

සියලු ම හිමිකම් ඇවිරිණි /All Rights Reserved]

Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc
 Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc
 Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc
 Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc
 Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc
 Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc Ravindu Bandaranayake - ictfromabc

Information and Communication Technology

තොරතුරු හා සන්නිවේදන තාක්ෂණය

2026 QUIZ NO 535 MARKING

2026/05/29
ictfromabc
Ravindu Bandaranayake

1. One of the fundamental principles of the Von Neumann architecture is:
වොන් නියුමාන් ආකෘතියේ මූලික මූලධර්මවලින් එකක් වන්නේ:

- 1) Utilizing multiple processing units for parallel execution.
සමාන්තරව ක්‍රියාත්මක කිරීම සඳහා බහු සැකසුම් ඒකක භාවිතා කිරීම.
- 2) Storing both instructions and data in separate memory locations.
උපදෙස් සහ දත්ත යන දෙකම වෙනම මතක ස්ථානවල ගබඩා කිරීම.
- 3) Fetching and executing instructions sequentially from memory.
මතකයෙන් අනුපිළිවෙලින් උපදෙස් ලබා ගැනීම සහ ක්‍රියාත්මක කිරීම.
- 4) Employing a complex control unit for pipelined instruction processing.
මාර්ගගත උපදෙස් සැකසීම සඳහා සංකීර්ණ පාලන ඒකකයක් භාවිතා කිරීම.
- 5) Implementing a non-uniform memory access scheme.
ඒකාකාර නොවන මතක ප්‍රවේශ යෝජනා ක්‍රමයක් ක්‍රියාත්මක කිරීම.

Answer : 3)

Explanation / පැහැදිලි කිරීම:

The Von Neumann architecture is characterized by the use of a single address space for both instructions and data. This means that the CPU fetches instructions and data from the same memory unit. Because there's only one pathway (or bus) to this memory, the CPU must fetch an instruction, execute it, and then fetch the next instruction or the necessary data. This sequential process of fetching and executing instructions is a fundamental principle of the Von Neumann architecture.

වොන් නියුමාන් ආකෘතිය සංලක්ෂිත වන්නේ උපදෙස් සහ දත්ත යන දෙකම සඳහා තනි ලිපින අවකාශයක් භාවිතා කිරීමෙනි. මෙයින් අදහස් කරන්නේ CPU එකම මතක ඒකකයෙන් උපදෙස් සහ දත්ත ලබා ගන්නා බවයි. මෙම මතකයට එක් මාර්ගයක් පමණක් ඇති බැවින්, CPU විසින් උපදෙස් ලබා ගත යුතු සහ එය ක්‍රියාත්මක කළ යුතු අතර, ඊළඟ උපදෙස් හෝ අවශ්‍ය දත්ත ලබා ගත යුතුය. උපදෙස් ලබා ගැනීමේ සහ ක්‍රියාත්මක කිරීමේ මෙම අනුක්‍රමික ක්‍රියාවලිය වොන් නියුමාන් ආකෘතියේ මූලික මූලධර්මයකි.

Therefore, the correct answer is 3).

එබැවින්, නිවැරදි පිළිතුර 3) වේ.

2. The second and third rows of the following table contain two English words and their binary representations according to the ASCII code. The binary representation of Run! is kept blank.

පහත වගුවේ දෙවන සහ තෙවන පේළි වල ඉංග්‍රීසි වචන දෙකක් සහ ASCII කේතයට අනුව ඒවායේ ද්වීමය නිරූපණයන් අඩංගු වේ. Run! හි ද්වීමය නිරූපණය හිස්ව තබා ඇත.

Word	Binary Representation / ද්වීමය නිරූපණය
go	01100111 01101111
R!	01010010 00100001
Run!	?

What is the correct replacement for the blank / හිස් තැනට නිවැරදි ආදේශකය කුමක්ද?

- 1) 01010010 01110101 01101110 00100001
- 2) 01010010 01110101 01101110 00100001
- 3) 01010010 01101111 00100001 00100001
- 4) 01010010 01010101 00100001 01110101
- 5) 01010010 01110101 01101111 01110101

Answer: 2)

Explanation / පැහැදිලි කිරීම:

We are told / අපට මෙසේ කියනු ලැබේ:

Word	Binary Representation / ද්වීමය නිරූපණය
go	01100111 01101111
R!	01010010 00100001
Run!	?

We need to find the ASCII binary for the word "Run!".

අපි "Run!" යන වචනය සඳහා ASCII ද්වීමය නිරූපණය සොයා ගත යුතුය.

Step-by-step Breakdown / පියවරෙන් පියවර බිඳවැටීම:

Word: Run!

It has 4 characters / එහි අක්ෂර 4 ක් ඇත:

R u n !

We convert each character to ASCII binary

අපි සෑම අක්ෂරයක්ම ASCII ද්වීමය බවට පරිවර්තනය කරමු:

Character	ASCII Decimal	ASCII Binary
R	82	01010010 (already given / දැනටමත් ලබා දී ඇත)
u	117	01110101
As the binary code of "o" is given we can find this easily by just counting forward. "o" හි ද්වීමය කේතය ලබා දී ඇති බැවින් අපට මෙය ඉඳිරියට ගණනය කිරීමෙන් පහසුවෙන් සොයාගත හැකිය.		
n	110	01101110
As the binary code of "o" is given we can find this easily by just subtracting only 1 bit. "o" හි ද්වීමය කේතය ලබා දී ඇති බැවින් අපට මෙය බිට් 1ක් පමණක් අඩු කිරීමෙන් පහසුවෙන් සොයාගත හැකිය.		
!	33	00100001 (already given / දැනටමත් ලබා දී ඇත)

Therefore, the answer is 2)

එබැවින් පිළිතුර 2) වේ.

3. A smart greenhouse system uses a soil moisture sensor (A), a rain sensor (B), a temperature sensor (C), a humidity sensor (D), and a light sensor (E) to decide watering. It turns on the sprinkler if the soil is dry ($A=0$), either it is not raining ($B=0$) or the humidity is low ($D=0$), the temperature is hot ($C=1$) and it is not too bright ($E=0$).

සුහුරු හරිතාගාර පද්ධතියක් ජලය දැමීම තීරණය කිරීම සඳහා පාංශු තෙතමනය සංවේදකයක් (A), වැසි සංවේදකයක් (B), උෂ්ණත්ව සංවේදකයක් (C), ආර්ද්‍රතා සංවේදකයක් (D) සහ ආලෝක සංවේදකයක් (E) භාවිතා කරයි. පස වියළි නම් ($A=0$), වැසි නොවැටෙන්නේ නම් ($B=0$) හෝ ආර්ද්‍රතාවය අඩු නම් ($D=0$), උෂ්ණත්වය උණුසුම් නම් ($C=1$) සහ ආලෝකය වැඩියෙන් නැතැම් ($E=0$), එය ජල ඉසිනය ක්‍රියාත්මක කරයි.

Which Boolean expression represents this logic?

මෙම තර්කනය නිවැරදිව නියෝජනය කරන්නේ කුමන බුලියානු ප්‍රකාශනයද?

- 1) $\overline{A}B\overline{D}C\overline{E}$ 2) $\overline{A}(B + \overline{D})C\overline{E}$ 3) $\overline{A}(\overline{B} + \overline{D})C\overline{E}$
 4) $(A + B + D)(\overline{C} + \overline{E})$ 5) $(\overline{A} + \overline{B} + \overline{D})(C + \overline{E})$

Answer: 3)

Explanation / පැහැදිලි කිරීම

The sprinkler is turned on if the / ජල ඉසිනය ක්‍රියාත්මක වීමට නම්:

- Soil is dry / පස වියළි නම්:
 $\overline{A}$ (because $A=0$, which is $\overline{A}$ in Boolean terms because $\overline{A}$ represents the NOT operation).
 $\overline{A}$ ($A = 0$ නිසා, එය බුලිය පද වලින් $\overline{A}$ වේ. මන්ද $\overline{A}$, NOT මෙහෙයුම නියෝජනය කරයි).
- Either it is not raining or humidity is low / වර්ෂාව නොලැබීම හෝ ආර්ද්‍රතාවය අඩු වීම:
 This can be expressed as $\overline{B} + \overline{D}$ ($B = 0$, $\overline{B}$ in Boolean / $D = 0$, $\overline{D}$ in Boolean and one of them must be true).
 මෙය $\overline{B} + \overline{D}$ ලෙස ප්‍රකාශ කළ හැකිය ($B = 0$, බුලිය පද වලින් $\overline{B}$ වේ / $D = 0$, බුලිය පද වලින් $\overline{D}$ වේ සහ ඒවායින් එකක් පමණක් සත්‍ය විය යුතුයි).
- Temperature is hot / උෂ්ණත්වය උණුසුම්:
 C , because $C=1$.
 $C=1$ නිසා, C වේ.
- Not too bright / ආලෝකය වැඩියෙන් නැත:
 $\overline{E}$ (because $E=0$, $\overline{E}$ in Boolean).
 $\overline{E}$ ($E = 0$, බුලිය පද වලින් $\overline{E}$ වේ).

Combining everything / සියල්ල එකතුවීම කිරීමෙන්:

The Boolean expression should be / බුලිය ප්‍රකාශනය විය යුත්තේ: $\overline{A}(\overline{B} + \overline{D})C\overline{E}$

Therefore, the correct answer is 3).

එබැවින්, නිවැරදි පිළිතුර 3) වේ.

4. Which file system feature allows for the protection of files from unauthorized access?
අනවසර පිවිසුමෙන් ගොනු ආරක්ෂා කිරීමට ඉඩ දෙන ගොනු පද්ධති විශේෂාංගය කුමක් ද?

1) Journaling 2) Permissions 3) Encryption 4) Compression 5) Snapshots

Answer: 2)

The correct answer is 2) Permissions.

නිවැරදි පිළිතුර 2) Permissions වේ.

File system permissions control who can access specific files and directories, ensuring that only authorized users can perform operations such as reading, writing, or executing files.

ගොනු පද්ධති අවසර මඟින් නිශ්චිත ගොනු සහ නාමාවලි වෙත ප්‍රවේශ විය හැක්කේ කාටද යන්න පාලනය කරයි, බලයලත් පරිශීලකයින්ට පමණක් ගොනු කියවීම, ලිවීම හෝ ක්‍රියාත්මක කිරීම වැනි මෙහෙයුම් සිදු කළ හැකි බව සහතික කරයි.

Other options / වෙනත් විකල්ප:

Option / විකල්පය 1)

Journaling is a feature that ensures file system consistency in case of system crashes or power failures.

ජර්නලින් (Journaling) යනු පද්ධති බිඳවැටීම් හෝ බල බිඳවැටීම් වලදී ගොනු පද්ධති අනුකූලතාව සහතික කරන විශේෂාංගයකි.

Option / විකල්පය 3)

Encryption protects data by converting it into a coded format, but it is not the same as permissions.

ගුප්ත කේතනය මඟින් දත්ත සංකේතාත්මක ආකෘතියකට පරිවර්තනය කිරීමෙන් ආරක්ෂා කරයි, නමුත් එය අවසර වලට සමාන නොවේ.

Option / විකල්පය 4)

Compression reduces the size of files to save storage space, but it does not provide access control.

සම්පීඩනය ගබඩා ඉඩ ඉතිරි කිරීම සඳහා ගොනු වල ප්‍රමාණය අඩු කරයි, නමුත් එය ප්‍රවේශ පාලනය ලබා නොදේ.

Option / විකල්පය 5)

Snapshots creates point-in-time copies of the file system, but it does not directly handle access control.

Snapshots මඟින් ගොනු පද්ධතියේ ලක්ෂ්‍ය-කාලීන පිටපත් නිර්මාණය කරයි, නමුත් එය ප්‍රවේශ පාලනය සෘජුව හසුරුවන්නේ නැත.

5. Which of the following SQL queries will not return the second highest salary from the employees table?
පහත සඳහන් කුමන SQL විමසුම් වලින් සේවක වගුවෙන් දෙවන ඉහළම වැටුප ආපසු ලබා නොදෙන්නේද?

- 1) `SELECT MAX(salary) FROM employees
WHERE salary < (SELECT MAX(salary) FROM employees);`
- 2) `SELECT DISTINCT salary FROM employees
ORDER BY salary DESC LIMIT 1 OFFSET 1;`
- 3) `SELECT salary FROM employees
ORDER BY salary DESC LIMIT 1,1;`
- 4) `SELECT salary FROM employees ORDER BY salary LIMIT 1 OFFSET (SELECT
COUNT(DISTINCT salary) FROM employees) - 2;`
- 5) `SELECT salary FROM employees WHERE salary < (SELECT MAX(salary) FROM
employees)
ORDER BY salary DESC LIMIT 1;`

Answer : 4)

Let's examine each query / එක් එක් විමසුම පරීක්ෂා කරමු.

Option / විකල්ප 1)

This query finds the maximum salary that is less than the overall maximum salary, which correctly identifies the second highest salary. This query will return the second highest salary.

මෙම විමසුම මඟින් දෙවන ඉහළම වැටුප නිවැරදිව හඳුනා ගන්නා සමස්ත උපරිම වැටුපට වඩා අඩු උපරිම වැටුප සොයා ගනී. මෙම විමසුම දෙවන ඉහළම වැටුප ආපසු ලබා දෙනු ඇත.

Option / විකල්ප 2)

This query selects distinct salaries, orders them in descending order, and skips the highest salary (OFFSET 1), then retrieves the next one. This effectively returns the second highest salary. This query will return the second highest salary.

මෙම විමසුම වෙනස් වැටුප් තෝරන අතර, ඒවා අවරෝහණ අනුපිළිවෙලට ඇණවුම් කරයි, සහ ඉහළම වැටුප මත හරියි (OFFSET 1) ඉන්පසු ඊළඟ එක ලබා ගනී. මෙය එලඳායි ලෙස දෙවන ඉහළම වැටුප ආපසු ලබා දෙයි. මෙම විමසුම දෙවන ඉහළම වැටුප ආපසු ලබා දෙනු ඇත.

Option / විකල්ප 3)

This query orders salaries in descending order and uses LIMIT 1,1, which skips the highest salary and returns the next one. This correctly retrieves the second highest salary. This query will return the second highest salary.

මෙම විමසුම අවරෝහණ අනුපිළිවෙලින් වැටුප් ඇණවුම් කරන අතර LIMIT 1,1 භාවිතා කරයි, එය ඉහළම වැටුප මත හැර ඊළඟ එක ආපසු ලබා දෙයි. මෙය නිවැරදිව දෙවන ඉහළම වැටුප ලබා ගනී. මෙම විමසුම දෙවන ඉහළම වැටුප ලබා දෙයි.

Option / විකල්ප 4)

This query orders salaries in ascending order and uses a complex OFFSET calculated as (SELECT COUNT(DISTINCT salary) FROM employees) - 2. If there are duplicate salaries or if the calculation does not align with the data distribution, this may not accurately return the second highest salary. It could be incorrect if the OFFSET calculation is wrong or if it doesn't handle distinct values properly. This query might not return the second highest salary reliably.

මෙම විමසුමෙන් වැටුප් වැඩිවන අනුපිළිවෙලට ඇණවුම් කරන අතර (SELECT COUNT(DISTINCT salary) FROM employees) - 2. අනුපිටපත් වැටුප් තිබේ නම් හෝ ගණනය කිරීම දත්ත බෙදා හැරීම සමඟ නොගැලපේ නම්, මෙය නිවැරදිව ආපසු ලබා නොදෙනු ඇත. දෙවන ඉහළම වැටුප. OFFSET ගණනය වැරදි නම් හෝ එය වෙනස් අගයන් නිසි ලෙස හසුරුවන්නේ නැතිනම් එය වැරදි විය හැක. මෙම විමසුම දෙවන ඉහළම වැටුප විශ්වාසදායක ලෙස ආපසු ලබා නොදෙනු ඇත.

Option / විකල්ප 5)

This query selects salaries less than the maximum salary, orders them in descending order, and retrieves the highest of these, which is the second highest salary overall. This query will return the second highest salary.

මෙම විමසුම උපරිම වැටුපට වඩා අඩු වැටුප් තෝරන අතර, ඒවා අවරෝහණ අනුපිළිවෙලට ඇණවුම් කරයි, සහ මේවායින් ඉහළම වැටුප ලබා ගනී, එය සමස්තයක් ලෙස දෙවන ඉහළම වැටුප වේ. මෙම විමසුම දෙවන ඉහළම වැටුප ආපසු ලබා දෙනු ඇත.

Therefore, the correct answer is 4).

එබැවින්, නිවැරදි පිළිතුර 4).

6. Which of the following statements about Python parameters is correct?

පයිතන් පරාමිතින් පිළිබඳ පහත සඳහන් ප්‍රකාශ අතුරෙන් නිවැරදි වන්නේ කුමක්ද?

- 1) Parameters defined after * must always be passed as positional arguments.
* ට පසුව අර්ථ දක්වා ඇති පරාමිතින් සැම විටම ස්ථානීය තර්ක ලෙස සම්මත කළ යුතුය.
- 2) Parameters defined before / can only be passed as positional arguments, never as keywords.
/ ට පෙර අර්ථ දක්වා ඇති පරාමිතින් ස්ථානීය තර්ක ලෙස පමණක් සම්මත කළ හැකි අතර, කිසි විටෙකත් මූල පද ලෙස නොවේ.
- 3) Default parameters in Python must always come before non-default parameters in the function definition.
පයිතන් හි පෙරනිමි පරාමිතින් සැම විටම ශ්‍රිත අර්ථ දැක්වීමේදී පෙරනිමි නොවන පරාමිතින්ට පෙර පැමිණිය යුතුය.
- 4) A function cannot have both / and * in the same signature.
ශ්‍රිතයකට / සහ * යන දෙකම එකම signature එකක තිබිය නොහැක.
- 5) Parameters between / and * must always be passed as keyword-only.
/ සහ * අතර පරාමිතින් සැමවිටම මූල පද පමණක් ලෙස සම්මත කළ යුතුය.

Answer : 2)

Parameters defined before / can only be passed as positional arguments, never as keywords.

/ට පෙර අර්ථ දක්වා ඇති පරාමිතින් ස්ථානීය තර්ක ලෙස පමණක් සම්මත කළ හැකි අතර, කිසි විටෙකත් මූල පද ලෙස නොවේ.

Correct - නිවැරදියි

/ marks positional-only parameters.

/ ස්ථානීය-පමණක් පරාමිතින් සලකුණු කරයි.

Example:

```
def f(a, /, b):
    pass
f(1, 2) # works
f(a=1, b=2) # error, 'a' is positional-only
```

1. Incorrect. - වැරදියි

Parameters after * are keyword-only

* ට පසු පරාමිතින් මූල පද සඳහා පමණි.

Example:

```
def f(a, *, b):
    pass
f(1, b=2) # works
f(1, 2) # error, b must be keyword
```

3. Incorrect - වැරදියි

Actually, it's the opposite: non-default parameters must come before default ones.

ඇත්ත වශයෙන්ම, එය ප්‍රතිවිරුද්ධ දෙයයි: පෙරනිමි නොවන පරාමිතින් පෙරනිමි ඒවාට පෙර පැමිණිය යුතුය.

Example:

```
def f(a, b=10): # valid
    pass
# invalid
def f(a=10, b):
    pass
```

4. Incorrect - වැරදියි

Python does allow mixing / and *.

පයිතන් / සහ * මිශ්‍ර කිරීමට ඉඩ දෙයි.

Example:

```
def f(a, /, b, *, c):
    pass
f(1, 2, c=3) # works
```

5. Incorrect - වැරදියි

Parameters between / and * can be passed as positional OR keyword.

/ සහ * අතර පරාමිතින් ස්ථානීය OR මූල පදය ලෙස ලබා දිය හැක.

Example:

```
def f(a, /, b, *, c):
    pass
f(1, 2, c=3) # b passed as positional
f(1, b=2, c=3) # b passed as keyword
```

Therefore, the correct answer is 2).

එබැවින්, නිවැරදි පිළිතුර 2).

7. What will be the output of the following Python code?

පහත පයිතන් කේතයේ ප්‍රතිදානය කුමක් වේවිද?

```
def book_info(title, author="Unknown", year=2020):
    print(f'{title}' was written by {author} in {year}.")
book_info("The Alchemist", year=1988)
```

- 1) 'The Alchemist' was written by Unknown in 2020.
- 2) 'The Alchemist' was written by Unknown in 1988.
- 3) 'The Alchemist' was written by Paulo Coelho in 1988.
- 4) 'The Alchemist' was written by Unknown in Unknown.
- 5) 'The Alchemist' was written by Paulo Coelho in 2020.

Answer : 2)

Given below is the analysis of the code / කේතයේ විශ්ලේෂණය පහත දැක්වේ.

Function Definition / ශිෂ්‍ර අර්ථ දැක්වීම

The book_info function takes three arguments.

The book_info ශිෂ්‍ර ය තර්ක තුනක් ගනී.

title

The title of the book / පොතේ නි මාතෘකාව

author

The author of the book (default value is "Unknown")

පොතේ නි කර්තෘ (පෙරනිමි අගය "Unknown")

year

The year the book was published (default value is 2020)

පොත ප්‍රකාශයට පත් කළ වර්ෂය (පෙරනිමි අගය 2020)

When the function is called, it prints out a formatted string displaying the title of the book, the author, and the year it was written.

ශීතල කැඳින්වූ විට, එය පොතේ මාතෘකාව, කතෘ රසා සහ එය ලියා ඇති වර්ෂය ප්‍රදර්ශනය කරමින් ආකෘතිගත (formatted) string එකක් මුද්‍රණය කරයි.

book_info("The Alchemist", year=1988)

The title argument is provided as "The Alchemist".

මාතෘකා තර්කය "The Alchemist" ලෙස සපයා ඇත.

The author argument is not provided, so it defaults to "Unknown".

කර්තෘ තර්කය සපයා නැත, ඒබැවින් එය "Unknown" වෙත පෙරනිමි වේ.

The year argument is provided as 1988, overriding the default value.

පෙරනිමි අගය අභිබවා යමින් වසර තර්කය 1988 ලෙස සපයා ඇත.

The output will be / ප්‍රතිදානය වනුයේ

'The Alchemist' was written by Unknown in 1988.

The correct answer is 2) 'The Alchemist' was written by Unknown in 1988.

නිවැරදි පිළිතුර 2) 'The Alchemist' was written by Unknown in 1988 වේ.

8. In SSADM, which of the following best describes the purpose and outcome of the Logical Data Modeling step within the Analysis phase?

SSADM හි, විශ්ලේෂණ අදියර තුළ තාර්කික දත්ත ආකෘතිකරණ පියවරේ අරමුණ සහ ප්‍රතිඵලය වඩාත් හොඳින් විස්තර කරන්නේ පහත සඳහන් කවරක් ද?

- 1) To convert high-level business requirements into detailed system specifications, focusing on the hardware and software infrastructure required for implementation.
ක්‍රියාත්මක කිරීම සඳහා අවශ්‍ය දෘඩාංග සහ මෘදුකාංග යටිතල පහසුකම් කෙරෙහි අවධානය යොමු කරමින්, ඉහළ මට්ටමේ ව්‍යාපාරික අවශ්‍යතා සවිස්තරාත්මක පද්ධති පිරිවිතර බවට පරිවර්තනය කිරීම.
- 2) To create a detailed blueprint of the system's physical data storage, including file structures and indexing methods, to ensure efficient data retrieval and storage.
කාර්යක්ෂම දත්ත ලබා ගැනීම සහ ගබඩා කිරීම සහතික කිරීම සඳහා, ගොනු ව්‍යුහයන් සහ සුවිගත කිරීමේ ක්‍රම ඇතුළුව, පද්ධතියේ භෞතික දත්ත ගබඩාවේ සවිස්තරාත්මක සැලැස්මක් නිර්මාණය කිරීම.
- 3) To develop a conceptual model that represents the data structures and relationships in a manner independent of physical considerations, allowing for an abstract view of the data needs.
දත්ත අවශ්‍යතා පිළිබඳ විශුක්ත දර්ශනයකට ඉඩ සලසමින් භෞතික සලකා බැලීම්වලින් ස්වාධීනව දත්ත ව්‍යුහයන් සහ සම්බන්ධතා නියෝජනය කරන සංකල්පීය ආකෘතියක් සංවර්ධනය කිරීම.
- 4) To perform a feasibility analysis that assesses the practicality of proposed solutions in terms of cost, time, and technology constraints before proceeding to system design.
පද්ධති සැලසුම් කිරීමට පෙර පිරිවැය, කාලය සහ තාක්ෂණික සීමාවන් අනුව යෝජිත විසඳුම්වල ප්‍රායෝගිකත්වය තක්සේරු කරන ගණනා විශ්ලේෂණයක් සිදු කිරීම.
- 5) To evaluate the user interface design and interactions to ensure usability and accessibility requirements are met before the system is developed.
පරිශීලක අතුරුමුහුණත සැලසුම් කිරීම සහ පද්ධතිය සංවර්ධනය කිරීමට පෙර උපයෝගීතාවය සහ ප්‍රවේශන අවශ්‍යතා සපුරාලීම සහතික කිරීම සඳහා අන්තර්ක්‍රියා ඇගයීම.

Answer: 3)

In SSADM (Structured Systems Analysis and Design Method), the Logical Data Modeling step within the Analysis phase focuses on creating a conceptual representation of the data structures and their relationships.

SSADM (ව්‍යුහගත පද්ධති විශ්ලේෂණය සහ නිර්මාණකරණ ක්‍රමවේදය) හි, විශ්ලේෂණ අදියර තුළ තාර්කික දත්ත ආකෘතිකරණ පියවර දත්ත ව්‍යුහයන් සහ ඒවායේ සම්බන්ධතා පිළිබඳ සංකල්පීය නිරූපණයක් නිර්මාණය කිරීම කෙරෙහි අවධානය යොමු කරයි.

Explanation of each option / එක් එක් විකල්පය පැහැදිලි කිරීම:

Option / විකල්පය 1)

Describes the purpose of the System Design phase rather than the Analysis phase. It involves defining detailed system specifications and infrastructure requirements, not data modeling.

විශ්ලේෂණ අවධියට වඩා පද්ධති සැලසුම් අවධියේ අරමුණ විස්තර කරයි. එයට දත්ත ආකෘතිකරණය නොව සවිස්තරාත්මක පද්ධති පිරිවිතර සහ යටිතල පහසුකම් අවශ්‍යතා නිර්වචනය කිරීම ඇතුළත් වේ.

Option / විකල්පය 2)

This is related to the Physical Data Modeling step, which deals with the physical aspects of data storage and retrieval, not the logical data modeling.

මෙය තාර්කික දත්ත ආකෘතිකරණයට නොව දත්ත ගබඩා කිරීමේ සහ නැවත ලබා ගැනීමේ භෞතික අංශ සමඟ කටයුතු කරන භෞතික දත්ත ආකෘතිකරණ පියවරට සම්බන්ධ වේ.

Option / විකල්පය 3)

Accurately describes the Logical Data Modeling step. It focuses on creating a conceptual representation of data structures and relationships without considering physical implementation details.

තාර්කික දත්ත ආකෘතිකරණ පියවර නිවැරදිව විස්තර කරයි. භෞතික ක්‍රියාත්මක කිරීමේ විස්තර නොසලකා දත්ත ව්‍යුහයන් සහ සම්බන්ධතා පිළිබඳ සංකල්පය නිරූපණයක් නිර්මාණය කිරීම කෙරෙහි එය අවධානය යොමු කරයි.

Option / විකල්පය 4)

Describes the Feasibility Study phase, which assesses the practicality of solutions in terms of cost, time, and technology, not the Logical Data Modeling step.

තාර්කික දත්ත ආකෘතිකරණ පියවර නොව පිරිවිැය, කාලය සහ තාක්ෂණය අනුව විසඳුම්වල ප්‍රායෝගික බව තක්සේරු කරන ශක්‍යතා අධ්‍යයන අදියර විස්තර කරයි.

Option / විකල්පය 5)

Related to the User Interface Design phase, focusing on ensuring usability and accessibility, not data modeling.

පරිශීලක අතුරුමුහුණත් සැලසුමට අදාළව, දත්ත ආකෘතිකරණය නොව, උපයෝගීතාව සහ ප්‍රවේශ්‍යතාව සහතික කිරීම කෙරෙහි අවධානය යොමු කරයි.

The best description of the purpose and outcome of the Logical Data Modeling step within the Analysis phase is 3).

විශ්ලේෂණ අදියර තුළ තාර්කික දත්ත ආකෘතිකරණ පියවරේ අරමුණ සහ ප්‍රතිඵලය පිළිබඳ හොඳම විස්තරය වන්නේ 3).

9. What is the primary purpose of using PHP in web development?

වෙබ් සංවර්ධනයේදී PHP භාවිතා කිරීමේ මූලික අරමුණ කුමක්ද?

- 1) To style web pages / වෙබ් පිටු හැඩගැන්වීමට
- 2) To create dynamic web pages / ගතික වෙබ් පිටු නිර්මාණය කිරීමට
- 3) To manage databases directly / දත්ත සමුදායන් සෘජුවම කළමනාකරණය කිරීම
- 4) To create animations on web pages / වෙබ් පිටු මත සජීවීකරණ නිර්මාණය කිරීමට
- 5) To optimize images for faster loading / වේගවත් පූරණය සඳහා පින්තූර ප්‍රශස්ත කිරීමට

Answer: 2)

The correct answer is 2) To create dynamic web pages.

නිවැරදි පිළිතුර 2) ගතික වෙබ් පිටු නිර්මාණය කිරීමට.

Explanation / පැහැදිලි කිරීම

The primary purpose of using PHP in web development is to create dynamic web pages. PHP (Hypertext Preprocessor) is a server-side scripting language that allows developers to interact with databases, process forms, generate content dynamically, and perform various server-side tasks, enabling web pages to be more interactive and personalized.

වෙබ් සංවර්ධනයේදී PHP භාවිතා කිරීමේ මූලික අරමුණ වන්නේ ගතික වෙබ් පිටු නිර්මාණය කිරීමයි. PHP යනු සංවර්ධකයින්ට දත්ත සමුදායන් සමඟ අන්තර් ක්‍රියා කිරීමට, ආකෘති සකසීමට, ගතිකව අන්තර්ගතය උත්පාදනය කිරීමට සහ විවිධ සේවාදායක පාර්ශවික කාර්යයන් ඉටු කිරීමට ඉඩ සලසන සේවාදායක පැත්තේ ස්ක්‍රිප්ටින් භාෂාවකි.

10. What is the purpose of the <form> tag in HTML?

HTML හි <form> උසුලනයේ අරමුණ කුමක්ද?

- 1) To link pages / පිටු සම්බන්ධ කිරීමට
- 2) To format text / පාඨ හැඩතල ගැන්වීමට
- 3) To define a section of a document / ලේඛනයක කොටසක් අර්ථ දැක්වීමට
- 4) To collect user input / පරිශීලක ආදානය රැස් කිරීමට
- 5) To style tables / වගු හැඩගැන්වීමට

Answer: 4)

Explanation / පැහැදිලි කිරීම:

The <form> tag in HTML is used to create a section in a webpage where users can enter and submit data. It typically contains various input elements like text fields, checkboxes, radio buttons, and submit buttons. The data entered by the user is sent to a server for processing when the form is submitted.

HTML හි <form> උසුලනය පරිශීලකයින්ට දත්ත ඇතුළත් කර ඉදිරිපත් කළ හැකි වෙබ් පිටුවක කොටසක් සෑදීමට භාවිතා කරයි. එහි සාමාන්‍යයෙන් text ක්ෂේත්‍ර, checkbox, radio බොත්තම් සහ submit බොත්තම් වැනි විවිධ ආදාන මූලාංග අඩංගු වේ. පෝරමය ඉදිරිපත් කරන විට පරිශීලකයා විසින් ඇතුළත් කළ දත්ත සැකසීම සඳහා සේවාදායකයකට යවනු ලැබේ.

Let's explore the given options / ලබා දී ඇති විකල්ප ගවේෂණය කරමු.

1) To link pages / පිටු සම්බන්ධ කිරීමට

Hyperlinks are created between pages using the <a> tag.

<a> උසුලනය භාවිතයෙන් පිටු අතර අධිසන්නිධාන සාදනු ලැබේ.

2) To format text / පාඨ හැඩතල ගැන්වීමට

Formatting text (like bold, italic, etc.) is done using tags like , <i>, or CSS.

පාඨ හැඩතල ගැන්වීම (bold, italic ඇතුළත් වැනි) , <i>, හෝ CSS වැනි උසුලන භාවිතයෙන් සිදු කෙරේ.

3) To define a section of a document / ලේඛනයක කොටසක් අර්ථ දැක්වීමට

Defining a section of a document is done using tags like <section>, <div>, or <article>.

ලේඛනයක කොටසක් අර්ථ දැක්වීම <section>, <div> හෝ <article> වැනි උසුලන භාවිතයෙන් සිදු කෙරේ.

4) To collect user input / පරිශීලක ආදානය රැස් කිරීමට

The <form> tag is used to create a form where users can enter information like text, passwords, checkboxes, and radio buttons.

පරිශීලකයින්ට පාඨ, මුරපද, checkbox සහ radio බොත්තම් වැනි තොරතුරු ඇතුළත් කළ හැකි පෝරමයක් නිර්මාණය කිරීමට <form> උසුලනය භාවිතා කරයි, ඒවා පසුව ඉදිරිපත් කළ හැකිය.

5) To style tables / වගු හැඩගැන්වීමට

Styling tables is done using the <table>, <th>, and <td> tags along with CSS

CSS සමඟින් <table>, <th> සහ <td> උසුලනය භාවිතයෙන් වගු හැඩගැන්වීම සිදු කෙරේ.

Therefore, the answer is 4).

එබැවින්, පිළිතුර 4) වේ.

1. (a) A data processing team at a national weather station is upgrading its historical climate data records. The weather station tracks daily records for different provinces, storing them in text format (.txt) and CSV format (.csv). The following Python program processes these files to mark them as archived by adding a suffix and logs the summary.

ජාතික කාලගුණ විද්‍යා මධ්‍යස්ථානයක දත්ත සැකසුම් කණ්ඩායමක් විසින් එහි ඓතිහාසික දේශගුණික දත්ත වාර්තා යාවත්කාලීන කරනු ලබයි. මෙම කාලගුණ මධ්‍යස්ථානය විවිධ පළාත් සඳහා දෛනික වාර්තා, පෙළ ආකෘතියෙන් (.txt) සහ CSV ආකෘතියෙන් (.csv) ගබඩා කරයි. පහත දැක්වෙන Python වැඩසටහන මගින් මෙම ගොනු සංරක්ෂණය කර ඇති බව දැක්වීම සඳහා ප්‍රත්‍යයක් (suffix) එකතු කර නැවත නම් කර, එහි සාරාංශය සටහන් කර ගනියි.

```
data_registry = {
    "Western": ["colombo_2025.txt", "gampaha_temp.csv", "kalutara_rain.txt"],
    "Central": ["kandy_2025.txt", "nuwara_eliya.csv"]
}

ARCHIVE_TAG = "_archived"
LOG_FILE_NAME = "archive_log.txt"

def get_archived_name(filename):
    parts = filename.rsplit(".", 1)
    return parts[0] + ARCHIVE_TAG + "." + parts[1]

def archive_province_files(file_list):
    archived = []
    for f in file_list:
        archived.append(get_archived_name(f))
    return archived

summary = []
for province, files in data_registry.items():
    result = archive_province_files(files)
    summary.append({"province": province, "count": len(result), "files": result})

with open(LOG_FILE_NAME, "w") as log:
    for entry in summary:
        log.write(entry["province"] + " -> " + str(entry["count"]) + " files processed\n")
```

- i. Write a Python function `load_archive_log(log_file)` that reads `archive_log.txt` (written in the structure shown in the code above) and returns a dictionary where each key is a province name and the value is the number of processed files as an integer.

`archive_log.txt` ගොනුව (ඉහත කේතයේ දක්වා ඇති ව්‍යුහයට අනුව ලියැවුණු) කියවා, එහි සැම යතුරක්ම (key) පළාත් නාමයක් වන අතර අගය (value) සකසන ලද ගොනු ගණන පූර්ණ සංඛ්‍යාවක් (integer) ලෙස දැක්වෙන ගබ්දකෝෂයක් (dictionary) ආපසු ලබා දෙන `load_archive_log(log_file)` නමැති Python ශ්‍රිතය ලියන්න.

```
def load_archive_log(log_file):
    dictionary = {}
    with open(log_file, "r") as f:
        for line in f:
            # Splitting line by " -> " to separate province and message
            parts = line.strip().split(" -> ")
            if len(parts) == 2:
                province = parts[0]
                # Extracting the integer count from "X files processed"
                count = int(parts[1].split(" ")[0])
                dictionary[province] = count
    return dictionary
```

The log file saves lines in the format: "Western -> 3 files processed\n". The function opens the file in read mode ("r"), iterates through each line, and uses string manipulation (split()) to divide the province name from the message data. It extracts the numerical value, converts it to an integer using int(), and populates the dictionary.

ලොග් ගොනුව (log file) තුළ දත්ත ගබඩා වන්නේ "Western -> 3 files processed\n" යන රටාවටයි. මෙම ශ්‍රිතය මඟින් ගොනුව කියවීමේ මාදිලියෙන් ("r") විවෘත කර, එක් එක් පේළිය වෙන වෙනම ගෙන split() ක්‍රමය මඟින් පළාත් නාමය සහ දත්ත වෙන් කරගනී. ඉන්පසු ගොනු ගණන ලබාගෙන එය int() මඟින් පූර්ණ සංඛ්‍යාවකට හරවා dictionary එකට ඇතුළත් කරයි.

- ii. Fill in the blanks (A) – (D) to complete the function below. It should return a dictionary mapping each province to a status: "High Volume" if it has 3 or more files processed, otherwise "Standard Volume". පහත ශ්‍රිතය සම්පූර්ණ කිරීම සඳහා හිස්තැන් (A) සිට (D) තෙක් පුරවන්න. එය එක් එක් පළාත තත්ත්වයකට (status) සිතියම්ගත කරන dictionary එකක් ආපසු ලබා දිය යුතුය: ගොනු 3ක් හෝ ඊට වැඩි ගණනක් සකසා තිබේ නම් "High Volume" ලෙසද, එසේ නොමැති නම් "Standard Volume" ලෙසද ලබා දිය යුතුය.

```
def generate_volume_report(summary_list):
    report = _(A)_
    for item in summary_list:
        if item[_ (B) _] >= 3:
            status = "High Volume"
        else:
            status = "Standard Volume"
        report[_ (C) _] = _(D)_
    return report
```

- (A) {} or dict()
 (B) "count"
 (C) item["province"]
 (D) status

(A)
 initializes an empty dictionary to collect results.
 හිස් dictionary එකක් ආරම්භ කරයි.

(B)
 accesses the key "count" from the summary elements to check the condition.
 අගය 3ට වඩා වැඩිදැයි බැලීමට summary හි ඇති "count" යතුර (key) පරීක්ෂා කරයි.

(C)
 references the current province name using item["province"] to create the target key in the report dictionary.
 නව වාර්තා ගබඩකෝෂයේ (report dictionary) යතුර ලෙස පළාත් නාමය ලබා දීමට item["province"] යොදාගනී.

(D)
 assigns the calculated status variable string value to that key.
 අදාළ තත්ත්වය වන status විචල්‍යයෙහි අගය ඊට පවරයි.

- iii. Write the exact value stored in `summary[1]` after the execution of the main for loop finishes.
 ප්‍රධාන for උපය අවසන් වූ පසු `summary[1]` ලෙස ගබඩා වී ඇති නිශ්චිත අගය ලියන්න.

```
{"province": "Central", "count": 2, "files": ["kandy_2025_archived.txt", "nuwara_eliya_archived.csv"]}
```

`summary` is a list of dictionaries. Python lists use 0-based indexing. `summary[0]` stores data for the "Western" province, and `summary[1]` stores the information for the "Central" province. The strings within the "files" list have `_archived` successfully appended before their file extension.

`summary` යනු ලැයිස්තුවකි (List). එහි දර්ශකය (index) ආරම්භ වන්නේ 0 සිටය. එබැවින් `summary[0]` මගින් "Western" පළාතද, `summary[1]` මගින් "Central" පළාතට අදාළ දත්තද නිරූපණය වේ. මෙහිදී ගොනු නාමයන් අවසානයට `_archived` කොටස සාර්ථකව එකතු වී ඇත.

- iv. Write a Python function `count_txt_files(directory)` that lists all files in a given directory using the `os` module, identifies how many of those filenames end with the string ".txt", and returns that total count as an integer.

`os` මොඩියුලය භාවිතයෙන් දී ඇති නාමාවලියක (directory) ඇති සියලුම ගොනු නාමයන් ලැයිස්තුගත කර, ඒවායින් ".txt" වලින් අවසන් වන ගොනු ගණන කොපමණදැයි හඳුනාගෙන, එම මුළු එකතුව පූර්ණ සංඛ්‍යාවක් ලෙස ආපසු ලබා දෙන `count_txt_files(directory)` ශ්‍රිතය ලියන්න.

```
import os
```

```
def count_txt_files(directory):
    count = 0
    file_list = os.listdir(directory)
    for filename in file_list:
        if filename.endswith(".txt"):
            count += 1
    return count
```

The function implements standard filesystem interaction using the `os` module. `os.listdir(directory)` fetches all filenames inside the specified folder. A loop validates whether each file finishes with ".txt" using the `.endswith()` method (or alternative string slicing) and updates a counter.

ශ්‍රිතය `os` මොඩියුලය භාවිතයෙන් සම්මත ගොනු පද්ධති අන්තර්ක්‍රියා ක්‍රියාත්මක කරයි. `os.listdir(directory)` නිශ්චිත ෆෝල්ඩරය තුළ ඇති සියලුම ගොනු නාම ලබා ගනී. `.endswith()` ක්‍රමය (හෝ විකල්ප string slicing) භාවිතයෙන් සෑම ගොනුවක්ම ".txt" සමඟ අවසන් වේද යන්න ලපයක් වලංගු කරන අතර කවුන්ටරයක් යාවත්කාලීන කරයි.