ශිෂ්‍යයාගේ නම: .....

MEMORY MANAGEMENT / 01

PAGE TABLE

01. Page Table එකක් යනු කුමක්ද?

ක්‍රියාවලියක Page එකක් ප්‍රධාන මතකයේ (RAM) කුමන Frame එකේ තැන්පත් වී ඇත්ද යන්න දැක්වීමට භාවිත කරන දත්ත ව්‍යුහය **Page Table** ලෙස හැඳින්වේ.

සාමාන්‍යයෙන් එක් ක්‍රියාවලියක් සඳහා වෙනම Page Table එකක් පවතී. එහි එක් සටහනක් **Page Table Entry (PTE)** ලෙස හැඳින්වේ.

Q01. Page Table එක මඟින් ..... අංකයක්, ඊට අදාළ ..... අංකයට අනුරූපණය කරයි.

02. වගුව කියවා පිළිතුරු සපයමු

Page Number	Frame Number	Present Bit
0	5	1
1	2	1
2	—	0
3	7	1

Present Bit = 1: Page එක RAM තුළ පවතී.
Present Bit = 0: Page එක දැනට RAM තුළ නොපවතී.

Q02. Page 0 පවතින්නේ Frame ..... තුළය.

Q03. Frame 7 තුළ පවතින්නේ Page ..... ය.

Q04. RAM තුළ නොමැති Page එක ..... ය.

Q05. එම Page එකට ප්‍රවේශ වන විට ..... ඇති වේ.

ලිපින පරිවර්තනය

Logical Address: [Page Number | Offset]
↓ Page Table ↓
Physical Address: [Frame Number | Offset]

Q06. මෙහි වෙනස් නොවන කොටස: .....

INSIDE AN ENTRY

03. Page Table Entry එකක ක්ෂේත්‍ර

පද්ධතිය අනුව PTE එකක ක්ෂේත්‍ර සහ ඒවාට වෙන් කරන බිටු ගණන වෙනස් විය හැකිය.

ක්ෂේත්‍රය	කාර්යය
Frame Number	Page එක තැන්පත් වී ඇති Frame එක හඳුනාගනී.
Present Bit	Page එක දැනට RAM තුළ තිබේද යන්න දක්වයි.
Valid / Invalid Bit	Page එක ක්‍රියාවලියේ වලංගු ලිපින අවකාශයට අයත්ද යන්න දක්වයි.
Modified / Dirty Bit	RAM වෙත ගෙනා පසු Page එකේ දත්ත වෙනස් වී තිබේද යන්න දක්වයි.
Reference / Accessed Bit	Page එකට ප්‍රවේශ වී තිබේද යන්න සටහන් කරයි.
Protection Bits	Read, Write, Execute වැනි අවසර දක්වයි.

සටහන: සමහර පද්ධති/පාඩම්වල Valid Bit යන නම RAM තුළ Page එක තිබීම දැක්වීමටත් භාවිත කරයි. ප්‍රශ්නයේ දී ඇති අර්ථය අනුව පිළිතුරු සපයන්න.

Q07. දත්ත වෙනස් වී තිබේද? .....
ලිවීමට අවසර තිබේද? .....
Page එක RAM තුළ තිබේද? .....

04. Page Table Size ගණනය කරමු

සරල Single-level Page Table එකක එක් Page එකකට එක් Entry එකක් ඇති විට:

Number of Entries = Logical Address Space Size ÷ Page Size

Page Table Size = Number of Entries × Size of One Entry

බෙදීමෙන් ශේෂයක් ලැබේ නම් Pages ගණන ඊළඟ පූර්ණ සංඛ්‍යාවට ඉහළට වටයන්න. ඒකක එකම ආකාරයට යොදාගන්න. 1 KiB = 1,024 bytes.

විසඳූ උදාහරණය

Logical address space = 16 KiB; Page size = 1 KiB;
Entry size = 4 bytes.

Entries ගණන = 16 ÷ 1 = **16**
Page Table Size = 16 × 4 = **64 bytes**

Q08. Page එකකට වෙන්වූ එක් Page Table සටහනක් හඳුන්වන්නේ ..... යනුවෙනි.

LET'S PRACTISE

ගණනයට මතක තබාගන්න

Frames ගණන = RAM ප්‍රමාණය ÷ Frame size
Frame number සඳහා බිටු n නම්, හඳුනාගත හැකි Frames = 2^n
අවම PTE බිටු = Frame number බිටු + අමතර ක්ෂේත්‍රවල බිටු

Entry size එක ලබාදී ඇත්නම් එය භාවිත කරන්න. අවම බිටු ගණන සහ සැබවින් වෙන් කරන bytes ගණන වෙනස් විය හැකිය.

Logical address space = **8 KiB**, Page size = **1 KiB**,
Entry size = **2 bytes**.

Q09. Entries ගණන = ... ÷ ..... = ...

Q10. Page Table Size = ... × ..... = ...

ක්‍රියාවලියක logical address space එක **32 KiB** වේ. Page size එක **2 KiB** වන අතර RAM එකේ Frames **256ක්** පවතී.

එක් PTE එකක **Frame Number**, **Present Bit එකක්**, **Dirty Bit එකක්** සහ **Reference Bit එකක්** පමණක් අඩංගු වේ.

Q11. Page Table එකේ Entries ගණන ගණනය කරන්න.

Q12. Frame Number එක සඳහා අවශ්‍ය බිටු ගණන සොයන්න.

Q13. එක් PTE එකකට අවශ්‍ය අවම බිටු ගණන සොයන්න.

Q14. එක් Entry එකකට 2 bytes වෙන් කරයි නම් මුළු Page Table Size එක සොයන්න.

Q15. Present Bit සහ Dirty Bit අතර වෙනස ලියන්න.

MEMORY MANAGEMENT

07. මතක කළමනාකරණය

ක්‍රියාවලි සඳහා **RAM වෙන් කිරීම, භාවිතය පාලනය කිරීම සහ නැවත නිදහස් කිරීම** මෙහෙයුම් පද්ධතිය මඟින් සිදු කිරීමයි.

මෙහෙයුම් පද්ධතියේ කාර්යය	කෙටි අදහස
වාර්තා තබාගැනීම	භාවිත වන සහ නිදහස් මතක ප්‍රදේශ හඳුනාගනී.
මතකය වෙන් කිරීම	ක්‍රියාවලි සඳහා අවශ්‍ය මතකය ලබාදෙයි.
හුවමාරුව පාලනය	RAM සහ ද්විතීයික ආචයනය අතර කොටස් හුවමාරු කරයි.
මතකය නිදහස් කිරීම	අවශ්‍ය නොවන වෙන් කළ මතකය නැවත ලබාගනී.

08. Memory Management Unit (MMU)

Logical Address එකක් Physical Address එකකට පරිවර්තනය කරන දෘඩාංග ඒකකයයි.

Logical Address	Physical Address
CPU මඟින් ජනනය කරන, වැඩසටහන භාවිත කරන ලිපිනය.	RAM තුළ දත්ත පවතින සැබෑ ලිපිනය.

සරල Base Register ක්‍රමය

Physical Address = Base Register Value + Logical Address

Base = 1,000; Logical Address = 100 නම්:
Physical Address = 1,000 + 100 = **1,100**

Q16. Base = 2,000 සහ Logical Address = 250 නම් Physical Address එක ගණනය කරන්න.

Paging ක්‍රමයේදී ලිපින පරිවර්තනයට Page Table එක භාවිත කරයි.

09. Paging — පිටුකරණය

තාර්කික මතකය සමාන ප්‍රමාණයේ **Pages** ලෙසත්, RAM එක එම ප්‍රමාණයේම **Frames** ලෙසත් බෙදයි.

Page size = Frame size
Pages, RAM තුළ එක ළඟ Framesවල තිබීම අවශ්‍ය නොවේ.
Page එකේ Frame එක හඳුනාගන්නේ **Page Table** මඟිනි.

Q17. Page size = 4 KiB නම්, Frame size = ...

Q18. මතක කළමනාකරණය සිදු කරන්නේ ..... මඟිනි.

VIRTUAL MEMORY

10. අතථ්‍ය මතකය යනු කුමක්ද?

වැඩසටහනක **අවශ්‍ය කොටස් RAM තුළත්, අනෙක් කොටස් ද්විතීයික ආචයනයේත්** තබාගෙන ක්‍රියාත්මක කිරීමට ඉඩ සලසන මතක කළමනාකරණ ක්‍රමයකි.

> **මූලික අදහස**
> වැඩසටහනේ සියලුම කොටස් එකවර RAM තුළ තිබීම අවශ්‍ය නොවේ.

11. Paging සමඟ ක්‍රියා කරන ආකාරය

පියවර	සිදුවන දේ
01	දැනට අවශ්‍ය Pages, RAM තුළ තබයි.
02	අනෙක් Pages, SSD/HDD වැනි ද්විතීයික ආචයනයේ තබයි.
03	අවශ්‍ය Page එක RAM තුළ නොමැති නම් **Page Fault** ඇති වේ.
04	OS එක එම Page එක RAM වෙත ගෙන එයි. ඉඩ නොමැති නම් වෙනත් Page එකක් ඉවත් කරයි.

Page එකට අදාළ Frame එක සහ එහි තත්ත්වය පිළිබඳ තොරතුරු **Page Table** එකේ සටහන් කරයි.

12. ප්‍රධාන අරමුණු

- RAM ධාරිතාවට වඩා විශාල ලිපින අවකාශයක් අවශ්‍ය වැඩසටහන් ක්‍රියාත්මක කිරීමට ඉඩ සැලසීම.
- අවශ්‍ය කොටස් පමණක් RAM වෙත ගෙන ඒම.
- ක්‍රියාවලි කිහිපයක් සඳහා RAM කාර්යක්ෂමව භාවිත කිරීම.

> SSD/HDD, RAM වලට වඩා මන්දගාමීය. එබැවින් අධික Page හුවමාරුව නිසා පද්ධතිය මන්දගාමී විය හැකිය.

Q19. MMU එකේ කාර්යය ලියන්න.

Q20. අවශ්‍ය Page එක RAM තුළ නොමැති විට ඇති වන්නේ කුමක්ද?

Q21. Virtual Memory භාවිතයේ වාසියක් ලියන්න.

FILES & DIRECTORIES

14. File Management — ගොනු කළමනාකරණය

ගොනු සහ නාමාවලි **නිර්මාණය, නම් කිරීම, සංවිධානය, ගබඩා කිරීම, ප්‍රවේශය පාලනය සහ මකා දැමීම** මෙහෙයුම් පද්ධතිය මඟින් කළමනාකරණය කිරීමයි.

ගොනුවක් (File) යනු නමක් යටතේ ගබඩා කළ, එකිනෙකට සම්බන්ධ දත්ත එකතුවකි.

Logical View — තාර්කික දැක්ම	Physical View — භෞතික දැක්ම
පරිශීලකයාට/වැඩසටහනට ගොනුව පෙනෙන ආකාරය. උදා: පාඨය, වාර්තා, රූපයේ pixels.	ආචයන උපාංගයේ blocks/clusters මත ගොනුව ගබඩා කර ඇති ආකාරය.

15. File Attributes — ගොනු ලාක්ෂණික

ලාක්ෂණිකය	අදහස / උදාහරණය
Name / Type	ගොනු නාමය සහ වර්ගය: lesson.txt
Location / Size	ගබඩා කළ ස්ථානය සහ ගොනුවේ ප්‍රමාණය
Owner / Permissions	අයිතිකරු සහ Read / Write / Execute අවසර
Date & Time	නිර්මාණය, වෙනස් කිරීම, අවසන් ප්‍රවේශය පිළිබඳ කාල සටහන්
Organization / Access	දත්ත සංවිධානය සහ ප්‍රවේශ ක්‍රමය: Sequential / Direct

16. File Types හා File Structure

File name: lesson.docx → නම: **lesson** | දිගුව: **.docx**
දිගුව ගොනු වර්ගය හඳුනාගැනීමට උපකාරී වේ; දිගුව වෙනස් කිරීමෙන් ගොනුවේ සැබෑ ආකෘතිය වෙනස් නොවේ.

වර්ගය	දිගු උදාහරණ	වර්ගය	දිගු උදාහරණ
පාඨ / ලේඛන	.txt, .docx	රූප	.jpg, .png, .bmp
ශබ්ද	.mp3, .wav	වීඩියෝ	.mp4, .avi
ක්‍රියාත්මක	.exe	සම්පීඩිත	.zip, .rar

File Structure: ගොනුව තුළ දත්ත සංවිධානය වී ඇති ආකාරයයි. පාඨ ගොනුවක අක්ෂර/පේළි; object file එකක යන්ත්‍ර කේත හා අදාළ තොරතුරු වැනි ව්‍යුහ තිබිය හැකිය.

17. Directory — නාමාවලිය

ගොනු සහ උපනාමාවලි පිළිබඳ නාම හා යොමු සංවිධානය කරන ව්‍යුහයකි.
Path උදාහරණය: /ICT/OS/lesson.pdf

Q22. lesson.pdf හි දිගුව: ..... | ගොනු වර්ගය: .....
Read අවසරයෙන් කළ හැක්කේ: .....

Q23. ගොනු හා උපනාමාවලි සංවිධානය කරන ව්‍යුහය ..... ලෙස හැඳින්වේ.

FILE SYSTEMS & SECURITY

18. File System — ගොනු පද්ධතිය

ආචයන උපාංගයක ගොනු **නම් කිරීම, සංවිධානය, ගබඩා කිරීම සහ නැවත සොයාගැනීම** සඳහා භාවිත වන ව්‍යුහ සහ නීති එකතුවයි.

කරුණ	FAT / FAT32	NTFS
සංවිධානය	File Allocation Table මඟින් ගොනුවේ cluster දාමය සටහන් කරයි.	Master File Table (MFT) මඟින් ගොනු metadata පවත්වාගනී.
ලක්ෂණ	සරලය; උපාංග බොහොමයක් සමඟ අනුකූලය.	විශාල ගොනු/පරිමා සඳහා සහාය; වැඩි පහසුකම්.
ආරක්ෂාව	NTFS වැනි ගොනු මට්ටමේ permissions නොමැත.	ගොනු/ෆෝල්ඩර අවසර; encryption සඳහා සහාය.
විශ්වසනීයත්වය	සාමාන්‍යයෙන් FAT පිටපත් දෙකක් පවත්වාගනී.	Journaling මඟින් බිඳවැටීමකට පසු ව්‍යුහමය ප්‍රතිසාධනයට උපකාර කරයි.

FAT පවුල MS-DOS සමඟ ප්‍රචලිත විය. NTFS යනු Microsoft ගොනු පද්ධතියකි; එය FAT වගුවේ වැඩිදියුණුවක් පමණක් නොවන වෙනම සැලසුමකි.

19. File Security — ගොනු ආරක්ෂාව

ක්‍රමය	අරමුණ
Authentication	“ඔබ කවුද?” යන්න තහවුරු කිරීම: මුරපදය, ඇඟිලි සලකුණ, අක්ෂි රටා.
Authorization / Permissions	“ඔබට කළ හැක්කේ මොනවාද?” යන්න පාලනය: Read, Write, Execute.
Encryption	නිසි යතුරක් නොමැති අයට කියවිය නොහැකි ලෙස දත්ත සංකේතනය කිරීම.
Backup	දත්ත අහිමි වුවහොත් නැවත ලබාගැනීමට පිටපතක් තබාගැනීම.

Q24. ඇඟිලි සලකුණෙන් පිවිසීම: .....
ගොනුව කියවීමට පමණක් ඉඩ දීම: .....

20. Maintenance of Secondary Storage

ද්විතීයික ආචයනයේ source programs, executable programs, වැඩසටහන් දත්ත සහ තාවකාලික දත්ත ගබඩා වේ.

නඩත්තු කාර්යයන්:
- නිදහස් ඉඩ හා block වෙන් කිරීම පිළිබඳ වාර්තා පවත්වාගැනීම.
- අනවශ්‍ය/තාවකාලික ගොනු ඉවත් කිරීම.
- දෝෂ පරීක්ෂාව සහ backups පවත්වාගැනීම.
- අවශ්‍ය විට HDD ගොනු විඛණ්ඩනය අඩු කිරීම.

Journaling යනු backup එකකට ආදේශයක් නොවේ.

Q25. දත්ත අහිමි වූ විට නැවත ලබාගැනීමට තබන පිටපත ..... වේ.

STORAGE ALLOCATION

21. ප්‍රධාන විභජන ක්‍රම තුන

ගොනුවේ දත්ත ගබඩා කිරීමට disk blocks වෙන් කරන ආකාරයයි. පහත සියලු උදාහරණවල අංක මඟින් **block අංක** දැක්වේ.

A. Contiguous Allocation — යාබද විභජනය

ගොනුව සඳහා **එක ළඟ blocks** වෙන් කරයි. ආරම්භක block එක සහ දිග සටහන් කරයි.

File A: [4] [5] [6] [7]
Start = 4 Length = 4 blocks

වාසි: සරලය; Sequential සහ Direct ප්‍රවේශය වේගවත්ය.
සීමා: ගොනුව වර්ධනය කිරීම අපහසු විය හැකිය; බාහිර ඛණ්ඩනීකරණය ඇති විය හැකිය. ප්‍රමාණය කලින් දැනගැනීම/ඉඩ වෙන් කිරීම වැදගත්ය.

B. Linked Allocation — සබැඳි විභජනය

ගොනුවේ blocks විවිධ ස්ථානවල තිබිය හැකිය. එක් block එක ඊළඟ block එකට යොමු කරයි.

Start → [1] → [12] → [20] → END

වාසි: ගොනුව වර්ධනය කිරීම පහසුය; බාහිර ඛණ්ඩනීකරණය නැත.
සීමා: Direct ප්‍රවේශය මන්දගාමීය; pointers සඳහා ඉඩ අවශ්‍යය.
FAT: ඊළඟ cluster යොමු දත්ත block තුළ වෙනුවට වගුවක තබන සබැඳි ක්‍රමයකි.

C. Indexed Allocation — සූචක විභජනය

ගොනුවේ data blocksවල අංක **වෙනම index block එකක** සටහන් කරයි. Directory entry එක index block එකට යොමු කරයි.

Directory → Index Block 17
Index entries: [13 | 7 | 21 | 28] → Data blocks

වාසි: Direct සහ Sequential ප්‍රවේශය කළ හැකිය; බාහිර ඛණ්ඩනීකරණය නැත.
සීමා: index සඳහා අමතර ඉඩ අවශ්‍යය; කුඩා ගොනුවලට එය සාපේක්ෂව වැඩිය.
උදාහරණ සංකල්පය: UNIX inode මඟින් block යොමු පවත්වාගනී.

Q26. ආරම්භක block = 10, දිග = 3 නම් blocks: ....., ..., ...

Q27. Blocks 2 → 9 → 5 ලෙස සම්බන්ධ නම් ක්‍රමය: .....

Q28. Index එක [6, 14, 3] නම් දෙවන data block එක: .....

Linked / Indexed ක්‍රමවල බාහිර ඛණ්ඩනීකරණය නොතිබුණත්, අවසාන allocation unit එක තුළ භාවිත නොවන ඉඩ තිබිය හැකිය.

FILE MANAGEMENT / 09

INTERNAL FRAGMENTATION

23. අභ්‍යන්තර ඛණ්ඩනීකරණය

ගොනුවකට වෙන් කළ block/cluster එකක් **ඇතුළත භාවිත නොවී ඉතිරි වන ඉඩ** අභ්‍යන්තර ඛණ්ඩනීකරණයයි. වෙනත් ගොනුවකට එම ඉඩ වෙනම වෙන් කළ නොහැකිය.

උදාහරණය: Block size = 4 KiB; File size = 10 KiB.

[4 KiB used] [4 KiB used] [2 used | 2 free]

අවශ්‍ය blocks = 3; වෙන් කළ ඉඩ = 12 KiB;
අභ්‍යන්තරව අපතේ ගිය ඉඩ = 12 − 10 = **2 KiB**.

Blocks ගණන = File size ÷ Block size → ඉහළ පූර්ණ සංඛ්‍යාවට වටයන්න.
වෙන් කළ ඉඩ = Blocks ගණන × Block size
අපතේ ගිය ඉඩ = වෙන් කළ ඉඩ − File size

පහත අභ්‍යාසවල metadata, pointers සහ index blocks සඳහා ඉඩ නොසලකා හරින්න. 1 KiB = 1,024 bytes.

Block size එක **4 KiB** වන පද්ධතියක **13 KiB** ගොනුවක් ගබඩා කරයි.

Q29. අවශ්‍ය blocks ගණන සොයන්න.

Q30. වෙන් කළ මුළු ඉඩ සහ අභ්‍යන්තරව අපතේ ගිය ඉඩ ගණනය කරන්න.

Cluster size එක **2 KiB** වේ. වෙන වෙනම ගබඩා කරන ගොනු තුනක ප්‍රමාණ **3 KiB, 6 KiB සහ 7 KiB** වේ.

Q31. වගුව සම්පූර්ණ කරන්න.

File size	Clusters ගණන	වෙන් කළ ඉඩ	අපතේ ගිය ඉඩ
(1) 3 KiB	.....	.....	.....
(2) 6 KiB	.....	.....	.....
(3) 7 KiB	.....	.....	.....

Q32. මුළු අභ්‍යන්තර අපතේ යාම = ...

වෙන් කළ කොටස **ඇතුළත** හිස් ඉඩ → Internal.

EXTERNAL FRAGMENTATION

24. බාහිර ඛණ්ඩනීකරණය

නිදහස් ඉඩ **වෙන් කළ කොටස් අතර කුඩා වෙන්වූ කොටස් ලෙස** පැවතීමයි. මුළු නිදහස් ඉඩ ප්‍රමාණවත් වුවත් අවශ්‍ය තරම් **එක දිගට ඉඩක්** නොමැති විය හැකිය.

පහත අභ්‍යාසවල Contiguous Allocation භාවිත කරයි. නිදහස් කොටස් අතර භාවිත වන blocks ඇත; ගොනු ගෙනයාම නොකරයි.

නිදහස් කොටස්: **3 blocks, 2 blocks, 4 blocks**. නව ගොනුවට **6 blocks** අවශ්‍යය.

Q33. මුළු නිදහස් blocks: ..... | විශාලතම නිදහස් කොටස: .....

Q34. ගොනුව වෙන් කළ හැකිද? හේතුව ලියන්න.

.....

.....

නිදහස් කොටස්: **8 KiB, 12 KiB, 4 KiB**. නව ගොනුවට **16 KiB** අවශ්‍යය.

Q35. මුළු නිදහස් ඉඩ: ..... | විශාලතම නිදහස් කොටස: .....

Q36. ගොනුව වෙන් කළ හැකිද? නිදහස් ඉඩ එකට එකතු කළහොත් තත්ත්වය කුමක්ද?

.....

.....

25. Disk Fragmentation / Defragmentation

Disk Fragmentation: එකම ගොනුවේ කොටස් තැටියේ වෙන්වූ ස්ථානවල ගබඩා වීමයි. HDD එකක ඒවා කියවීමට වැඩි head movement අවශ්‍ය විය හැකිය.

Before: [A1][B1][A2][free][B2][A3]
After: [A1][A2][A3][B1][B2][free]

Defragmentation: විසිරුණු ගොනු කොටස් නැවත යාබදව සකස් කිරීමයි. නිදහස් ඉඩ එකතු කිරීමටත් උපකාර විය හැකිය.

ගොනුවක කොටස් විසිරීම සහ නිදහස් ඉඩ විසිරීම එකම අර්ථය නොවේ. සාම්ප්‍රදායික defragmentation ප්‍රධාන වශයෙන් HDD සඳහාය; SSD සඳහා සාමාන්‍ය නඩත්තුව TRIM වේ.

26. Formatted Disk එකක දත්ත නැවත ලබාගැනීම

Quick format කිරීමේදී ගොනු පද්ධති යොමු/ව්‍යුහ නැවත සකස් කළත් දත්ත සියල්ල වහාම මකා නොදැමිය හැකිය. **Overwrite වීමට පෙර** recovery software මඟින් සමහර දත්ත ලබාගත හැකිය.

ප්‍රතිසාධනය සහතික නොවේ. Overwrite, full format, encryption සහ SSD TRIM නිසා දත්ත නැවත ලබාගත නොහැකි විය හැකිය. අවශ්‍ය නම් උපාංගයට නව දත්ත ලිවීම නවත්වන්න.

PROCESSES & PCB

27. Process — ක්‍රියාවලිය

ක්‍රියාත්මක වෙමින් පවතින වැඩසටහනක් ක්‍රියාවලියකි. Program එක උපදෙස් අඩංගු නිෂ්ක්‍රිය ගොනුවකි; Process එක ක්‍රියාකාරී අවස්ථාවකි. එකම program එකෙන් processes කිහිපයක් ඇති විය හැකිය.

CPU-bound	I/O-bound
වැඩි කාලයක් CPU ගණනය කිරීම් සඳහා යොදයි. උදා: විශාල ගණනයක්.	වැඩි කාලයක් I/O සඳහා රැඳී සිටී. උදා: නිතර disk/ network දත්ත කියවීම.

28. අවශ්‍යතා සහ ජීවන චක්‍රය

ක්‍රියාවලියකට **PID, executable code, දත්ත, මතකය සහ execution context** අවශ්‍ය වේ.

නිර්මාණයට හේතු	අවසන් වීමට හේතු
පරිශීලකයා program එකක් ආරම්භ කිරීම OS සේවාවක් ආරම්භ කිරීම Parent process එක child එකක් නිර්මාණය කිරීම නව batch job එකක් ආරම්භ කිරීම	සාමාන්‍යයෙන් කාර්යය අවසන් වීම දෝෂයක් / අනවසර memory access අවශ්‍ය සම්පත් නොලැබීම පරිශීලකයා/OS එක terminate කිරීම අවසර දී ඇති කාල සීමාව ඉක්මවීම

29. Process Control Block — PCB

එක් ක්‍රියාවලියක් කළමනාකරණය කිරීමට අවශ්‍ය තොරතුරු OS එක තබාගන්නා **දත්ත ව්‍යුහයයි**. එක් ක්‍රියාවලියකට අදාළ PCB එකක් පවතී.

තොරතුර	කාර්යය
PID / Process State	ක්‍රියාවලිය හඳුනාගැනීම / වර්තමාන තත්ත්වය
Program Counter (PC)	ඊළඟ ක්‍රියාත්මක කළ යුතු උපදෙසේ ලිපිනය
CPU Registers	නවත්වන මොහොතේ CPU register අගයන්
Scheduling Information	Priority, queue යොමු සහ අදාළ තොරතුරු
Memory Information	වෙන් කළ මතකය / page table යොමු
I/O / Accounting	විවෘත ගොනු, උපාංග, CPU භාවිත කාලය ආදිය

PCB හි නිශ්චිත ක්ෂේත්‍ර OS එක අනුව වෙනස් වේ. ක්‍රියාවලිය අවසන් වූ පසු අවශ්‍ය පිරිසිදු කිරීම/තොරතුරු ලබාගැනීමෙන් අනතුරුව PCB සම්පත් නිදහස් කෙරේ.

Q37. ඊළඟ උපදෙසේ ලිපිනය තබන්නේ: .....
ක්‍රියාවලිය අනන්‍යව හඳුනාගන්නේ: .....

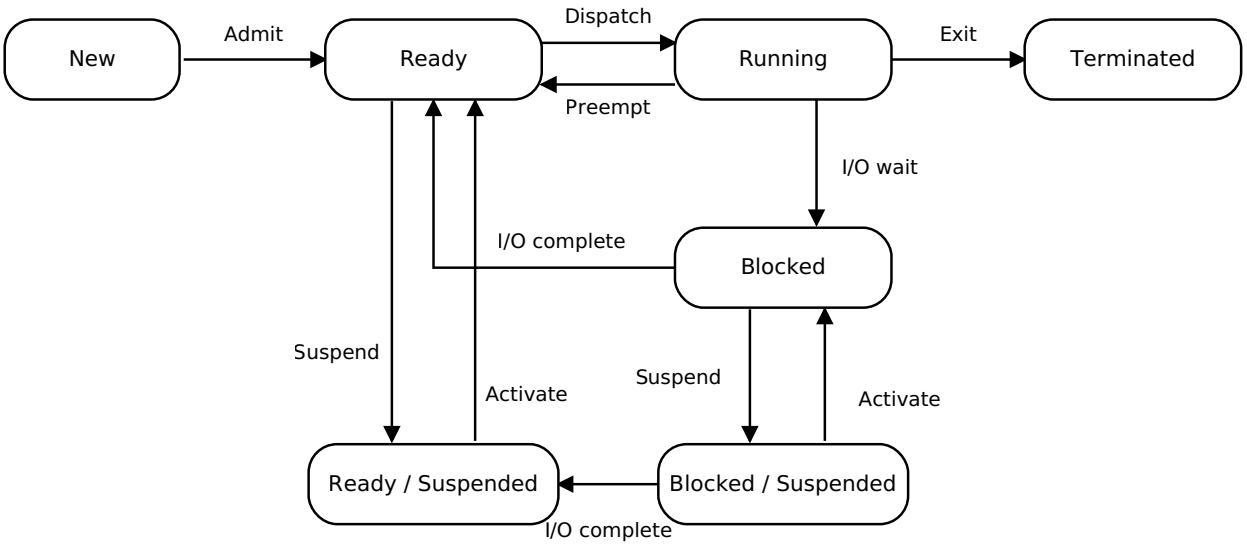
Q38. CPU ගණනය කිරීම් සඳහා වැඩි කාලයක් යොදන process එක ..... වේ.

PROCESS STATES

30. තත්ත්ව හත හඳුනාගනිමු

තත්ත්වය	අර්ථය
New	ක්‍රියාවලිය නිර්මාණය වෙමින් පවතී.
Ready	CPU හැර අනෙක් අවශ්‍යතා සපුරා CPU ලැබෙන තෙක් බලා සිටී.
Running	CPU මත උපදෙස් ක්‍රියාත්මක කරයි.
Blocked / Waiting	I/O හෝ වෙනත් event එකක් සම්පූර්ණ වන තෙක් බලා සිටී.
Terminated / Exit	ක්‍රියාවලිය අවසන් වී ඇත.
Ready / Suspended	CPU සඳහා සූදානම් නමුත් ප්‍රධාන මතකයෙන් ඉවත් කර ඇත.
Blocked / Suspended	Event එකක් බලා සිටින අතර ප්‍රධාන මතකයෙන්ද ඉවත් කර ඇත.

31. ප්‍රධාන සංක්‍රාන්ති



සරල අධ්‍යයන ආකෘතියකි. සමහර ආකෘතිවල New → Ready/Suspended සහ Running → Ready/Suspended වැනි අමතර මාර්ග දැක්විය හැකිය.

Ready → CPU ලැබීම බලා සිටී.
Blocked → Event/I/O සම්පූර්ණ වීම බලා සිටී.
Suspended → ක්‍රියාවලිය ප්‍රධාන මතකයෙන් ඉවත් කර ඇත.

Q39. ප්‍රශ්නය: I/O අවසන් වූ ක්‍රියාවලියක් සෘජුවම Running වෙත නොයන්නේ ඇයි?

STATE TRANSITIONS

32. මාරුවට හේතුව

සංක්‍රාන්තිය	හේතුව / සිදුවීම
New → Ready	Admit: ක්‍රියාවලිය ක්‍රියාත්මක වීමට පිළිගැනීම
Ready → Running	Dispatch: CPU ලබාදීම
Running → Ready	Preemption: time quantum අවසන් වීම වැනි හේතුවකින් CPU නැවත ලබාගැනීම
Running → Blocked	I/O ඉල්ලීම / event එකක් සඳහා බලා සිටීම
Blocked → Ready	I/O / event සම්පූර්ණ වීම
Running → Terminated	ක්‍රියාවලිය අවසන් වීම
Ready ↔ Ready/Suspended	Suspend: පිටතට මාරු කිරීම / Activate: RAM වෙත නැවත ගෙන ඒම
Blocked ↔ Blocked/Suspended	I/O තවම අවසන් නොවී ඇති විට Suspend / Activate කිරීම
Blocked/Suspended → Ready/ Suspended	පිටත සිටියදී I/O / event සම්පූර්ණ වීම

P ක්‍රියාවලිය නිර්මාණය කර පිළිගනී. CPU ලැබී ධාවනය වේ. Disk read එකක් ඉල්ලයි. එය අවසන් වී නැවත CPU ලැබුණු පසු කාර්යය අවසන් කරයි.

Q40. New → ..... → ..... → .....
→ ..... → ..... → Terminated

Q41. වගුව සම්පූර්ණ කරන්න.

පෙර තත්ත්වය	සිදුවීම	පසු තත්ත්වය
(1) Running	Time quantum අවසන් වේ.	.....
(2) Blocked	ප්‍රධාන මතකයෙන් ඉවත් කරයි.	.....
(3) Blocked/Suspended	I/O අවසන් වේ.	.....
(4) Ready/Suspended	RAM වෙත ගෙන එයි.	.....

Q42. Ready සහ Blocked අතර ප්‍රධාන වෙනස කුමක්ද?

.....

.....

එක් CPU core එකක් මත එක් මොහොතක Running වන්නේ එක් ක්‍රියාවලියක්/Thread එකක් පමණි.

CONTEXT & INTERRUPTS

33. Context Switching

ධාවනය වන ක්‍රියාවලියේ CPU තත්ත්වය සුරක්ෂිත කර, වෙනත් ක්‍රියාවලියක සුරැකි තත්ත්වය නැවත CPU වෙත පටවා ධාවනය මාරු කිරීමයි.

පියවර	P0 සිට P1 වෙත මාරුව
1. Save	P0 හි PC සහ register අගයන් එයට අදාළ PCB/context තුළ සුරකියි.
2. Update	P0 හි state සහ queue තොරතුරු යාවත්කාලීන කරයි.
3. Select	Scheduler එක Ready ක්‍රියාවලියක් වන P1 තෝරයි.
4. Restore	P1 හි PC, registers සහ අවශ්‍ය context නැවත පටවයි.
5. Resume	P1, නතර කළ ස්ථානයේ සිට නැවත ක්‍රියාත්මක වේ.

P0 running → Save P0 → Restore P1 → P1 running
P1 running → Save P1 → Restore P0 → P0 resumes

Overhead: Context මාරු කිරීමට කාලයක් වැය වේ; ඒ අතර user process එකක ප්‍රයෝජනවත් උපදෙස් ක්‍රියාත්මක නොවේ. නමුත් multitasking සඳහා එය අවශ්‍ය වේ.

34. Interrupt — අතුරුබිඳුම

CPU හි අවධානය අවශ්‍ය සිදුවීමක් දැනුම් දෙන සංඥාවකි. එවිට CPU එක අදාළ **Interrupt Service Routine (ISR)** වෙත පාලනය මාරු කරයි.

උදාහරණ: I/O completion interrupt, timer interrupt. System call එකක් වැඩසටහන මඟින් හිතාමතා ඉල්ලන OS සේවාවකි; exception එකක් උපදෙසක් නිසා ඇති වන සිදුවීමකි (උදා: divide-by-zero).

I/O අතුරුබිඳුම හැසිරවීමේ පියවර

① ක්‍රියාවලිය I/O ඉල්ලයි → එය Blocked වේ.
② OS එක වෙනත් Ready ක්‍රියාවලියක් ධාවනය කළ හැකිය.
③ I/O අවසන් වූ විට උපාංගය interrupt එකක් දන්වයි.
④ CPU/OS අවශ්‍ය context සුරකින අතර ISR එක සිදුවීම හසුරුවයි.
⑤ I/O බලා සිටි ක්‍රියාවලිය Ready වේ.
⑥ Scheduler තීරණය අනුව පෙර ක්‍රියාවලිය හෝ වෙනත් ක්‍රියාවලියක් ධාවනය වේ.

වැදගත්: සෑම interrupt එකකදීම වෙනත් process එකකට context switch වීම අනිවාර්ය නොවේ. ISR එක අවසානයේ එම process එකම නැවත ධාවනය විය හැකිය.

Q43. නැවත ආරම්භ කළ යුතු උපදෙස හඳුනාගැනීමට අවශ්‍ය අගය: .....

Q44. I/O අවසන් වූ process එකේ නව තත්ත්වය: .....

Q45. Context Switching කාලය overhead වන්නේ ඇයි?

PROCESS SCHEDULING

35. Process Scheduling

කුමන ක්‍රියාවලියට, කුමන අවස්ථාවේ CPU ලබාදිය යුතුද යන්න තීරණය කිරීමයි. OS එක CPU/process තත්ත්ව වාර්තා තබා CPU ලබාදීම හා නැවත ලබාගැනීම පාලනය කරයි.

Scheduler	ප්‍රධාන කාර්යය	සාපේක්ෂ වාර ගණන
Long-term / Job	නව jobs පිළිගෙන Ready කාණ්ඩයට ඇතුළත් කරයි; multiprogramming මට්ටම පාලනය කරයි.	අඩු වාර ගණනක්
Medium-term	Suspend / Activate (swap out/in) කරමින් RAM හි ක්‍රියාවලි ගණන කළමනාකරණය කරයි.	අවශ්‍ය පරිදි
Short-term / CPU	Ready queue එකෙන් ඊළඟ CPU ලබාදිය යුතු ක්‍රියාවලිය තෝරයි.	ඉතා නිතර; ඉක්මනින් කළ යුතුය

36. Scheduling Policies

Non-preemptive	Preemptive
ක්‍රියාවලිය අවසන් වන තෙක් හෝ I/O වැනි හේතුවකින් CPU ස්වේච්ඡාවෙන් අත්හරින තෙක් CPU තබාගනී.	ක්‍රියාවලිය අවසන් වීමට පෙර OS එකට CPU නැවත ලබාගත හැකිය. උදා: time quantum අවසන් වීම.

37. කාර්යසාධන මිනුම්

මිනුම	අර්ථය
Turnaround Time	ඉදිරිපත් වූ මොහොතේ සිට සම්පූර්ණ වූ මොහොත දක්වා මුළු කාලය. Completion − Arrival.
Waiting Time	Ready queue තුළ බලා සිටි මුළු කාලය. I/O බලා සිටීම මෙයට ඇතුළත් නොවේ.
Response Time	Arrival සිට පළමු වරට CPU සේවාව ලැබෙන තෙක් කාලය.
Throughput	ඒකක කාලයක් තුළ අවසන් කළ ක්‍රියාවලි ගණන.

සාමාන්‍ය අරමුණ: waiting/response/turnaround අවම කිරීම සහ throughput ඉහළ දැමීම; අවශ්‍යතාව අනුව ප්‍රමුඛතා වෙනස් වේ.

Q46. RAM වෙතින් ක්‍රියාවලියක් ඉවත් කරන්නේ කුමන scheduler එකද?

Q47. Arrival = 2 ms; පළමු CPU සේවාව = 5 ms; Completion = 14 ms.
Response time = ... | Turnaround time = ...

Q48. කාලය 20 ms තුළ processes 5ක් අවසන් නම් Throughput එක සොයන්න.

Q49. Timer interrupt එකක් නිසා CPU නැවත ලබාගැනීම කුමන policy එකට අයත්ද?

Q50. Ready queue එකෙන් ඊළඟ process එක තෝරන්නේ ..... ය.

DEVICE MANAGEMENT

39. Device Management යනු කුමක්ද?

පරිගණකයේ **ආදාන හා ප්‍රතිදාන (I/O) උපාංග භාවිතය පාලනය හා සම්බන්ධීකරණය** මෙහෙයුම් පද්ධතිය මඟින් සිදු කිරීමයි.

OS එකේ කාර්යය	සරල අදහස
උපාංග තත්ත්වය සටහන් කිරීම	උපාංගය නිදහස්ද, භාවිතයේද, දෝෂයක් තිබේද යන්න හඳුනාගනී.
වෙන් කිරීම හා නිදහස් කිරීම	ක්‍රියාවලි සඳහා උපාංග ප්‍රවේශය ලබාදී, කාර්යයෙන් පසු නිදහස් කරයි.
I/O ඉල්ලීම් කළමනාකරණය	ඉල්ලීම් පෙළගස්වා දත්ත හුවමාරුව සම්බන්ධීකරණය කරයි.
Drivers සමඟ සන්නිවේදනය	උපාංගයට අදාළ විධාන ලබාදෙයි; සම්පූර්ණ වීම සහ දෝෂ හසුරුවයි.

40. Device Driver — උපාංග ධාවකය

OS එක සහ විශේෂිත දෘඪාංග උපාංගයක් අතර සන්නිවේදනයට උපකාර කරන මෘදුකාංගයකි. සාමාන්‍ය I/O ඉල්ලීම්, අදාළ උපාංගයට තේරෙන විධාන බවට පරිවර්තනය කරයි.

Application → Operating System → Device Driver → Device
Device status / completion → Driver → Operating System

උදාහරණය: ලේඛනයක් මුද්‍රණය කිරීමට OS එක print ඉල්ලීම ලබාගනී. Printer driver එක මුද්‍රණ යන්ත්‍රයට අවශ්‍ය ආකෘතිය/විධාන සකස් කිරීමට උපකාර කරයි.

සමහර drivers OS එක සමඟම ලැබේ. අනෙක්වා උපාංග නිෂ්පාදකයාගෙන් ස්ථාපනය කළ යුතුය.

41. Buffer — අවරෝධකය / තාවකාලික රඳවනය

දත්ත එක් ස්ථානයකින් තවත් ස්ථානයකට මාරු කරන අතරතුර තාවකාලිකව තබන මතක ප්‍රදේශයකි. Buffering යනු එම රඳවනය භාවිත කිරීමයි.

Buffer එකක් අවශ්‍ය ඇයි?

දත්ත සපයන වේගය සහ ලබාගන්නා වේගය වෙනස් විය හැකිය. Buffer එක දත්ත ටිකක් රඳවා එම වේග වෙනස සම්බන්ධීකරණය කරයි.

උදාහරණය: CPU එකෙන් දත්ත වේගයෙන් ලැබුණත් printer එක සෙමින් මුද්‍රණය කරයි. Buffer එකේ දත්ත තබා printer එකට හැකි වේගයෙන් ලබාදෙයි.

Buffer පිරුණොත්: නව දත්ත භාරගැනීම තාවකාලිකව නවත්වන්න/වේගය පාලනය කරන්න සිදු විය හැකිය. Buffer හිස් වූ විට නැවත දත්ත භාරගත හැකිය.

Q51. OS එක සහ දෘඪාංගය සම්බන්ධ කරන මෘදුකාංගය: .....

Q52. Buffer එක භාවිත කරන්නේ දත්ත ..... තබාගැනීමටය.

Q53. Buffer අවශ්‍ය වීමට ප්‍රධාන හේතුවක් ලියන්න.

.....

.....

SPOOLING

42. Spooling යනු කුමක්ද?

උපාංගයකින් ඉටු කළ යුතු **වැඩ හා ඒවායේ දත්ත තාවකාලික ගබඩාවක පෙළගස්වා**, උපාංගය සූදානම් වූ විට ඒවා සැකසීමට යැවීමයි. සාමාන්‍ය උදාහරණය **Print Spooling** වේ.

Spool ගොනු සාමාන්‍යයෙන් **disk/SSD** මත තබයි. Spooler මෘදුකාංගය queue එක පාලනය කර උපාංගයට වැඩ ලබාදෙයි.

Applications A, B, C
↓ submit print jobs
Spool Queue: [Job A][Job B][Job C]
↓ Spooler sends the selected job
Printer: processes one job at a time

43. ක්‍රියා කරන ආකාරය

පියවර	සිදුවීම
01	ශිෂ්‍යයන් තිදෙනෙක් A, B, C ලේඛන මුද්‍රණයට යවයි.
02	OS එක එම print jobs spool ගොනු ලෙස ගබඩා කර queue එකට එක් කරයි.
03	Printer එක A මුද්‍රණය කරන අතර B සහ C queue එකේ රැඳේ.
04	A අවසන් වූ විට spooler එක ඊළඟ job එක යවයි.
05	Print job එක භාරදුන් පසු application එකට වෙනත් වැඩ දිගටම කළ හැකිය.

උදාහරණයේ arrival order භාවිත කර ඇත. සැබෑ queue එකක priority අනුව අනුපිළිවෙළ වෙනස් කළ හැකි අතර jobs pause/cancel කළ හැකිය.

44. Buffering සහ Spooling අතර වෙනස

Buffering	Spooling
හුවමාරු වන දත්ත කොටස් තාවකාලිකව රඳවයි.	සම්පූර්ණ වැඩ/ගොනු උපාංග සේවාව සඳහා පෙළගස්වයි.
ප්‍රධාන අරමුණ: වේග වෙනස්කම් සම්බන්ධීකරණය.	ප්‍රධාන අරමුණ: උපාංගය වැඩ කිහිපයක් අතර කාර්යක්ෂමව බෙදා භාවිත කිරීම.
සාමාන්‍යයෙන් RAM buffer භාවිත වේ.	සාමාන්‍යයෙන් disk/SSD spool ගොනු භාවිත වේ.

එකටත් භාවිත වේ: Spool queue එකෙන් තෝරාගත් print job එකේ දත්ත, printer එකට යැවීමේදී buffer එකක් හරහා යා හැකිය.

A මුද්‍රණය වෙමින් පවතී. B සහ C queue එකේය.

Q54. B සහ C තාවකාලිකව තබාගන්නා ක්‍රමය: .....

Q55. A අවසන් වූ විට, arrival order අනුව යැවෙන්නේ: .....

Q56. Spooling නිසා පරිශීලකයාට ලැබෙන වාසියක් ලියන්න.

.....

.....

EVOLUTION OF OS

46. No Operating System — මුල් යුගය

1940 දශකය හා 1950 දශකයේ මුල් කාලය: මෙහෙයුම් පද්ධතියක් නොමැතිව වැඩසටහන් අතින් සකස් කර එකින් එක ධාවනය කළේය. පරිශීලකයා/ක්‍රමලේඛකයා දෘඩාංග සමඟ සෘජුව කටයුතු කළේය.

- එක් කාර්යයකට පසු අනෙක් කාර්යය සිදු කළ **Serial Processing** ක්‍රමය භාවිත විය.
- උපදෙස්/දත්ත ලබාදීමට switches, punched cards වැනි ක්‍රම භාවිත විය.
- Program සකස් කරන සහ I/O සිදුවන කාලයේ CPU බොහෝ විට අලස විය.

Q57. වැඩසටහන් අතර මාරුව සිදු කළේ ..... ආකාරයටය.
මෙහි ප්‍රධාන සීමාවක්: .....

47. Simple Batch System — සරල කාණ්ඩ පද්ධතිය

සමාන අවශ්‍යතා ඇති jobs කාණ්ඩ කර, **පරිශීලකයාගේ සෘජු අන්තර්ක්‍රියාවකින් තොරව එකින් එක** ක්‍රියාත්මක කරයි. Resident monitor/මුල් OS එක ඊළඟ job එක ස්වයංක්‍රීයව ආරම්භ කිරීමට උපකාර කරයි.

Jobs → Batch → Job 1 → Job 2 → Job 3 → Output

- Punched cards, magnetic tapes වැනි මාධ්‍ය භාවිත විය.
- Jobs අතර අතින් කරන සැකසුම් කාලය අඩු විය.
- සාමාන්‍යයෙන් එක් user job එකක් වරකට ක්‍රියාත්මක විය.
- I/O බලා සිටින විට CPU අලස විය හැකිය; ප්‍රතිඵල ලැබීමට ප්‍රමාදයක් තිබිණි.

Q58. Batch processing අතරතුර පරිශීලක අන්තර්ක්‍රියාව .....

48. Multiprogrammed Batch System

ප්‍රධාන මතකයේ **වැඩසටහන් කිහිපයක්** තබාගනී. එකක් I/O බලා සිටින විට CPU එකට වෙනත් Ready වැඩසටහනක් ලබාදෙයි.

P1 waits for I/O → CPU runs P2 → P2 waits → CPU runs P3

- CPU අලස කාලය අඩු කර **CPU utilization** සහ **throughput** වැඩි කිරීමට උපකාර කරයි.
- Memory management සහ scheduling අවශ්‍ය වේ.
- CPU එක සැමවිටම 100% ක්‍රියාකාරීව පවතින බව සහතික නොවේ.

Q59. මෙහි CPU අලස කාලය අඩු කරන්නේ .....

Q60. Multiprogramming සඳහා RAM තුළ වැඩසටහන් ..... තබාගනී.

OS CLASSIFICATION

49. Time Sharing System — කාල විභජන පද්ධතිය

CPU කාලය කුඩා **Time Slices / Time Quanta** ලෙස බෙදා processes අතර වේගයෙන් මාරු කරයි. පරිශීලකයන්ට/වැඩසටහන්වලට ඉක්මන් ප්‍රතිචාර ලබාදීම එහි ප්‍රධාන අරමුණකි.

CPU: [P1][P2][P3][P1][P2][P3] → Time

- **Context Switching** මඟින් processes අතර මාරු වේ.
- පරිශීලක අන්තර්ක්‍රියාවට හා multi-user සේවා සඳහා සුදුසුය.
- එක් core එකක මාරුව ඉතා වේගවත් නිසා වැඩ එකවර සිදුවන බවක් පෙනේ.

Q61. එක් ක්‍රියාවලියකට ලබාදෙන කුඩා කාල කොටස ..... ලෙස හැඳින්වේ.

50. පරිශීලක ගණන අනුව

Single-user — ඒක පරිශීලක	Multi-user — බහු පරිශීලක
වරකට එක් පරිශීලකයකුගේ භාවිතයට සකස් කළ පද්ධතියකි.	පරිශීලකයන් කිහිපදෙනෙකුට එකම කාල පරාසයේ පද්ධති සම්පත් භාවිත කිරීමට ඉඩ දෙයි.

Q62. Multi-user පද්ධතියක users අතර සම්පත් බෙදාදීම සහ ..... පාලනය වැදගත් වේ.

51. කාර්ය ගණන අනුව

Single-tasking — ඒක කාර්ය	Multitasking — බහු කාර්ය
වරකට එක් user task/program එකක් ක්‍රියාත්මක කරයි.	කාලය බෙදා processes/tasks කිහිපයක ප්‍රගතිය පවත්වාගනී. Cores කිහිපයක් තිබේ නම් සැබෑ parallel execution ද කළ හැකිය.

වර්ගීකරණ දෙක වෙන වෙනම බලන්න:

Single-user පද්ධතියක් අනිවාර්යයෙන් Single-tasking විය යුතු නැත.

සංයෝජනය	සරල අදහස
Single-user + Single-tasking	එක් user කෙනෙක්; වරකට එක් task එකක්. උදා: MS-DOS.
Single-user + Multitasking	එක් user කෙනෙක් වැඩසටහන් කිහිපයක් භාවිත කිරීම. උදා: ලේඛනයක් සකසන අතර සංගීතය ඇසීම.
Multi-user + Multitasking	Users කිහිපදෙනෙක් තම tasks ක්‍රියාත්මක කරති. උදා: බහු පරිශීලක UNIX/Linux server එකක්.

Q63. Multi-user හා Multitasking යන ලක්ෂණ දෙකම පවතින අවස්ථාවකට උදාහරණයක් ලියන්න.

OPERATING SYSTEMS / 20

THREADS & REAL TIME

52. Multithreading — බහු පොට

Thread එකක් යනු process එකක් තුළ ඇති ක්‍රියාත්මක වීමේ ප්‍රවාහයකි. එකම process එකේ threads, code/ data වැනි සම්පත් බෙදාගන්නා අතර තමන්ගේ **PC, registers සහ stack** පවත්වාගනී.

Multithreading: එක් process එකක් තුළ threads කිහිපයක් ක්‍රියාත්මක කිරීමයි. එක් core එකක කාලය බෙදාගෙනද, cores කිහිපයක parallel ලෙසද ක්‍රියා කළ හැකිය.

One Process → Thread A: User interface
→ Thread B: Background data processing

• ප්‍රතිචාරශීලීත්වය වැඩි කිරීමට සහ සම්පත් බෙදා භාවිතයට උපකාර කරයි.
• Threads වැඩි කිරීමෙන් පමණක් වේගය අනිවාර්යයෙන් වැඩි නොවේ.

Q64. එකම process එකේ threads බෙදාගන්නා සම්පතක්: .....

53. Real-Time Operating System — තත්‍ය කාල OS

නියමිත කාල සීමා (deadlines) තුළ විශ්වාසදායක හා පුරෝකථනය කළ හැකි ප්‍රතිචාර ලබාදීමට සැලසුම් කළ පද්ධතියකි. නිවැරදි ප්‍රතිඵලයත් එය ලැබෙන වේලාවත් වැදගත් වේ.

Hard Real-Time	Soft Real-Time
අත්‍යවශ්‍ය deadline එකක් අතපසු වීම පද්ධති අසාර්ථකත්වයක් ලෙස සැලකේ. උදා: කාර්මික හදිසි ආරක්ෂක පාලනය.	Deadline අතපසු වීමෙන් සේවා ගුණාත්මකභාවය අඩුවෙයි. උදා: සජීවී multimedia playback.

Real-time යනු "වේගවත්" පමණක් නොවේ.

Q65. අවශ්‍ය ප්‍රතිචාරය ..... කාල සීමාව තුළ ලැබිය යුතුය.

Q66. නියමිත deadline එකක් තුළ ප්‍රතිචාර දැක්විය යුතු Real-Time පද්ධතියකට උදාහරණයක් ලියන්න.

54. මෙහෙයුම් පද්ධතියේ ප්‍රධාන කාර්ය

කාර්යය	සාරාංශය
Process Management	ක්‍රියාවලි නිර්මාණය, scheduling සහ context පාලනය.
Memory Management	මතකය වෙන් කිරීම, ආරක්ෂාව සහ virtual memory පාලනය.
File / Storage Management	ගොනු, නාමාවලි සහ ආචයන ඉඩ කළමනාකරණය.
Device Management	I/O උපාංග හා drivers සම්බන්ධීකරණය.
User Interface / Security	පරිශීලක අතුරුමුහුණත, authentication සහ අවසර පාලනය.

Q67. මෙහෙයුම් පද්ධතියක ප්‍රධාන කාර්ය තුනක් ලියන්න. .....

ප්‍රශ්නවලට පිළිතුරු

Q අංකය අනුව පුරවන කොටස් සහ අභ්‍යාස සඳහා පිළිතුරු. විවෘත සටහන් සඳහා ආදර්ශ පිළිතුරු දක්වා ඇත.

Q01. Page අංකය → Frame අංකය.

Q02. 5.

Q03. 3.

Q04. 2.

Q05. Page Fault.

Q06. Offset.

Q07. පිළිවෙළින්: Dirty Bit; Protection Bits; Present Bit.

Q08. Page Table Entry (PTE).

Q09. 8 ÷ 1 = 8 entries.

Q10. 8 × 2 = 16 bytes.

Q11. 32 ÷ 2 = 16 entries.

Q12. 2^8 = 256 නිසා බිටු 8.

Q13. 8 + 1 + 1 + 1 = බිටු 11.

Q14. 16 × 2 = 32 bytes.

Q15. Present: RAM තුළ තිබීම. Dirty: RAM වෙත ගෙනා පසු දත්ත වෙනස් වී තිබීම.

Q16. 2,000 + 250 = 2,250.

Q17. 4 KiB.

Q18. මෙහෙයුම් පද්ධතිය (OS).

Q19. Logical Address එක Physical Address එකකට පරිවර්තනය කිරීම.

Q20. Page Fault.

Q21. RAM ධාරිතාවට වඩා විශාල ලිපින අවකාශයක් අවශ්‍ය වැඩසටහන් ක්‍රියාත්මක කිරීමට ඉඩ දීම.

Q22. .pdf; PDF ලේඛනය; ගොනුව කියවීම.

Q23. Directory / නාමාවලිය.

Q24. Authentication; Authorization / Access Permissions.

Q25. Backup.

Q26. 10, 11, 12.

Q27. Linked Allocation.

Q28. 14.

Q29. ⌈13 ÷ 4⌉ = blocks 4.

Q30. වෙන් කළ ඉඩ = 16 KiB; අපතේ ගිය ඉඩ = 3 KiB.

Q31. පේළි පිළිවෙළින් (clusters; වෙන් කළ ඉඩ; අපතේ): 2; 4 KiB; 1 KiB / 3; 6 KiB; 0 KiB / 4; 8 KiB; 1 KiB.

Q32. 2 KiB.

Q33. 9 blocks; 4 blocks.

Q34. නොහැකිය. එක දිගට blocks 6ක් නොමැත; විශාලතම කොටස 4කි.

Q35. 24 KiB; 12 KiB.

Q36. දැනට නොහැකිය. නිදහස් ඉඩ 24 KiB එකට එක් කළහොත් 16 KiB වෙන් කළ හැකිය.

Q37. Program Counter; PID.

Q38. CPU-bound Process.

Q39. CPU ලබාදීමට scheduler එක තෝරා dispatch කළ යුතු බැවිනි.

Q40. Ready → Running → Blocked → Ready → Running.

Q41. පේළි පිළිවෙළින්: Ready; Blocked/Suspended; Ready/Suspended; Ready.

Q42. Ready: CPU බලා සිටී. Blocked: I/O/event බලා සිටී.

Q43. Program Counter අගය.

Q44. Ready.

Q45. Context save/restore සඳහා කාලය වැය වන අතර user process වැඩ ප්‍රගතියක් එම කාලයේ නොලැබේ.

Q46. Medium-term Scheduler.

Q47. Response = 5 − 2 = 3 ms; Turnaround = 14 − 2 = 12 ms.

Q48. 5 ÷ 20 = 0.25 processes/ms (processes 250/s).

Q49. Preemptive.

Q50. Short-term / CPU Scheduler.

Q51. Device Driver.

Q52. තාවකාලිකව.

Q53. දත්ත සපයන හා ලබාගන්නා උපාංගවල වේග වෙනස සම්බන්ධීකරණය.

Q54. Spooling / Spool Queue.

Q55. Job B.

Q56. Printer එක අවසන් වන තෙක් නොසිට වෙනත් වැඩ කළ හැකිය.

Q57. අතින්; වැඩසටහන් සැකසීමේදී සහ I/O අතර CPU අලස වීම.

Q58. නොමැත / සෘජු අන්තර්ක්‍රියාවක් නොමැත.

Q59. එකක් I/O බලා සිටියදී වෙනත් Ready program එකක් CPU වෙත ලබාදීමෙනි.

Q60. කිහිපයක්.

Q61. Time Slice / Time Quantum.

Q62. ප්‍රවේශ අවසර / ආරක්ෂාව.

Q63. උදා: එකම UNIX/Linux server එකට users කිහිපදෙනෙක් සම්බන්ධ වී තම වැඩ කිරීම.

Q64. Code / Data / එකම address space (එකක් ප්‍රමාණවත්).

Q65. නියමිත.

Q66. උදා: නියමිත deadline තුළ ප්‍රතිචාර දිය යුතු කාර්මික හදිසි ආරක්ෂක පාලනය.

Q67. Process, Memory, File/Storage, Device Management සහ User Interface/Security අතරින් කාර්ය තුනක්.