19. Opisz rejestry CPU oraz PSW.

1. Rejestry procesora
    a. licznik programu (PC) - adres rozkazu do pobrania
    b. rejestr rozkazu (IR) - kod rozkazu
    c. rejestr adresowy pamięci (MAR) - adres lokacji
    d. rejestr buforowy pamięci (MBR) - dane do/z pamięci
    e. rejestry PSW - słowo stanu programu, informacje o stanie
2. PSW(bity, flagi)
    a. znak - bit znaku ostatniej operacji
    b. zero - wynik operacji = zero
    c. przeniesienie - przeniesienie w wielokrotnej precyzji
    d. równość - wynik porównania logicznego
    e. przepełnienie
    f. zezwolenie/blokowanie przerwań
    g. nadzorca - tryb systemu lub tryb użytkownika

20. Wymień kategorie urządzeń we/wy.

1. przeznaczone do odczytu przez człowieka
2. przeznaczone do odczytu przez maszynę
3. komunikacyjne

21. Wymień i opisz sposoby realizacji we/wy.

1. programowane – dane są wymieniane między procesorem a modułem we/wy, procesor czeka na zakończenie operacji we/wy,
2. sterowane przerwaniami – procesor wydaje operację we/wy i wykonuje dalsze rozkazy do momentu zakończenia operacji we/wy,
3. bezpośredni dostęp do pamięci – moduł we/wy wymienia dane bezpośrednio z pamięcią bez udziału procesora

22. Wymień i opisz metody dostępu do pamięci.

1. dostęp sekwencyjny
    a. dostęp liniowy blok po bloku w przód lub w tył,
    b. czas dostępu zależy od pozycji bloku względem pozycji bieżącej,
    c. np: taśmy,
2. dostęp bezpośredni
    a. każdy blok ma unikalny adres,
    b. czas dostępu realizowany przez skok do najbliższego otoczenia i sekwencyjne przeszukiwanie,
    c. np: dysk magnetyczny,
3. dostęp swobodny

    a. każda adresowalna lokacja w pamięci ma unikatowy, fizycznie wbudowany mechanizm adresowania,
    b. czas dostępu nie zależy od poprzednich operacji i jest stały
    c. np: RAM,
4. dostęp skojarzeniowy
    a. dane są lokalizowane raczej na podstawie porównania z ich zawartością niż na podstawie adresu,
    b. czas dostępu nie zależy od poprzednich operacji i jest stały,
    c. np: pamięć podręczna

23. Wymień rodzaje pamięci ze względu na własności fizyczne.

1. półprzewodnikowa;
2. magnetyczna;
3. magneto-optyczna;

24. Opisz zasadę lokalności odniesień.

1. zasada lokalności odniesień oznacza, że w czasie wykonania programu odwołania do danych i rozkazów mają tendencję do gromadzenia się
2. przyczyna: programy zwykle zawierają tablice deklaracji zmiennych oraz stałych i wiele pętli iteracyjnych i podprogramów
3. wykorzystanie zasady lokalności odniesień pozwala na zmniejszenie częstotliwości dostępu

25. Opisz działanie pamięci podręcznej.

1. Cache zawiera fragment pamięci głównej
2. Procesor sprawdza czy aktualnie potrzebne do wykonania rozkazu słowo z pamięci jest w cache'u
    a. jeśli nie, to blok pamięci o ustalonej liczbie K słów zawierający potrzebne słowo jest ściągnięty do pamięci podręcznej
3. Cache zawiera znaczniki identyfikujące bloki pamięci głównej

26. Opisz sposoby mapowania dla pamięci podręcznej.

1. Mapowanie bezpośrednie
    a. Każdy blok w pamięci głównej jest odwzorowywany na tylko jeden możliwy wiersz pamięci
        i. tzn. jeśli blok jest w cache'u to tylko w ściśle określonym miejscu
    b. Adres jest dzielony na dwie części
        i. najmniej znaczących w-bitów identyfikuje jednoznacznie słowo lub bajt w pamięci
        ii. najbardziej znaczących s-bitów określa jeden z 2^s bloków pamięci

            1. najbardziej znaczące bity są dzielone na pole wiersza złożone z r bitów oraz znacznik w postaci s-r bitów (najbardziej znacząca część)
2. Mapowanie skojarzeniowe
    a. Blok pamięci może zostać załadowany do dowolnego wiersza w cache'u
    b. Adres dzielimy na dwie części: znacznik i słowo
        i. znacznik jednoznacznie określa blok pamięci
    c. Aby stwierdzić czy blok jest w cache'u trzeba zbadać zgodność adresu ze znacznikiem każdego wiersza
    d. Kosztowna metoda zwłaszcza gdy rozmiar cache'a jest duży
        i. konieczność równoległego badania znaczników wszystkich wierszy w pamięci podręcznej
3. Mapowanie sekcyjno-skojarzeniowe
    a. k-drożne mapowanie sekcyjno-skojarzeniowe
    b. Cache jest dzielony na v sekcji
    c. Każda sekcja składa się z k wierszy
    d. Dany blok B może zostać odwzorowany na dowolny wiersz jakiejś sekcji i
        i. np. jeśli sekcja ma 2 wiersze
            1. 2 sposoby mapowania (mapowanie dwudrożne)
            2. blok może zostać umieszczony w jednym z dwu wierszy w jednej sekcji
        ii. np. jeśli adres sekcji jest 13 bitowy
            1. określa się numer bloku w pamięci modulo 213
            2. bloki 000000, 008000, 018000,...., FF8000 są mapowane na tę samą sekcję 0

44. Wymień składowe systemu operacyjnego.

1. Zarządzanie procesorami
2. Zarządzanie pamięcią operacyjną
3. Zarządzanie plikami
4. Zarządzanie systemem we/wy
5. Zarządzanie pamięcią pomocniczą
6. Praca sieciowa
7. System ochrony
8. System interpretacji poleceń

45. Wymień usługi systemu operacyjnego.

1. wykonanie programu - system powinien móc załadować program do pamięci i wykonać go,
2. operacje we/wy – program użytkowy nie wykonuje operacji we/wy bezpośrednio więc musi to oferować system
3. manipulowanie systemem plików - program musi mieć możliwość (pod kontrolą) do czytania, pisania, tworzenia i usuwania plików

4. komunikacja – wymiana informacji pomiędzy procesami wykonywanymi na tym samym lub zdalnym komputerze
    a. np. za pomocą pamięci dzielonej lub przekazywania komunikatów
5. wykrywanie błędów – zapewnienie prawidłowości działania komputera poprzez wykrywanie i obsługę wszystkich błędów w jednostce centralnej, pamięci operacyjnej, urządzeniach we/wy (np. błąd sumy kontrolnej) i w programie użytkownika (np. przekroczenie czasu)
6. dodatkowe funkcje systemu nie są przeznaczone do pomagania użytkownikowi, lecz do optymalizacji działania samego systemu:
    a. przydzielanie zasobów dla wielu użytkownikowi i wielu zadań w tym samym czasie
    b. rozliczanie – przechowywanie danych o tym, którzy użytkownicy i w jakim stopniu korzystają z poszczególnych zasobów komputera (statystyka użytkowania)
    c. ochrona – zapewnienie aby cały dostęp do zasobów systemu odbywał się pod kontrolą
        i. np. dostęp przez modem po podaniu hasła

46. Co to są wywołania (funkcje) systemowe?

1. funkcje systemowe tworzą interfejs między wykonywanym programem a systemem operacyjnym,
    a. dostępne na poziomie języka maszynowego (asemblera)
    b. pewne języki zastępują asembler w oprogramowaniu systemowym i umożliwiają bezpośrednie
    c. wykonywanie funkcji systemowych (np. C, C++, Bliss, PL/360, PERL)
    d. win32 API (Application Programmer Interface) - wielki zbiór procedur dostarczanych przez Microsoft, które umożliwiają realizację funkcji systemowych

47. Wymień podstawowe metody przekazywania parametrów między procesem a systemem.

1. umieszczenie parametrów w rejestrach jednostki centralnej
2. zapamiętanie parametrów w tablicy w pamięci operacyjnej i przekazanie adresu tej tablicy jako parametru w rejestrze
3. składowanie przez program parametrów na stosie i zdejmowanie ze stosu przez system operacyjny

48. Wymień rodzaje wywołań (funkcji) systemowych.

1. nadzorowanie procesów
2. operacje na plikach
3. operacje na urządzeniach
4. utrzymywanie informacji

5. komunikacja

49. Wymień struktury systemów operacyjnych oraz przykłady ich realizacji.

- Monolityczna (ang. monolithic) - jądro jednoczęściowe
  - OS/360 - 5000 programistów 1M kodu w ciągu 5 lat
  - IBM/360 MVT/TSO - koszt 50M $
  - AIX - Unix wersji IBM - jądro dwuczęściowe
- Warstwowa (ang. layered)
  - strukura hierarchiczna - skutki małych zmian w jednej warstwie trudne do przewidzenia w innych warstwach
- Maszyna wirtualna (ang. virtual machine)
- Egzojądro (ang. exokernel)
- Klient - serwer (ang. client-server)
  - mikrojądro (ang. microkernel, µ-kernel) - jedynie bezwzględnie niezbędne funkcje systemowe w jądrze systemu (np. mikrojądro L4 ma 12kB kodu, 7 funkcji systemowych)
- Modularność (Solaris, Linux)

50. Wyjaśnij zasadę maszyny wirtualnej.

1. maszyna wirtualna (ang. virtual machine) jest logiczną konsekwencją podejścia warstwowego: jądro systemu jest traktowane jako sprzęt
2. maszyna wirtualna dostarcza identycznego interfejsu dla sprzętu
3. system operacyjny tworzy wirtualne systemy komputerowe, każdy proces ma do dyspozycji własne(wirtualne) jądro, dyski, pamięć, drukarki
4. zasoby fizycznego komputera są dzielone w celu utworzenia maszyn wirtualnych
   a. planowanie przydziału procesora jest tak wykorzystane, że użytkownik ma wrażenie jakoby miał do dyspozycji własny procesor
   b. spooling i system zarządzania plikami jest wykorzystany tak, że powstaje wrażenie użytkowania drukarki,czytnika na wyłączność
   c. zwykłe terminale użytkownika funkcjonują jak konsole operatorskie maszyny wirtualnej

61. Podaj przykłady uruchamiania procesów w systemie Unix.

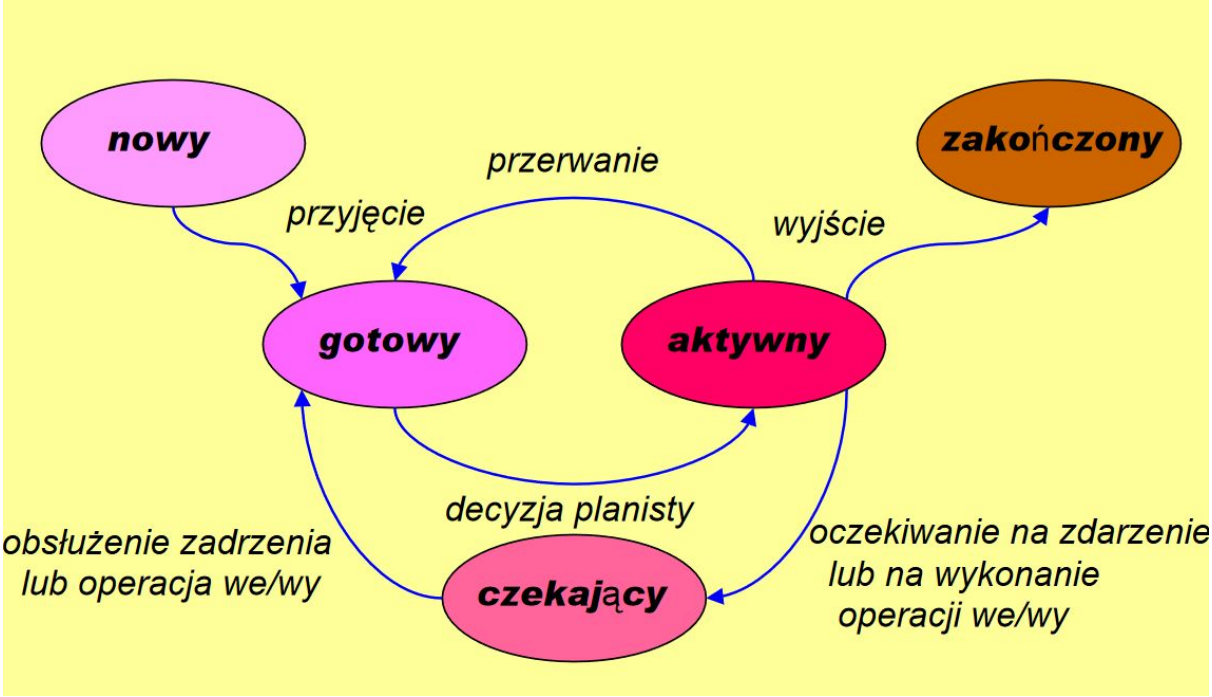
1. Proces to program działający we własnej przestrzeni adresowej
   a. proces wsadowy –polecenie batch, at, cron
   b. procesy interakcyjne
      i. procesy pierwszoplanowe – związane z terminalem (np. polecenie ls)

ii. procesy drugoplanowe –wykonywane w tle
   c. demony (ang. daemons) – wystartowane serwisy
      i. init, syslogd, sendmail, lpd, crond, getty, bdflush, pagedaemon, swapper, inetd, named,routed, dhcpd, portmap, nfsd, smbd, httpd, ntpd
2. Limity zasobów procesów: polecenia limit, ulimit

62. Podaj komendy do sterowania procesami w systemie Unix.

1. Uruchamianie w tle: &
2. Zatrzymanie procesu pierwszoplanowego: ^Z
3. Ponowne uruchamianie procesu w tle: bg
4. Listowanie procesów w tle: jobs
5. Odwołanie do procesu n w tle: %n
   a. Odwołanie do procesu xyz w tle: %?xyz
6. Przenoszenie z tła na pierwszy plan: fg %

63. Narysuj diagram stanów procesów.



64. Opisz zawartość PCB.

1. każdy proces w systemie operacyjnym jest reprezentowany przez blok kontrolny procesu zawierający
   a. stan procesu – gotowy, nowy, aktywny, czekający, zatrzymany
   b. licznik rozkazów - adres następnego rozkazu do wykonania w procesie

c. rejestry procesora - zależą od architektury komputera: akumulatory, rejestry (ogólne, bazowe,indeksowe) wskaźniki stosu przechowywane aby proces mógł być kontynuowany po przerwaniu
d. informacje o planowaniu przydziału procesora - priorytet procesu, wskaźniki do kolejek porządkujących zamówienia
e. informacje o zarządzaniu pamięcią - zawartości rejestrów granicznych, tablice stron, tablice segmentów w zależności od systemu używanej pamięci
f. informacje do rozliczeń - ilość zużytego czasu procesora i czasu rzeczywistego, ograniczenia czasowe, numery kont, numery zadań
g. informacje o stanie we/wy - lista zaalokowanych urządzeń, wykaz otwartych plików

65. Omów przyczyny przełączania procesu.

1. Przerwanie –zdarzenie zewnętrzne w stosunku do procesu
2. Pułapka –zdarzenie wewnątrz aktualnie przetwarzanego procesu, błąd lub warunek wyjątku
3. Polecenie administracyjne –wywołanie funkcji systemu operacyjnego

66. Omów rodzaje planistów i zadania przez nich realizowane.

1. planista długoterminowy lub planista zadań
    a. wybiera procesy, które powinny być sprowadzone do pamięci do kolejki procesów (niegotowych)
    b. jest wołany rzadko (sekundy, minuty) dlatego może nie być szybki
    c. nadzoruje stopień wieloprogramowości (liczbę procesów w pamięci)
    d. powinien dobrać mieszankę procesów zawierającą zarówno procesy ograniczone przez we/wy jak i procesor
2. planista krótkoterminowy lub planista przydziału procesora
    a. wybiera proces następny do wykonania z kolejki procesów gotowych i przydziela mu procesor
    b. jest wołany bardzo często (milisekundy) dlatego musi być bardzo szybki
3. planista średnioterminowy
    a. usuwa procesy z pamięci zmiejszajac stopień wieloprogramowości, a później sprowadza je ponownie w celu uzyskania lepszej mieszanki procesów

67. Omów przełączanie kontekstu.

1. gdy procesor przełącza do innego procesu system musi zachować stan starego procesu i załadować zachowany stan nowego procesu. Czynność tę nazywamy przełączaniem kontekstu
2. przełączanie kontekstu jest ceną za wieloprogramowość;
    a. system operacyjny nie wykonuje wtedy żadnej użytecznej pracy
3. czas przełączenia kontekstu zależy od sprzętu(zwykle od 1 do 1000 milisekund)

68. Omów tworzenie procesu zombie i orphan.

1. Zombie - W przypadku, gdy proces potomny kończy się w czasie, gdy jego proces rodzicielski nie wykonuje funkcji wait, wówczas kończący się proces jest umieszczany w stanie zawieszenia i staje się procesem zombie. Proces zombie zajmuje pozycję w tabeli utrzymywanej w jądrze dla kontroli procesów, ale nie używa Żadnych innych zasobów jądra. Zostanie on zakończony, gdy jego proces rodzicielski zażąda potomka za pomocą wykonania funkcji wait.
2. Orphan – to proces którego rodzic przestał istnieć (np. została wywołana funkcja exit). Przejmuje go proces init (Żeby nie mógł się stać zombie).

69. Omów komunikację pośrednią i bezpośrednią między procesami.

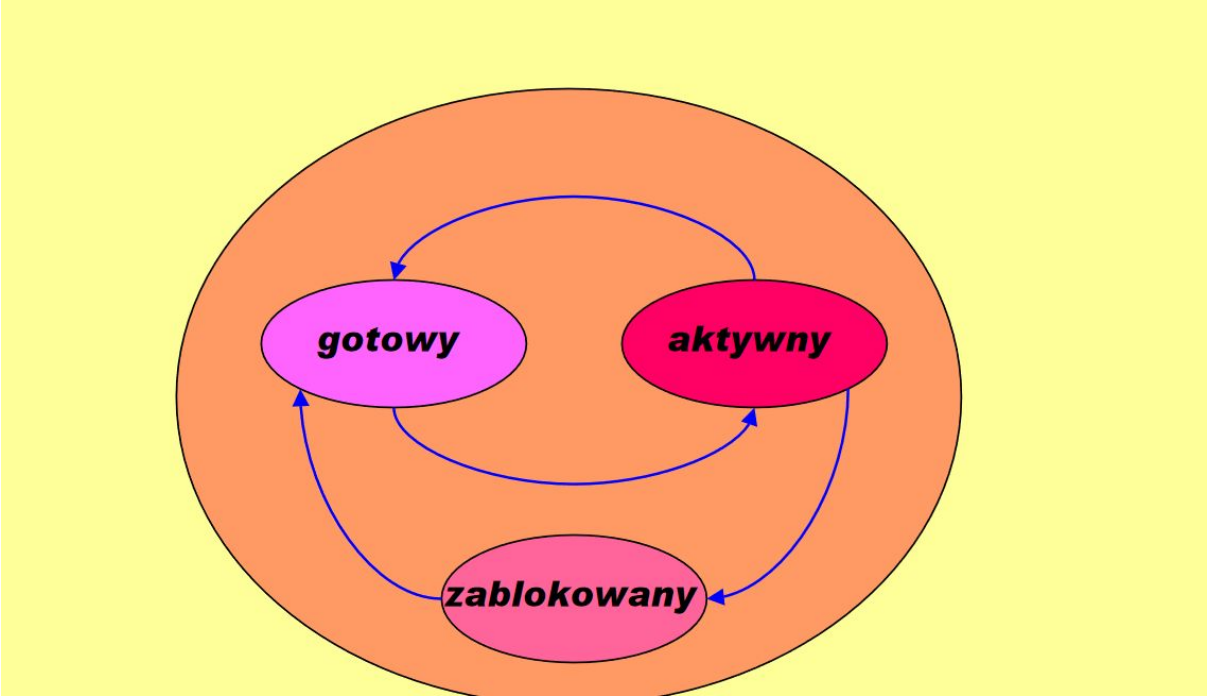
1. komunikacja bezpośrednia
    a. proces musi jawnie nazwać odbiorcę
        i. nadaj(P,komunikat) - nadaj komunikat do procesu P
        ii. odbierz(Q,komunikat) - odbierz komunikat od procesu Q
    b. własności łącza
        i. łącze jest ustanawiane automatycznie, do komunikowania wystarczy znajomość identyfikatorów
        ii. łącze dotyczy dokładnie dwóch procesów
        iii. między każdą parą procesów istnieje dokładnie jedno łącze
        iv. łącze jest zwykle dwukierunkowe, ale może być jednokierunkowe
2. komunikacja pośrednia
    a. komunikaty są nadawane i odbierane za pomocą skrzynek pocztowych nazywanych także portami
        i. każda skrzynka ma swój unikalny identyfikator
        ii. procesy komunikują się jeśli mają wspólną skrzynkę
    b. własności łącza
        i. łącze jest ustanawiane jedynie wtedy, gdy procesy dzielą skrzynkę
        ii. łącze może być związane z więcej niż dwoma procesami
        iii. każda para procesów może mieć kilka łączy, z których każdy odpowiada jakiejś skrzynce
        iv. łącze może być jedno- lub dwukierunkowe
    c. system operacyjny dostarcza mechanizmów do
        i. tworzenia nowej skrzynki
        ii. nadawania i odbierania komunikatów za pomocą skrzynki
        iii. likwidowania skrzynki

70. Wymień zasoby używane przez wątki.

1. wątek nazywany niekiedy procesem lekkim jest podstawową jednostką wykorzystania procesora. W skład tej jednostki wchodzą

   a. licznik rozkazów
   b. zbiór rejestrów
   c. obszar stosu
2. wątek współużytkuje z innymi równorzędnymi wątkami
   a. sekcję kodu
   b. sekcję danych
   c. zasoby systemu (takie jak otwarte pliki i sygnały) zwane wspólnie zadaniem

71. Narysuj diagram stanów wątku.



72. Wymień typy wątków.

1. wątki obsługiwane przez jądro
2. wątki tworzone na poziomie użytkownika za pomocą funkcji bibliotecznych
3. hybrydowe podejście
4. wątki zarządzane przez JVM
5. zielone – wątki w ramach maszyny wirtualnej
6. natywne – wątki w ramach systemu operacyjnego

73. Wymień sposoby odwzorowań wątków.

1. Wiele do jednego
2. Jeden do jednego
3. Wiele do wielu
4. Jeden do wielu

74. Wymień sytuacje w jakich planista przydziału procesora podejmuje decyzję o przydziale procesora.

1. gdy proces przeszedł od stanu aktywności do czekania (np. z powodu we/wy)
2. gdy proces przeszedł od stanu aktywności do gotowości (np. wskutek przerwania)
3. gdy proces przeszedł od stanu czekania do gotowości (np. po zakończeniu we/wy)
4. gdy proces kończy działanie

75. Wymień warunki planowania bez wywłaszczeń.

1. planowanie bez wywłaszczeń : proces, który otrzyma procesor, zachowuje go tak długo aż nie odda go z powodu przejścia w stan oczekiwania lub zakończenia (nie wymaga zegara)
2. planowanie w sytuacji 1 i 4 nazywamy nie wywłaszczeniowym – z poprzedniego pytania

76. Co to jest dispatcher oraz dispatch latency?

1. ekspedytor (ang. dispatcher) jest modułem, który faktycznie przekazuje procesor do dyspozycji procesu wybranego przez planistę krótkoterminowego; obowiązki ekspedytora to
    a. przełączanie kontekstu
    b. przełączanie do trybu użytkownika
    c. wykonanie skoku do odpowiedniej komórki w programie użytkownika w celu wznowienia działania programu
2. opóźnienie ekspedycji (ang. dispatch latency) to czas, który ekspedytor zużywa na wstrzymanie jednego procesu i uaktywnienie innego

77. Co to jest przepustowość oraz wykorzystanie CPU?

1. Wykorzystanie procesora –procent czasu, przez który procesor pozostaje zajęty
2. Przepustowość - liczba procesów kończących w jednostce czasu

78. Podaj definicje czasu (cyklu) przetwarzania, czasu oczekiwania, czasu odpowiedzi.

1. Czas cyklu przetwarzania - czas między nadejściem procesu do systemu a chwilą zakończenia procesu
    a. suma czasów czekania na wejście do pamięci, czekania w kolejce procesów gotowych, wykonywania procesu przez CPU i wykonywania operacji we/wy
2. Czas oczekiwania - suma okresów, w których proces czeka w kolejce procesów gotowych do działania

3. Czas odpowiedzi lub reakcji -ilość czasu pomiędzy wysłaniem Żądania a pojawieniem się odpowiedzi bez uwzględnienia czasu potrzebnego na wyprowadzenie odpowiedzi (np. na ekran).

79. Wymień kryteria optymalizacji algorytmu planowania.

1. maksymalne wykorzystanie procesora
2. maksymalna przepustowość
3. minimalny cykl przetwarzania
4. minimalny czas oczekiwania
5. minimalny czas odpowiedzi

80. Opisz algorytm FCFS.

1. pierwszy zgłoszony, pierwszy obsłużony (ang. first-come, first-served – FCFS)
2. implementuje się łatwo za pomocą kolejek FIFO - blok kontrolny procesu dołączany na koniec kolejki, procesor dostaje PCB z czoła kolejki
3. algorytm FCFS jest niewywłaszczający - proces utrzymuje procesor do czasu aż zwolni go wskutek zakończenia lub zamówi operację we/wy
4. algorytm FCFS jest kłopotliwy w systemach z podziałem czasu, bowiem w takich systemach ważne jest uzyskiwanie procesora w regularnych odstępach czasu
5. Proces zawsze dostanie się do CPU tj.. nie ma groźby zagłodzenia procesów
6. Niewydajne wykorzystanie CPU oraz we/wy
7. Krzywdzący dla procesów krótkich oraz ograniczonych przez we/wy bowiem faworyzuje dłuższe zadania

81. Opisz algorytm SJF wywłaszczający.

1. algorytm najpierw najkrótsze zadanie (ang. shortest-job-first - SJF) wiąże z każdym procesem długość jego najbliższej z przyszłych faz procesora. Gdy procesor staje się dostępny wówczas zostaje przydzielony procesowi o najkrótszej następnej fazie (gdy fazy są równe to mamy FCFS)
2. wywłaszczający – SJF usunie proces jeśli nowy proces w kolejce procesów gotowych ma krótszą następną fazę procesora od czasu do zakończenia procesu
3. SJF charakteryzuje to, że
    a. ma dobry czas odpowiedzi dla krótkich procesów
    b. jest krzywdzący dla procesów długich
    c. może powodować zagłodzenie procesów

82. Opisz algorytm HRRN.

1. Stosunkiem reaktywności nazywamy liczbę R = 1+ w/t, gdzie w oznacza czas oczekiwania na procesor zaś t -fazę procesora
2. Największy stosunek reaktywności jako następny
3. Podobnie jak SJF i SRTF również algorytm HRRN wymaga oszacowania dla następnej fazy procesora

4. Faworyzuje krótkie zadania jednak oczekiwanie dłuższych zadań zmienia ich współczynnik i w konsekwencji pozwala im uzyskać dostęp do CPU
5. Ma dobry czas odpowiedzi
6. Proces zawsze dostanie się do CPU (po pewnym czasie) tj. nie ma groźby zagłodzenia procesów

83. Podaj wzór na oszacowanie następnej fazy procesora.

1. t(n) = długość n-tej fazy procesora
2. a - liczba z przedziału [0,1], zwykle 0.5
3. s(n+1) = przewidywana długość następnej fazy
4. s(n) = dane z minionej historii
5. s(0) = stała (np. średnia wzięta z całego systemu)
6. s(n+1) = a*t(n) + (1-a)*s(n)

84. Opisz planowanie priorytetowe.

1. SJF jest przykładem planowania priorytetowego, w którym każdemu procesowi przypisuje się priorytet
2. Priorytety należą do pewnego skończonego podzbioru liczb naturalnych np. [0..7], [0,4095]
3. Procesor przydziela się procesowi o najwyższym priorytecie (jeśli priorytety są równe to FCFS)
    a. planowanie priorytetowe wywłaszczające
    b. planowanie priorytetowe nie wywłaszczające
4. SJF -priorytet jest odwrotnością następnej fazy

85. Opisz algorytm RR.

1. planowanie rotacyjne zaprojektowano dla systemów z podziałem czasu
2. każdy proces otrzymuje małą jednostkę czasu, nazywaną kwantem czasu. Gdy ten czas minie proces jest wywłaszczany i umieszczany na końcu kolejki zadań gotowych (FCFS z wywłaszczeniami)
3. średni czas oczekiwania jest stosunkowo długi
4. jeśli jest n procesów w kolejce procesów gotowych a kwant czasu wynosi q to każdy proces otrzymuje 1/n czasu procesora porcjami wielkości co najwyżej q jednostek czasu.
5. każdy proces czeka nie dłużej niż (n-1)*q jednostek czasu
6. wydajność algorytmu
    a. gdy q duże to RR przechodzi w FCFS
    b. gdy q małe to mamy dzielenie procesora ale wtedy q musi być duże w stosunku do przełączania kontekstu
7. mniejszy kwant czasu zwiększa przełączanie kontekstu
8. czas cyklu przetwarzania zależy od kwantu czasu
9. dobry czas odpowiedzi dla krótkich procesów
10. Efektywny w systemach z podziałem czasu

11. Sprawiedliwie traktowane procesów
12. Nie powoduje zagłodzenia procesów
13. Kwant powinien być nieco dłuższy od czasu wymaganego na typową interakcje
14. Procesy ograniczone przez CPU są faworyzowane kosztem procesów ograniczonych przez we/wy co prowadzi do nieefektywnego wykorzystania we/wy

105. Podaj przykład kodu programu z impasem i bez impasu.

```
typedef int semaphore;
 sempahore resource_1;
 sempahore resource_2;
void process_A(void) {
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
void process_B(void) {
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
```

kod bez impasu

```
typedef int semaphore;
 sempahore resource_1;
 sempahore resource_2;
void process_A(void) {
   down(&resource_1);
   down(&resource_2);
   use_both_resources();
   up(&resource_2);
   up(&resource_1);
}
void process_B(void) {
   down(&resource_2);
   down(&resource_1);
   use_both_resources();
   up(&resource_1);
   up(&resource_2);
}
```

kod z impasem

106. Podaj definicję impasu.

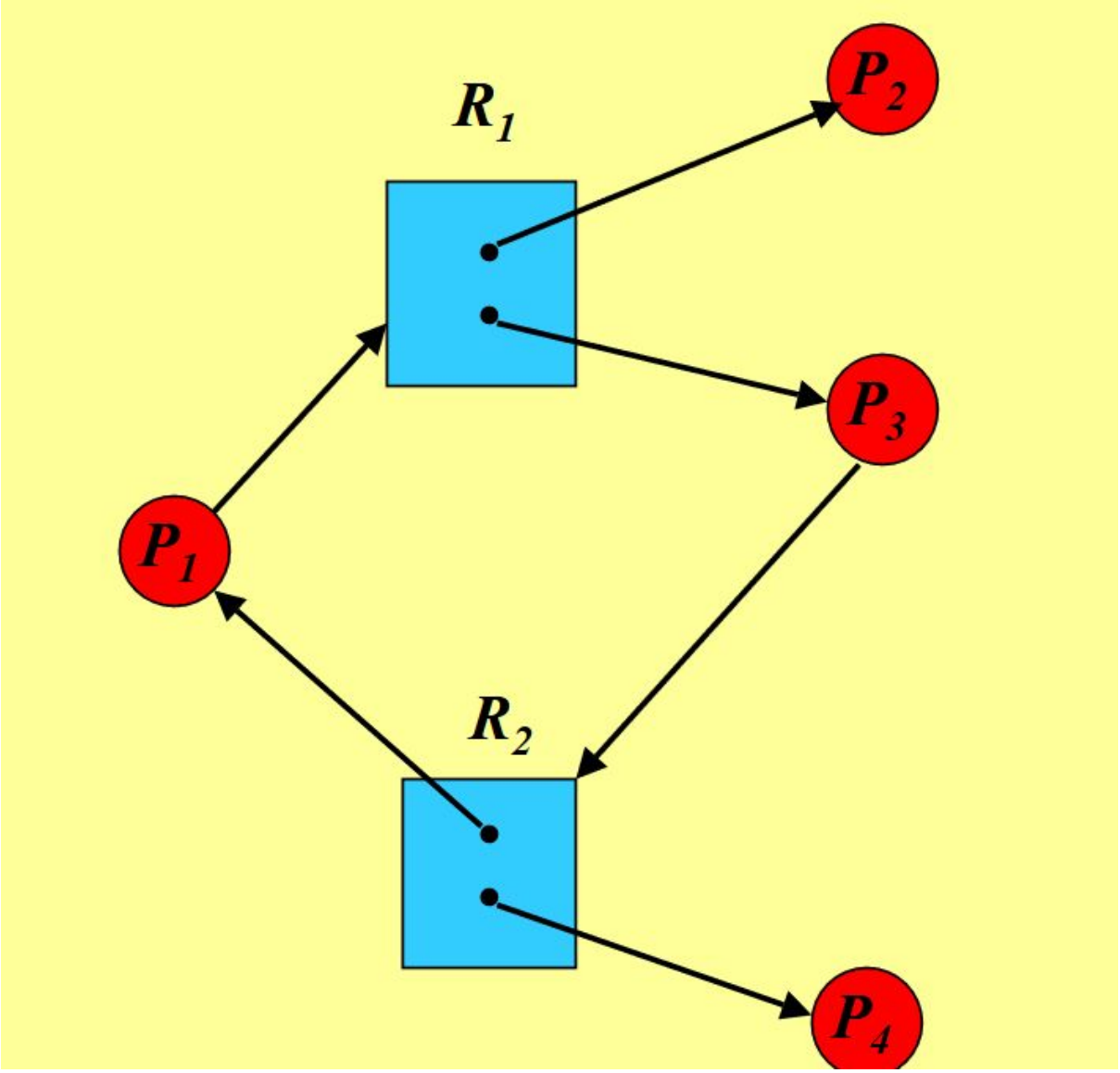
1. Zbiór procesów jest w stanie impasu, gdy każdy proces z tego zbioru czeka na zdarzenie, które może być spowodowane tylko przez inny proces z tego samego zbioru

107. Kiedy może wystąpić impas w systemie?

1. Do impasów może dochodzić wtedy, kiedy w systemie zachodzą jednocześnie cztery warunki
    a. wzajemne wykluczanie
    b. przetrzymywanie i oczekiwanie
    c. brak wywłaszczeń

d. czekanie cykliczne

108. Podaj przykład grafu przydziału zasobów z cyklem i bez impasu.



109. Opisz sposób zapobiegania impasom.

1. Zapewnić, aby nie mógł wystąpić przynajmniej jeden z czterech warunków koniecznych:
    a. Wzajemne wykluczanie - dotyczy jedynie zasobów niepodzielnych (np. drukarek); nie można zaprzeczyć temu warunkowi, bowiem niektóre zasoby są z natury niepodzielne

b. Przetrzymywanie i oczekiwanie - aby zapewnić, że warunek ten nigdy nie wystąpi w systemie trzeba zagwarantować, że gdy proces zamawia zasób to nie powinien trzymać innych zasobów
    i. każdy proces zamawia i otrzymuje wszystkie swoje zasoby przed rozpoczęciem działania
    ii. proces może zamówić zasób o ile nie ma Żadnych zasobów
        1. zanim proces zamówi zasób musi wpierw wszystkie oddać
c. Brak wywłaszczeń -trzeba zapewnić, że zasoby nie ulegają wywłaszczeniu jedynie z inicjatywy przetrzymującego je procesu
d. Czekanie cykliczne – sposobem, aby warunek ten nigdy nie wystąpił w systemie jest przyporządkowanie wszystkim zasobom danego typu kolejnych liczb i Żądanie, aby
    i. każdy proces zamawiał zasoby we wzrastającym porządku numeracji
    ii. alternatywnie można wymagać, aby proces zamawiający zasób miał zawsze zwolnione zasoby o numeracji wyższej od zamawianego

110. Opisz algorytm unikania impasu dla zasobów reprezentowanych jednokrotnie.

1. wymaga dodatkowej informacji o tym jak będzie następowało zamawianie zasobów
2. w najprostszym i najbardziej użytecznym modelu zakłada się, że system zadeklaruje maksymalną liczbę zasobów każdego typu, którą mógłby potrzebować
3. algorytm unikania impasu sprawdza dynamicznie stan przydziału zasobów aby zapewnić, że nigdy nie dojdzie do spełnienia warunku czekania cyklicznego
4. stan przydziału zasobów jest określony przez liczbę dostępnych i przydzielonych zasobów oraz przez maksymalne zapotrzebowanie procesów.

111. Podaj definicję stanu bezpiecznego.

1. kiedy proces Żąda wolnego zasobu system musi zadecydować czy przydzielenie zasobu pozostawi system w stanie bezpiecznym
2. system jest w stanie bezpiecznym, jeśli istnieje taki porządek przydzielania zasobów każdemu procesowi, który pozwala na uniknięcie impasu
3. system jest w stanie bezpiecznym, jeśli istnieje bezpieczny ciąg procesów
4. system jest w stanie zagrożenia jeśli nie jest w stanie bezpiecznym
5. ciąg procesów <P1, P2, …, Pn> jest bezpieczny jeśli dla każdego procesu Pi , jego potencjalne zapotrzebowanie na zasoby może być zaspokojone przez bieżąco dostępne zasoby + zasoby użytkowane przez wszystkie procesy Pj , przy czym j<i
    a. jeśli Pi Żąda zasobów, które nie są natychmiast dostępne, to Pi może poczekać aż skończą się wszystkie procesy Pj
    b. jeśli Pj skończą, Pi może otrzymać wszystkie potrzebne zasoby, dokończyć pracę, zwolnić zasoby i zakończyć
    c. jeśli Pi zakończy, Pi+1 może otrzymać niezbędne zasoby itd.

112. Opisz algorytm bankiera.

113. Opisz algorytm zamawiania zasobów.

$Request_i$ - wektor żądań P_i. Jeśli $Request_i[j] == k$ to P_i chce k egzemplarzy zasobu typu R_j

1. Jeśli $Request_i \leq Need_i$ goto 2. W przeciwnym razie błąd
2. Jeśli $Request_i \leq Available$, goto 3. W przeciwnym razie P_i czeka
3. System próbuje zaalokować zasoby P_i modyfikując stan

$$Available = Available - Request_i;$$
$$Allocation_i = Allocation_i + Request_i;$$
$$Need_i = Need_i - Request_i;$$

- *Jeśli stan safe ⇒ alokuj zasoby do P_i.*
- *Jeśli stan unsafe ⇒ P_i musi czekać na realizację i poprzedni stan przydziału zasobów jest odtwarzany*

115. Opisz algorytm wykrywania impasu dla zasobów reprezentowanych jednokrotnie.

1. Tworzymy graf oczekiwania
    a. węzły grafu są procesami
    b. Pi→Pj gdy Pi czeka na Pj.
2. Okresowo wykonujemy algorytm szukający cyklu w grafie oczekiwania
3. Rząd algorytmu wykrywania cykli w grafie oczekiwania wynosi n^2, przy czym n jest liczbą wierzchołków grafu

116. Opisz algorytm wykrywania impasu dla zasobów reprezentowanych wielokrotnie.

1. Metoda grafu oczekiwania jest bezużyteczna do wykrywania impasu, gdy każdy typ zasobu ma wiele egzemplarzy
2. Algorytm wykrywania impasu bada czy istnieje ciąg bezpieczny dla procesów, które trzeba dokończyć
    a. korzysta ze struktur danych algorytmu bankiera
3. Available: wektor długości m oznacza liczbę dostępnych zasobów każdego typu

4. Allocation: macierz n x m definiuje liczbę zasobów każdego typu aktualnie zaalokowanych do każdego z procesów
5. Request: macierz n x m określa bieżące zamówienie każdego procesu. Jeśli Request [i,j] = k, to proces Pi zamawia dodatkowo k egzemplarzy zasobu typu Rj

117. Opisz sposoby likwidowania impasu.

1. Zaniechanie wszystkich zakleszczonych procesów
2. Usuwanie procesów pojedynczo, aż do wyeliminowania cyklu impasu
3. Wycofanie - wycofanie procesu do bezpiecznego stanu, od którego można go będzie wznowić
4. Głodzenie -ten sam proces może być zawsze ofiarą, podobnie jak i ten sam proces może być ciągle wycofywany. Trzeba zadbać, aby proces mógł być delegowany do roli ofiary tylko skończoną liczbę razy
5. Podejście mieszane

137. Wymień zalety pamięci wirtualnej.

1. brak ograniczeń na pamięć
2. więcej programów – lepsze wykorzystanie procesora
3. można jedynie część programu załadować do pamięci w celu wykonania (reszta nieużywana (procedury obsługi rzadkich błędów) albo nadmiarowa (np. nadmiarowe tablice) jest w pamięci wirtualnej)
4. logiczna przestrzeń adresowa może być większa niż fizyczna
5. odseparowanie pamięci logicznej użytkownika od pamięci fizycznej,
6. kopiowanie przy zapisie przy tworzeniu procesu
7. odwzorowanie plików do pamięci przy tworzeniu procesu

138. Wymień sposoby implementacji pamięci wirtualnej.

1. stronicowanie na Żądanie
2. segmentacja na Żądanie

139. Opisz stronicowanie na żądanie.

1. System stronicowania na Żądanie jest podobny do stronicowania z wymianą
2. Procedura leniwej wymiany
    a. nigdy nie dokonuje się wymiany strony w pamięci jeśli nie jest to konieczne
    b. mniej operacji we/wy
    c. mniej pamięci
    d. szybsza reakcja
    e. więcej użytkownikowi
3. Jeśli strona jest potrzebna odwołaj się
    a. niepoprawne odwołanie -> abort
    b. brak strony w pamięci -> sprowadź stronę do pamięci
4. Zgodność z Zasadą Lokalności Odniesień

140. Opisz procedurę obsługi braku strony.

1. system operacyjny sprawdza wewnętrzną tablicę oraz decyduje że:
    a. jeśli odwołanie niedozwolone - kończy proces
    b. jeśli odwołanie dozwolone tylko zabrakło strony w pamięci to sprowadza tę stronę
2. system znajduje wolną ramkę na liście wolnych ramek
3. system wczytuje stronę z dysku do wolnej ramki
4. system wstawia bit 1 w tablicy stron
5. system wykonuje przerwany rozkaz

141. Podaj wzór na obliczenie sprawności stronicowania.

1. EAT – efektywny czas dostępu
2. p – prawdopodobieństwo braku strony (0 – brak braku stron , 1 - każde odwołanie generuje brak strony)
3. cd - czas dostępu do pamięci
4. cz - czas obsługi strony (obsługa przerwania wywołanego brakiem strony, czytanie strony, wznowienie procesu)
5. EAT = (1-p)*cd+p*cz

142. Opisz procedurę zastępowania stron.

1. Zlokalizowanie potrzebnej strony na dysku
2. Odnalezienie wolnej ramki
    a. jeśli ramka istnieje -zostaje użyta
    b. w przeciwnym razie typowanie ramki ofiary
    c. ramka ofiara zapisana na dysk; zmień tablicę stron i tablicę ramek
3. Wczytanie potrzebnej strony; zmień tablice stron i ramek
4. Wznowienie procesu

143. Opisz algorytm zastępowania stron FIFO.

1. algorytm FIFO stowarzysza z każdą ze stron czas kiedy została ona sprowadzona do pamięci
2. jeśli trzeba zastąpić stronę to zastępowana jest najstarsza ze stron
3. implementuje się za pomocą kolejek FIFO

144. Skonstruuj przykład ilustrujący anomalię Belady’ego.

1. anomalia Belady’ego odzwierciedla fakt, że w niektórych algorytmach zastępowania stron współczynnik braków stron może wzrastać ze wzrostem wolnych ramek
2. przykład dla algorytmu FIFO:
    a. ciąg odniesień: 1, 2, 3, 4, 1, 2, 5, 1, 2, 3, 4, 5
    b. 3 ramki (3 strony mogą być w pamięci w tym samym czasie) - 9 braków stron

c. 4 ramki (4 strony mogą być w pamięci w tym samym czasie) – 10 braków stron

145. Opisz algorytm zastępowania stron LRU.

1. zastąp tę stronę, która najdawniej była użyta
2. nie jest dotknięty anomalią Belady'ego
3. odwracalność

146. Opisz sposoby implementacji algorytmu LRU.

1. liczniki – do każdej pozycji w tablicy stron dołączamy rejestr czasu użycia, do procesora zaś dodajemy zegar logiczny lub licznik. Wskazania zegara są zwiększane wraz z każdym odniesieniem do pamięci. Ilekroć występuje odniesienie do pamięci, tylekroć zawartość rejestru zegara jest kopiowana do rejestru czasu użycia należącego do danej strony w tablicy stron
2. stos – przy każdym odwołaniu do strony jej numer wyjmuje się ze stosu i umieszcza na szczycie -najlepsza implementacja to dwukierunkowa lista ze wskaźnikami do czoła i do końca
    a. najwyżej 6 zmian wskaźników
    b. nie jest potrzebne przeszukiwanie listy

147. Opisz algorytm zastępowania stron OPT.

1. zastąp tę stronę, która najdłużej nie będzie używana; nazywany OPT lub MIN
2. nie ma anomalii Belady'ego
3. bardzo trudny do realizacji, bo wymaga wiedzy o przyszłej postaci ciągu odniesień (podobnie jak przy planowaniu procesora metodą SJF)
4. używany głównie w studiach porównawczych

148. Scharakteryzuj algorytmy stosowe.

1. Algorytm stosowy to taki algorytm dla którego zbiór stron w pamięci w przypadku n ramek jest podzbiorem zbioru stron w pamięci w przypadku n+1 ramek
2. Przykład: LRU
3. Własność: klasa algorytmów stosowych nie jest dotknięta anomalią Belady'ego
4. Implementacja LRU wymaga wsparcia sprzętowego: uaktualnianie pól zegara lub stosu musi być dokonywane przy każdym odniesieniu do pamięci
5. Możemy nie mieć takiego sprzętu

149. Opisz algorytm dodatkowych bitów odniesienia.

1. każda strona ma 8 bitowy rejestr w pamięci głównej

2. cyklicznie (co 100ms) przerwanie zegarowe powoduje wywołanie procedury która powoduje wprowadzenie bitu odniesienia na najbardziej znaczącą pozycję rejestru oraz przesunięcie pozostałych w prawo o 1 bit
    a. 00000000 –strona nieużywana przez osiem cykli
    b. 11111111 -strona używana co najmniej jeden raz w każdym cyklu
3. interpretujemy rejestry jako liczby bez znaku, tzn. strona najdawniej użyta ma najmniejszą zawartość rejestru

150. Opisz zegarowy algorytm drugiej szansy (CLOCK).

1. jeśli strona jest po raz pierwszy ładowana do pamięci odpowiadający jej bit odniesienia w ramce jest ustawiany na 1 a wskaźnik przesowamy cyklicznie na następna ramkę
2. jeśli nastąpiło odwołanie do strony to bit odniesienia = 1
3. jeśli brak jest strony to:
    a. następuje cykliczne przeszukiwanie ramek kandydatek do zastąpienia podczas którego bit odniesienia = 1 jest zerowany
    b. zostaje zastąpiona pierwsza strona z bitem odniesienia = 0
    c. wskaźnik zostaje ustawiony na następną ramkę
4. Implementacja przy pomocy kolejki cyklicznej

152. Opisz ulepszony algorytm drugiej szansy.

1. (x,y) -x -bit odniesienia, y-bit modyfikacji przypisany jest (w CPU) do każdej ramki
2. 4 klasy (od najniższej)
    a. (0,0) -nie używana ostatnio i nie zmieniana : najlepsza ofiara
    b. (0,1) -nie używana ostatnio ale zmieniona: gorsza ofiara bo wymaga zapisu na dysk choć zgodnie z zasadą lokalności pewnie nie zostanie użyta
    c. (1,0) -używana ostatnio i czysta: jeszcze gorsza ofiara bo zgodnie z zasadą lokalności może być wkrótce użyta
    d. (1,1) -używana ostatnio i zmieniana -najgorsza ofiara, prawdopodobnie (zgodnie z zasadą lokalności) będzie zaraz użyta a ponadto wymaga zapisu na dysku
3. 1. Skanowanie bufora ramek od pozycji wskaźnika
    a. bity nie są zmieniane; do wymiany pierwsza ramka (0,0)
4. 2. Jeśli w kroku 1 nie znaleziono ofiary to następuje skanowanie bufora w celu znalezienia ramki (0,1)
    a. do wymiany pierwsza znaleziona; bity odniesienia są w trakcie zerowane
5. 3. Jeśli w kroku 2 nie znaleziono ofiary to wskaźnik znajduje się w położeniu początkowym i wszystkie bity odniesienia są wyzerowane
    a. powtarzamy krok 1 a w przypadku nie znalezienia ofiary krok 2
    b. znajdujemy ramkę ofiarę; wymieniamy stronę; ustawiamy wskaźnik na następną ramkę w buforze
6. Ulepszony algorytm drugiej szansy w pierwszej kolejności zastępuje stronę, która nie była zmodyfikowana

153. Opisz schematy przydziału ramek.

1. Przydział równy - np. jeśli mamy 100 ramek i 5 procesów, to każdy proces może dostać 20 ramek
2. Przydział proporcjonalny - każdemu procesowi przydziela się dostępną pamięć proporcjonalnie do jego rozmiaru