopasować wykorzystanie CPU dla pozostałych, uruchomionych procesów.

# Buforowanie



[https://www.tutorialspoint.com/operating\\_system/images/spooling.jpg](https://www.tutorialspoint.com/operating_system/images/spooling.jpg)

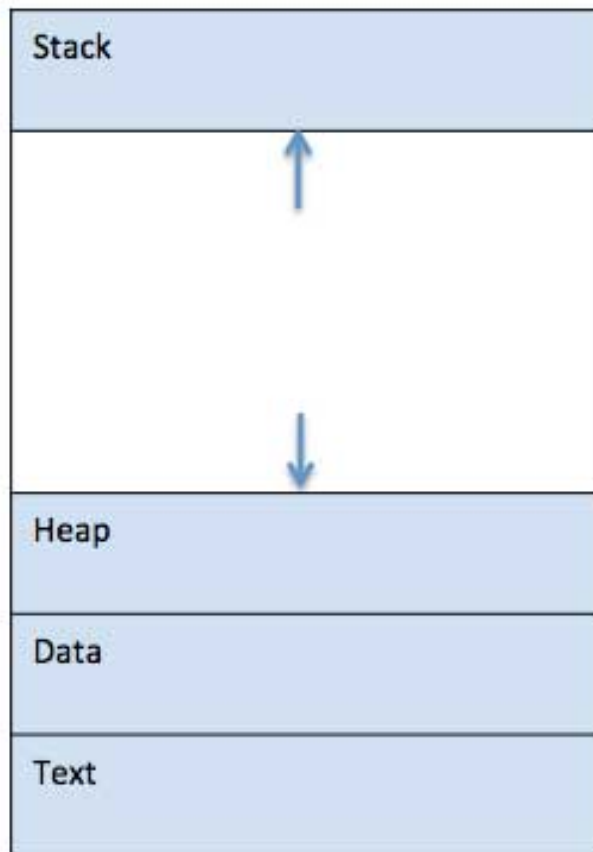
# Buforowanie

Obsługa urządzeń posiadających różne czasy dostępu do różnych nośników.

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

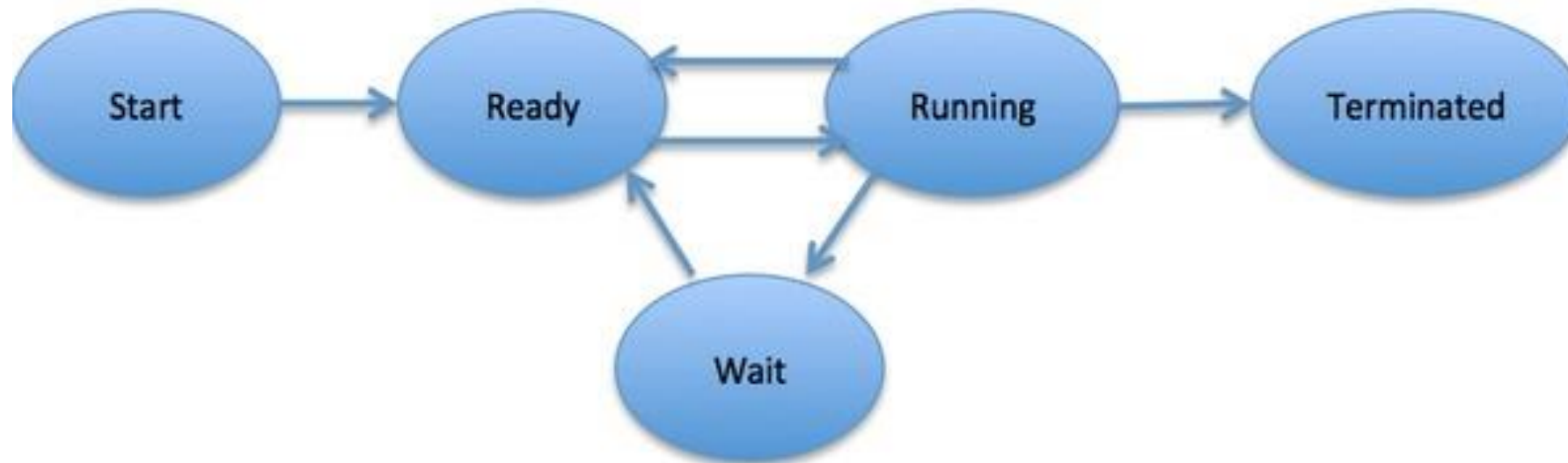
Pozwala na równoległe przetwarzania, np. Wczytywanie danych z kilku różnych nośników, zapisywanie danych na innym nośniku i jednocześnie drukowaniu wyników na kartce papieru.

# Działanie podprogramów



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

# Cykl życia procesu



[https://www.tutorialspoint.com/operating\\_system/images/process\\_state.jpg](https://www.tutorialspoint.com/operating_system/images/process_state.jpg)

# Cykl życia procesu

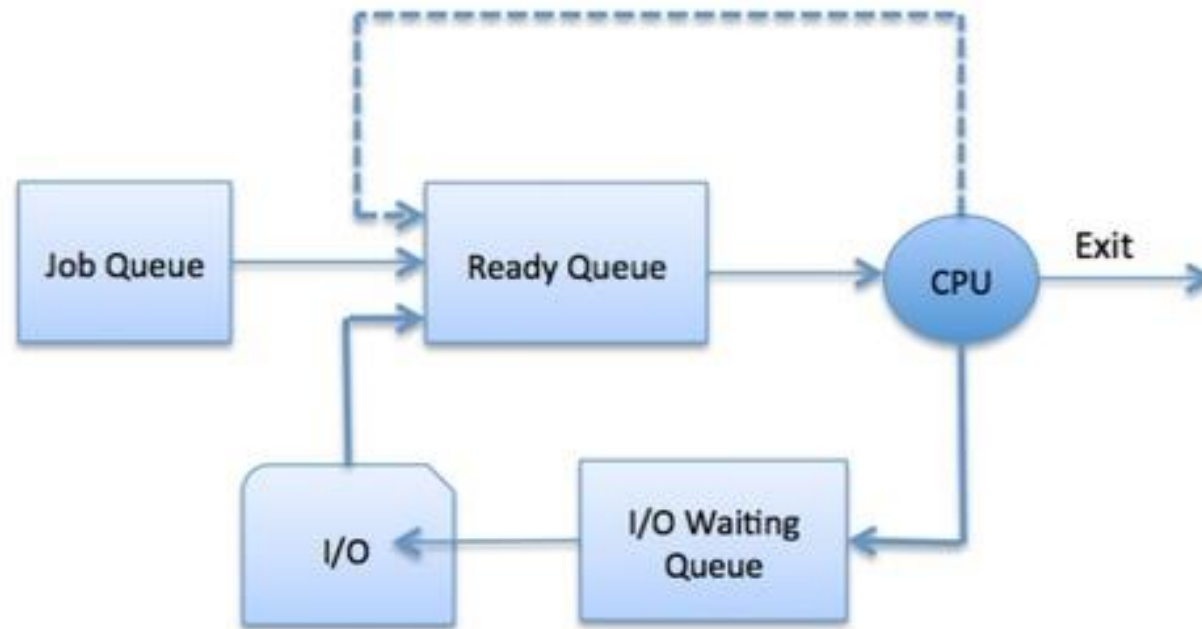
Start	Inicjalizacja procesu (i jego wszelkich wartości)
Gotowość	Oczekiwanie na dostęp do procesora. Ten stan jest osiągalny gdy proces jest po fazie startu lub przerwania działania (znajduje się w fazie oczekiwania) przez planistę systemu.
Działanie	Proces wykonuje przypisane mu instrukcje na procesorze
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

Process ID
State
Pointer
Priority
Program counter
CPU registers
I/O information
Accounting information
etc....

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

# Kolejka procesów



[https://www.tutorialspoint.com/operating\\_system/images/queuing\\_diagram.jpg](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

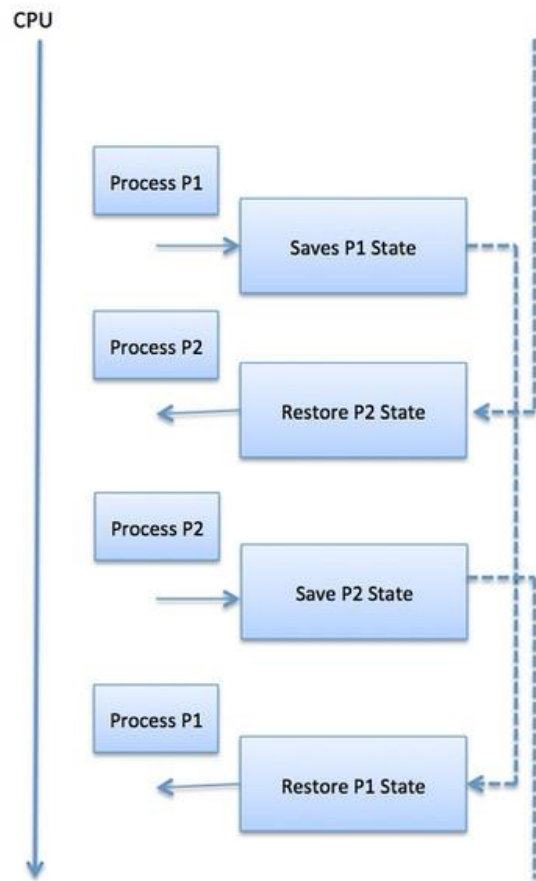
# Planista krótkoterminowy (Short Term Scheduler)

- 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

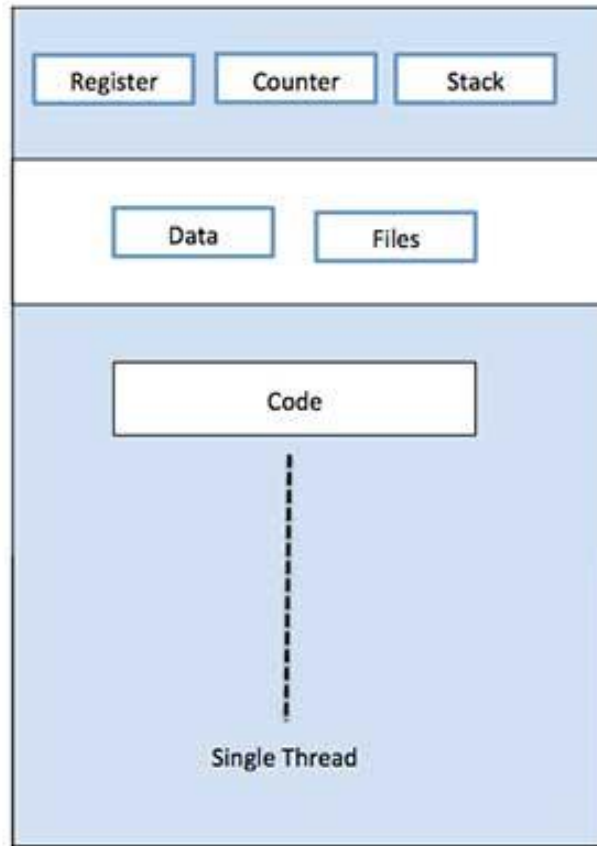
# Przełączanie kontekstowe



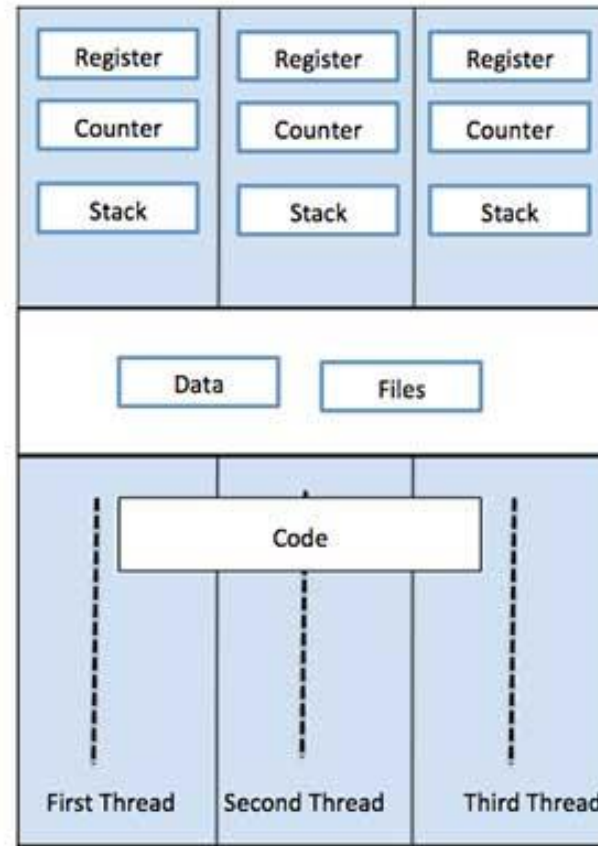
Informacje przechowywane podczas przełączenia procesów:

- Licznik programu
- Planer
- Wartości bazowe i maksymalne rejestrów
- Aktualnie użytkowane rejestry
- Stany zmian
- Informacje o stanie I/O
- Wartości liczników

# Wielowątkowość



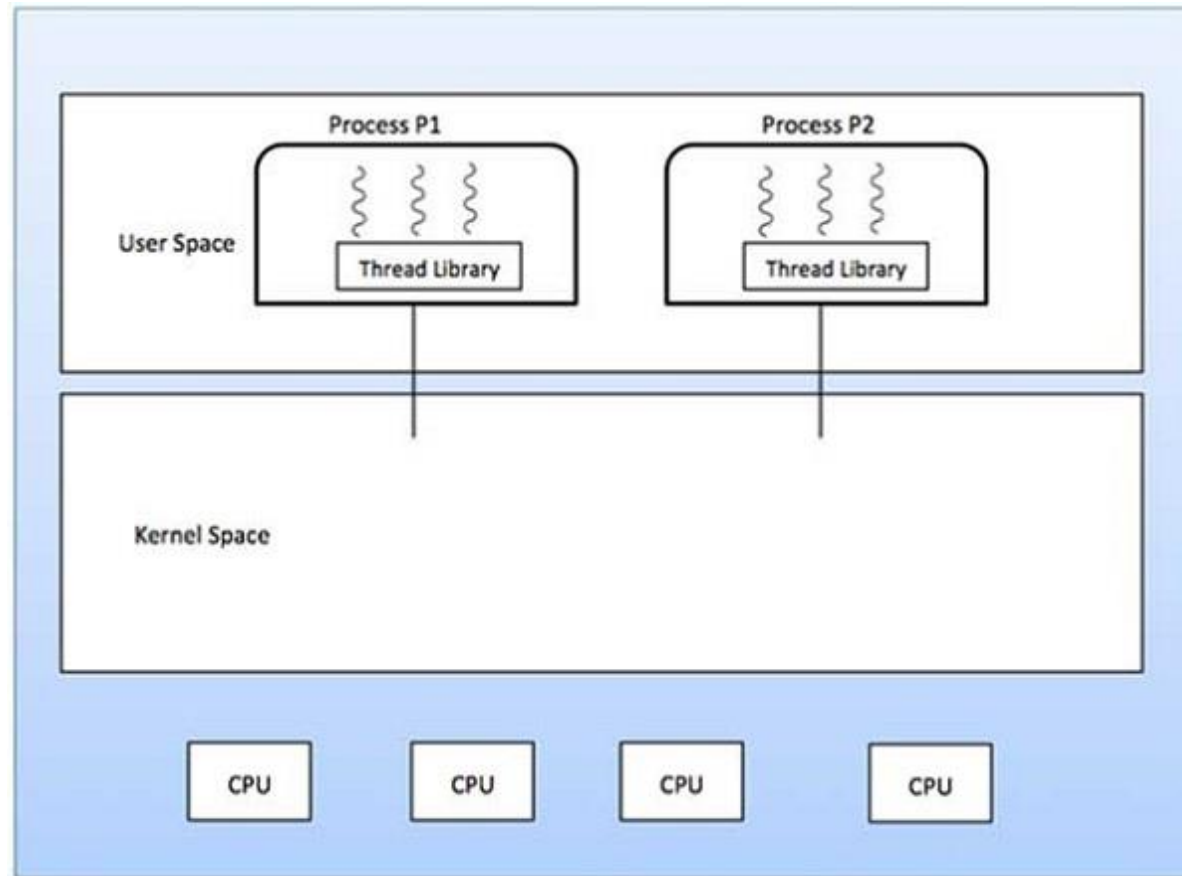
Single Process P with single thread



Single Process P with three threads

[https://www.tutorialspoint.com/operating\\_system/images/thread\\_processes.jpg](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](https://www.tutorialspoint.com/operating_system/images/user_threads.jpg)

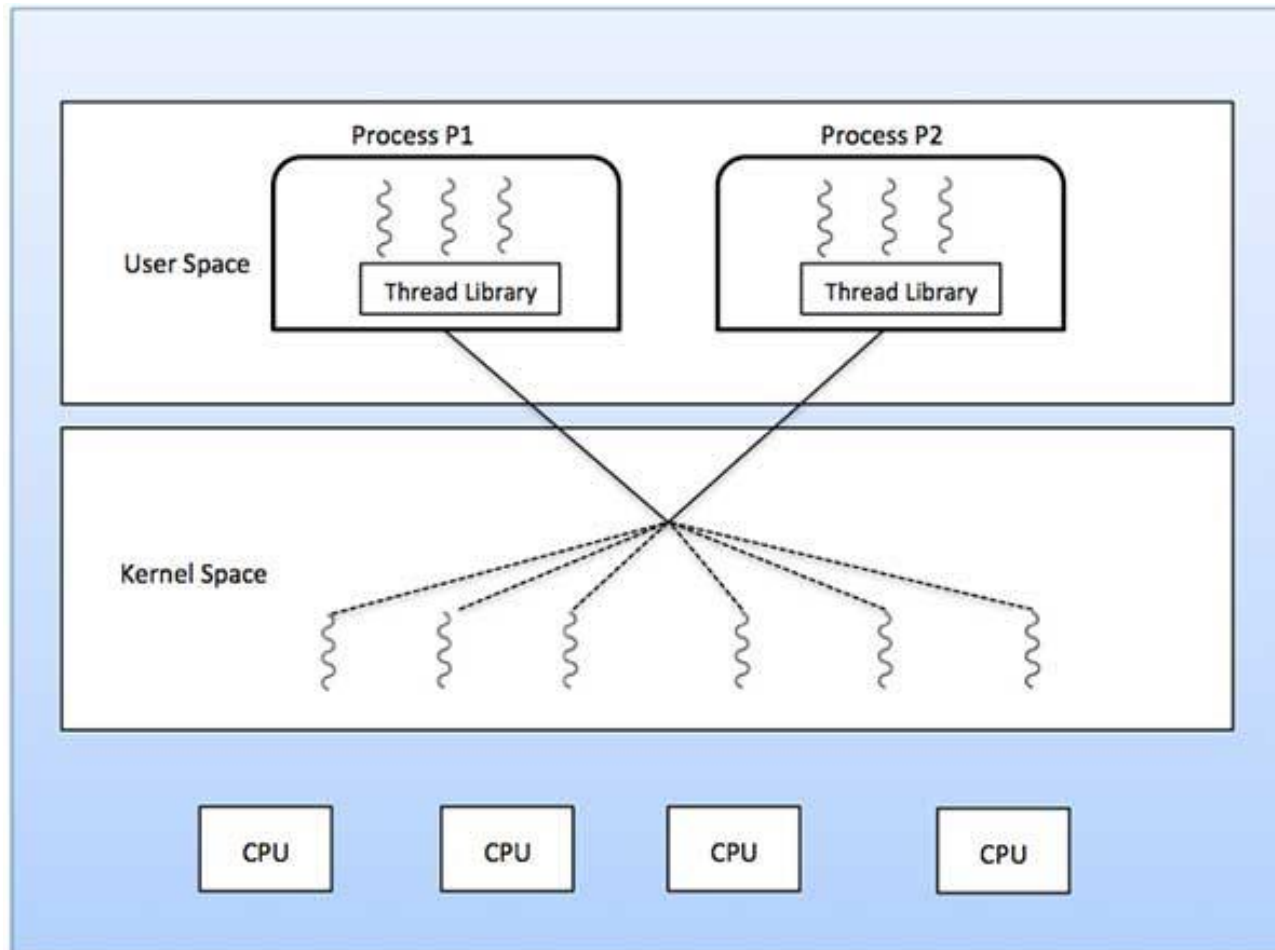
# Wątki użytkownika

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

# Wątki jądra

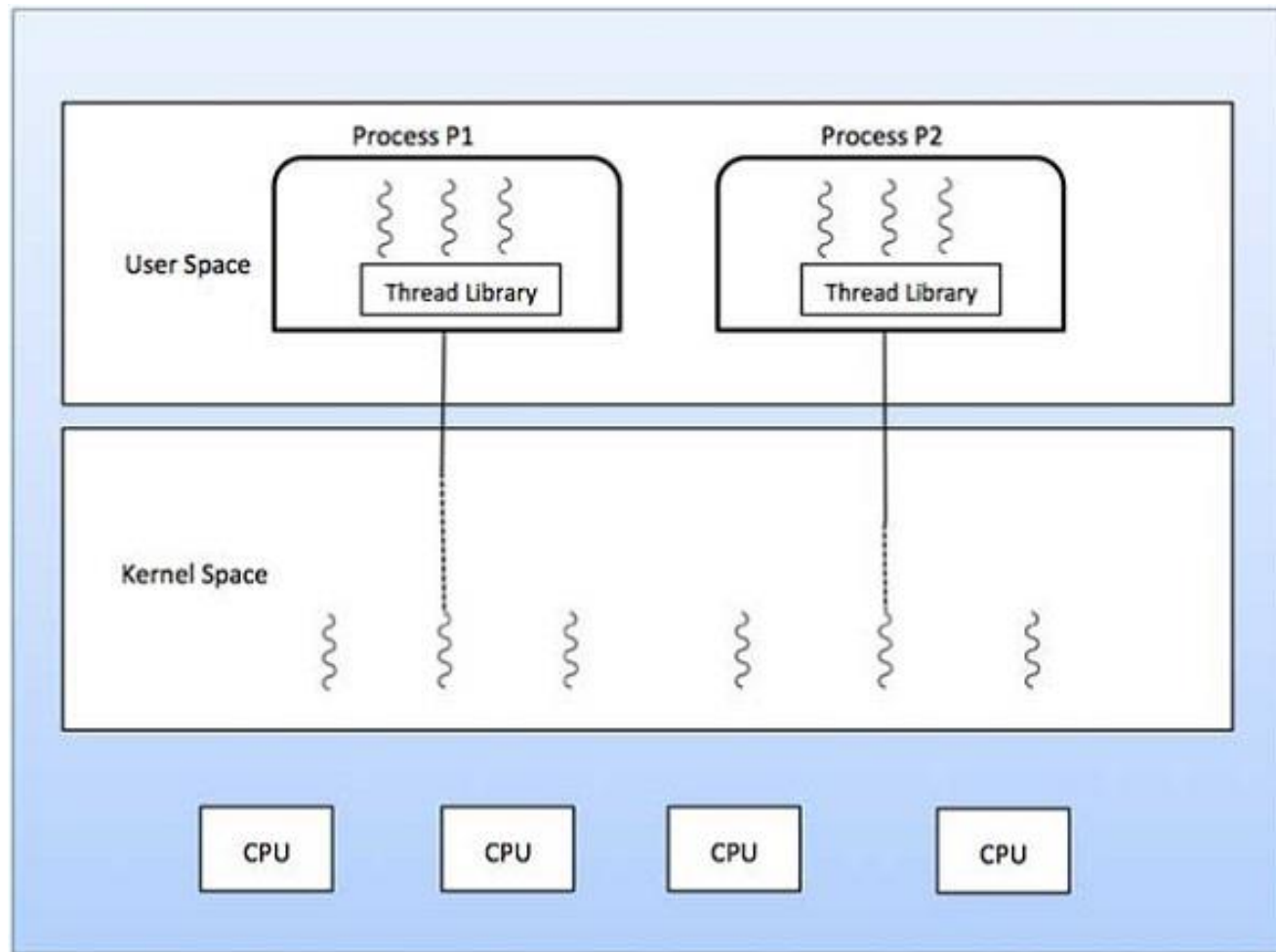
- jądro może planować przełączanie wątków pomiędzy poszczególnymi procesami, jednak wymaga to zawsze odwołania do jądra (mode switch)
- planista ma możliwość zarządzania każdym wątkiem wewnątrz procesu
- algorytmy jądra mogą być wielowątkowe
- niestety są one mniej wydajne (efektywne) przy tworzeniu i użytkowaniu niż wątki użytkownika

# Model wiele-do-wielu



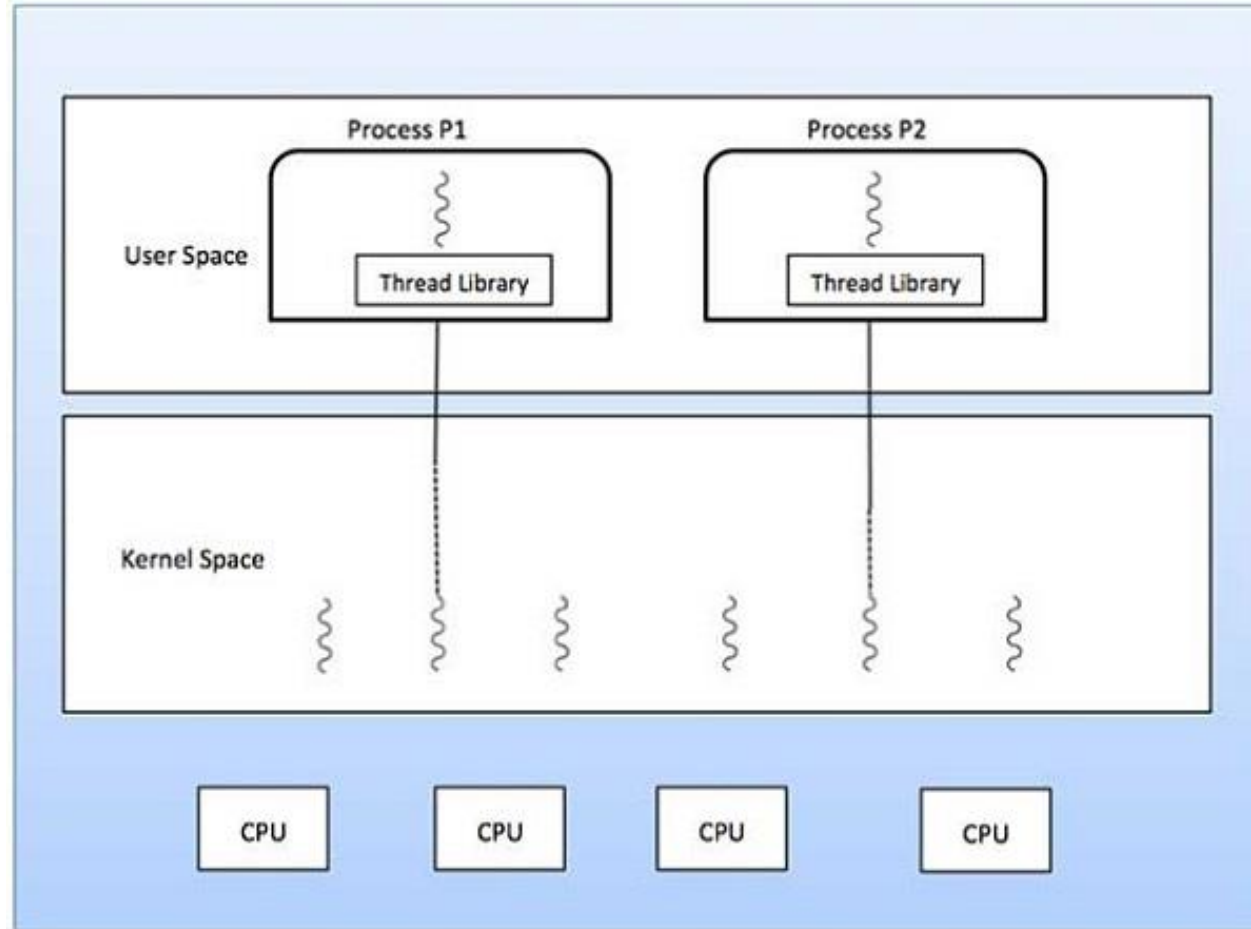
[https://www.tutorialspoint.com/operating\\_system/images/many\\_to\\_many.jpg](https://www.tutorialspoint.com/operating_system/images/many_to_many.jpg)

# Model wiele-do-jednego



[https://www.tutorialspoint.com/operating\\_system/images/many\\_to\\_one.jpg](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](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.

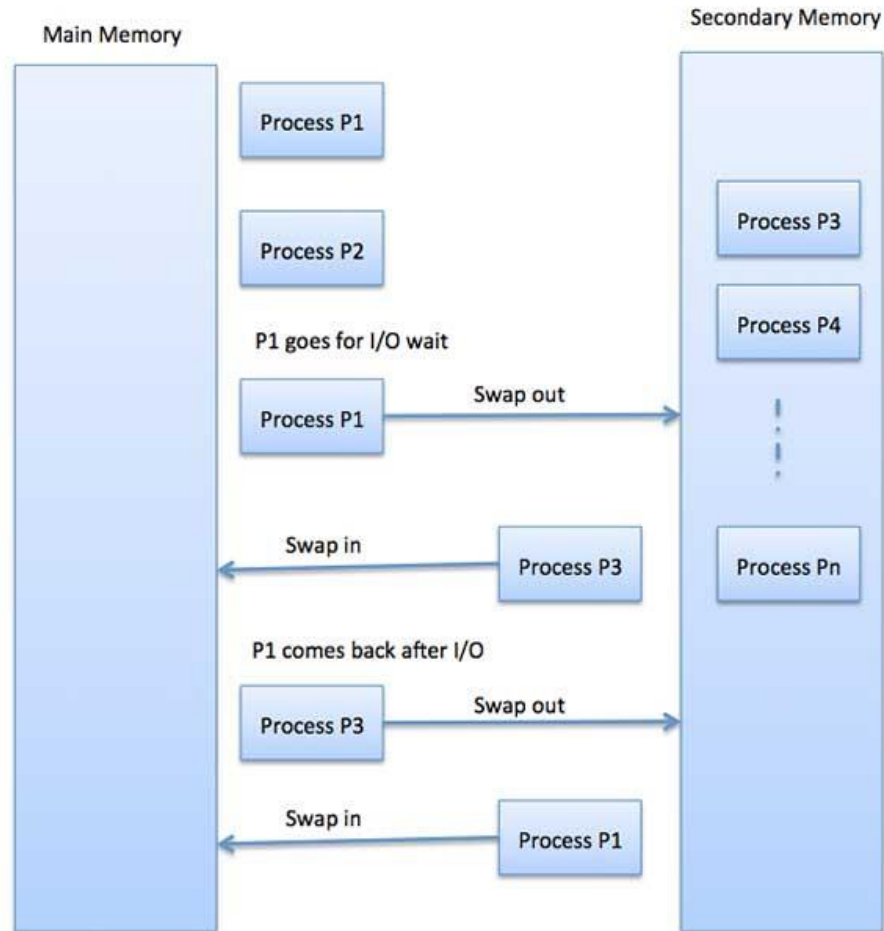
# Typy adresacji dostępne w systemach operacyjnych

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

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

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

# Przełączanie pamięci



# 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 (high memory). Pierwsza z nich zarezerwowana jest na główny kod systemu operacyjnego, druga zaś do użytku pozostałych procesów.

# Fragmentacja pamięci

Fragmented memory before compaction



Memory after compaction



[https://www.tutorialspoint.com/operating\\_system/images/memory\\_fragmentation.jpg](https://www.tutorialspoint.com/operating_system/images/memory_fragmentation.jpg)

# Rodzaje fragmentacji

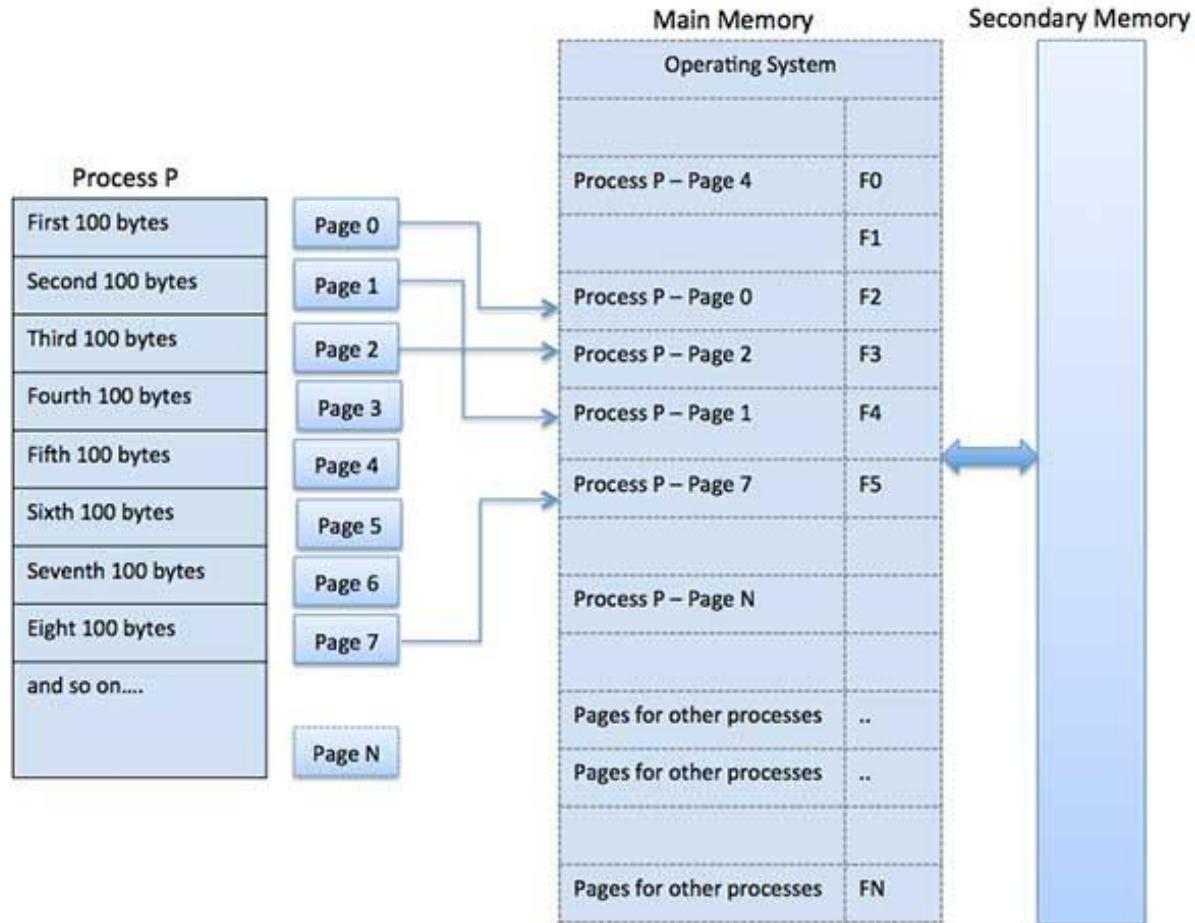
## **Fragmentacja zewnętrzna**

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

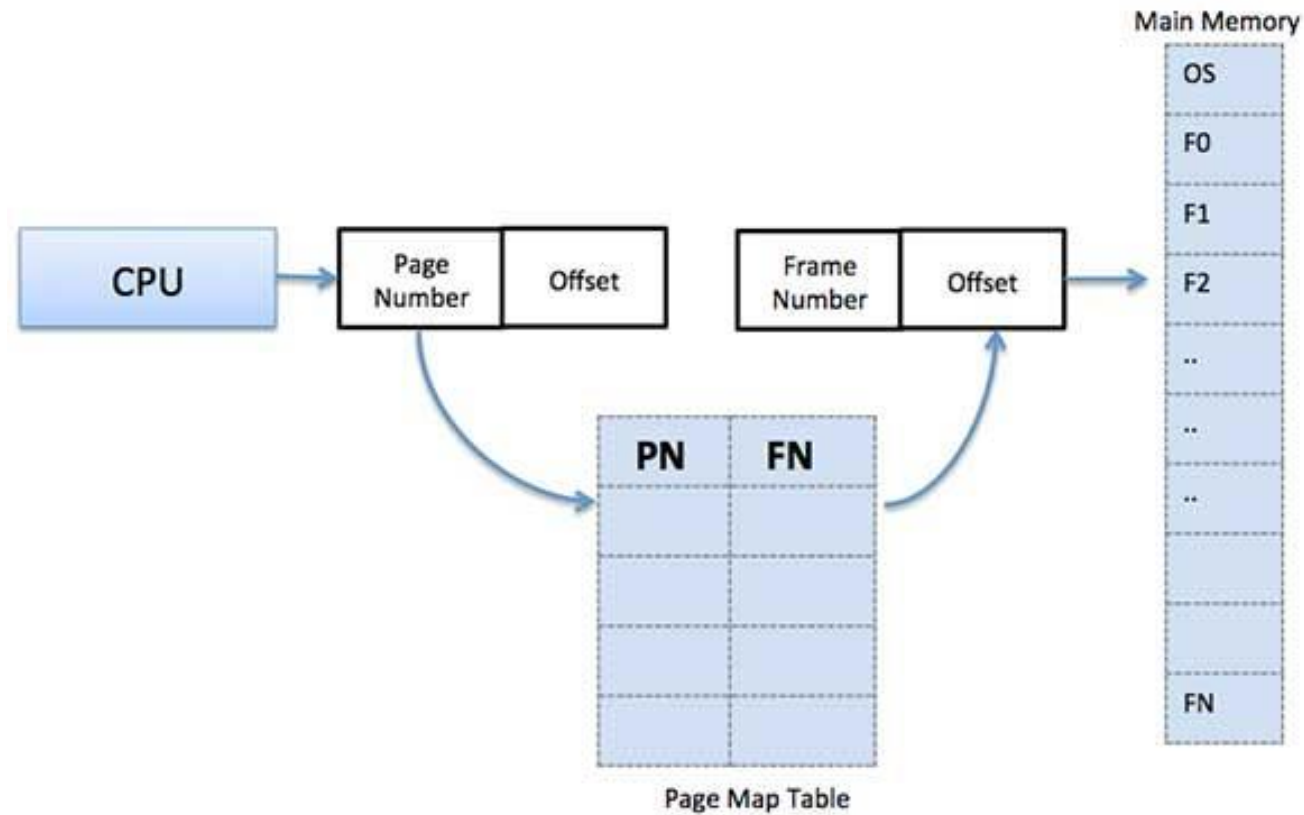


[https://www.tutorialspoint.com/operating\\_system/images/paging.jpg](https://www.tutorialspoint.com/operating_system/images/paging.jpg)

# 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)



# Tłumaczenie adresów

Adres strony nazywany jest adresem logicznym

$\text{Logical Address} = \text{Page number} + \text{page offset}$

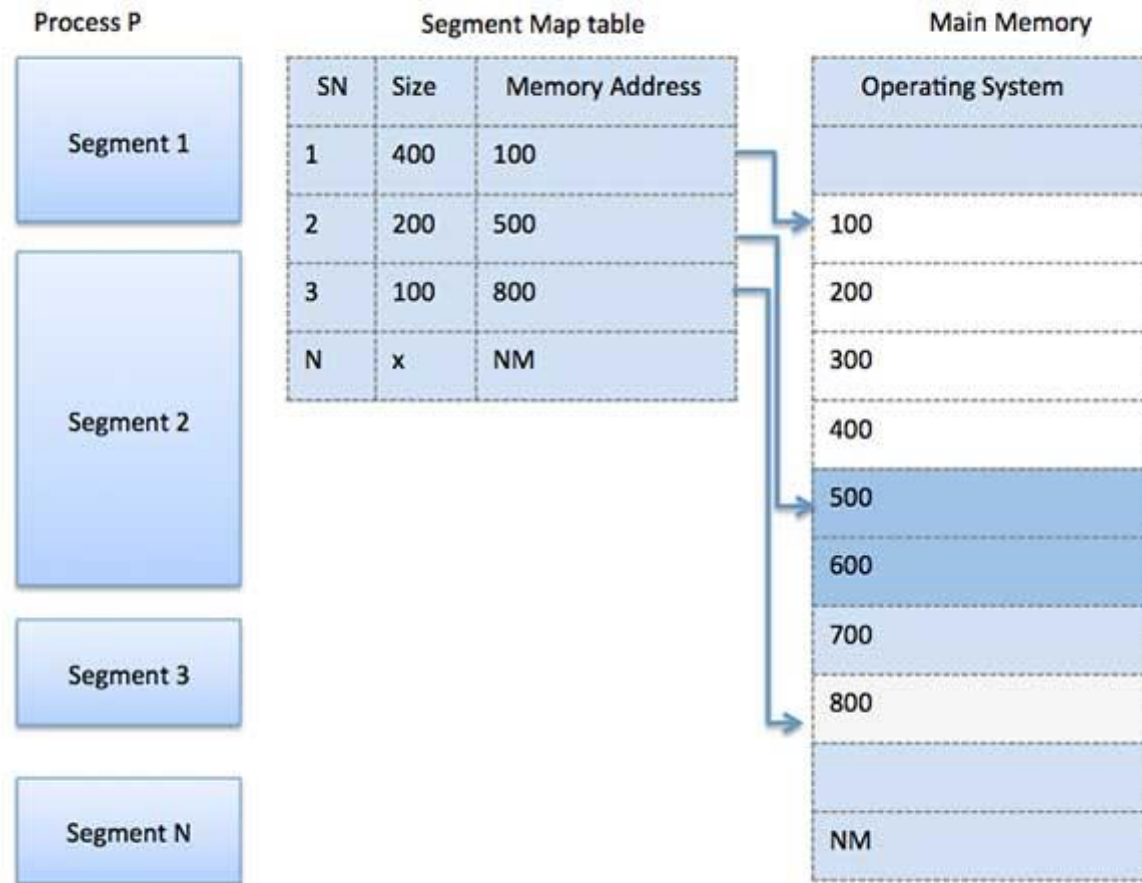
Adres ramki nazywany jest adresem fizycznym

$\text{Physical Address} = \text{Frame number} + \text{page offset}$

# 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](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 siebie 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

