සියලු ම හිමිකම් ඇවිරිණි / All Rights Reserved]

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

Information and Communication Technology
තොරතුරු හා සන්නිවේදන තාක්ෂණය

2026 QUIZ NO 350 MARKING

2025/11/25
ictfromabc
Ravindu Bandaranayake

1. Which of the following is incorrect about various computer types?
විවිධ පරිගණක වර්ග සම්බන්ධයෙන් පහත සඳහන් දෑ අතරින් කුමක් වැරදි වන්නේද?

    1) Super computers does parallel processing.
    සුපිරි පරිගණක සමාන්තර සැකසුම් සිදු කරයි.

    2) Mainframe computers are mainly made to handle huge number of input, output and storage data at once.
    ප්‍රධානරාමුවේ පරිගණක ප්‍රධාන වශයෙන් සැදී ඇත්තේ විශාල ආදාන, ප්‍රතිදානය සහ ගබඩා දත්ත එකවර හැසිරවීමටය.

    3) <u>Mainframe purpose computers are also known as dedicated computers.</u>
    ප්‍රධානරාමුවේ පරිගණක කැපවූ පරිගණක ලෙසද හැඳින්වේ.

    4) Analog computers are used to process continuous data.
    ප්‍රතිසම පරිගණක අඛණ්ඩ දත්ත සැකසීමට භාවිතා කරයි.

    5) Digital computers are used to process discrete data.
    අංකිත පරිගණක දත්ත විවික්ත කිරීමට භාවිතා කරයි.

Answer:3)

We can classify computers based on many things. Here we classified computers based on physical size.
බොහෝ කරුණු මත පදනම්ව අපට පරිගණක වර්ගීකරණය කළ හැකිය. මෙහිදී අපි භෞතික ප්‍රමාණය අනුව පරිගණක වර්ගීකරණය කරයි.

1. Super computers (සුපිරි පරිගණක)
2. Mainframe computers (ප්‍රධානරාමුවේ පරිගණක)
3. Mini computers (කුඩා පරිගණක)
4. Micro computers (ක්ෂුද්‍ර පරිගණක)

Super computers. / සුපිරි පරිගණක.

- Fastest and most powerful computers
  වේගවත්ම සහ බලවත්ම පරිගණක වේ.
- Memory capacity is very high
  මතක ධාරිතාව ඉතා ඉහළයි.
- Does parallel processing
  සමාන්තර සැකසුම් සිදු කරයි.
- Mainly focused on solving problems which are limited by calculation speed.
  ප්‍රධාන වශයෙන් අවධානය යොමු කර ඇත්තේ ගණනය කිරීමේ වේගයෙන් සීමා වූ ගැටළු විසඳීමට ය.

Mainframe computers. / ප්‍රධානරාමුවේ පරිගණක.

- Have large storage/ memory capacity
  විශාල ගබඩා / මතක ධාරිතාවක් ඇත.
- Used for scientific/ commercial processing
  විද්‍යාත්මක / වාණිජ සැකසුම් සඳහා භාවිතා වේ.
- Mainly focused on handling huge number of input, output and storage data at once.
  ප්‍රධාන වශයෙන් අවධානය යොමු කර ඇත්තේ ආදාන, ප්‍රතිදාන සහ ගබඩා දත්ත විශාල සංඛ්‍යාවක් එකවර හැසිරවීමටයි.
- Used in institutions like banks.
  බැංකු වැනි ආයතන වල භාවිතා වේ.

2. Which of the following is correct about CPU ?
CPU සම්බන්ධයෙන් පහත සඳහන් කුමක් නිවැරදිද?

A. Southbridge chipset helps to carry signals among the main devices.
Southbridge චිප්සෙට් ප්‍රධාන උපාංග අතර සංඥා රැගෙන යාමට උපකාරී වේ.

B. Pin Grid Array (PGA) contains small holes, and pins on the CPU can be inserted into the holes.
Pin Grid Array (PGA) හි කුඩා සිදුරු ඇති අතර CPU මත ඇති අල්පෙනෙති (pins) එම සිදුරුවලට ඇතුල් කළ හැක.

C. Northbridge chipset manages the basics of input/output such as USB, Audio and IDE.
Northbridge චිප්සෙට් USB Audio සහ IDE වැනි ආදාන හා ප්‍රතිදානයේ මූලික කරුණු කළමනාකරණය කරයි.

1) A only. A පමණි.
2) B only. B පමණි.
3) C only. C පමණි.
4) A and B only. A හා B පමණි.
5) All of the above. ඉහත සියල්ලම.

Answer:4)

While the Southbridge chipset does handle communication with slower peripheral devices, it carries signals among the main devices, such as the CPU and RAM. So, the statement A is correct.
Southbridge චිප්සෙට් එක මන්දගාමී පර්යන්ත උපාංග සමඟ සන්නිවේදනය හසුරුවන අතර, එය CPU සහ RAM වැනි ප්‍රධාන උපාංග අතර සංඥා දරයි. එබැවින් A ප්‍රකාශනය නිවැරදි ය.

Pin Grid Array (PGA) is a type of CPU packaging where the CPU has pins that are inserted into holes in the socket on the motherboard. The pins make contact with electrical contacts in the socket, facilitating communication between the CPU and other components.
Pin Grid Array (PGA) යනු CPU ඇසුරුම් වර්ගයකි, එහිදී CPU හි මවු පුවරුවේ සොකට් වල සිදුරුවලට ඇතුල් කරන අල්පෙනෙති ඇත. මෙම අල්පෙනෙති මගින් සොකට් එකෙහි ඇති විද්‍යුත් සම්බන්ධතා සමඟ සම්බන්ධතා ඇති කරයි, CPU සහ අනෙකුත් සංරචක අතර සන්නිවේදනය පහසු කරයි.

The Northbridge chipset handles communication between the CPU and high-speed components such as RAM and the GPU.
Northbridge චිප්සෙට් CPU සහ RAM, GPU වැනි අධිවේගී උපාංග අතර සන්නිවේදනය හසුරුවයි.

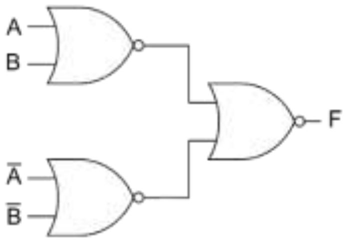
3. 002768.00751
The most significant digit and the least significant digit of the above decimal conversion respectively,
ඉහත සඳහන් දශමය සංඛ්‍යාවේ වැඩිම වෙසෙසි බිටුව හා අඩුම වෙසෙසි බිටුව පිළිවෙලින්,

1) 0,0 2) 2,1 3) 2,7 4) 8,1 5) 8,7

Answer :2)

The most significant digit is the transition with the maximum positional value.
වැඩිම වෙසෙසි බිටුව යනු වැඩිම ස්ථානීය වටිනාකමක් සහිත සංඛ්‍යාව වේ.
The least significant digit is the transition with the lowest positional value.
අඩුම වෙසෙසි බිටුව යනු අඩුම ස්ථානීය වටිනාකමක් සහිත සංඛ්‍යාව වේ.

4. What is the simplified output of the following logic circuit? (F - Output)
පහත දැක්වෙන තාර්කික පරිපථයේ සුළු කරන ලද ප්‍රතිදානය කුමක්ද? (F - ප්‍රතිදානය)



1) $(\overline{A}.B) + (A.\overline{B})$
2) $(\overline{A}.B).(\overline{A}.B)$
3) $(\overline{A}.\overline{B}) + (A.\overline{B})$
4) $(\overline{A}+\overline{B}).(A+\overline{B})$
5) $(\overline{A}+\overline{B}) + (\overline{A}+\overline{B})$

Answer:1)

We can get the Boolean expression of the above logic circuit as follows
ඉහත තාර්කික පරිපථයේ බූලීය ප්‍රකාශනය අපට පහත පරිදි ලබා ගත හැක

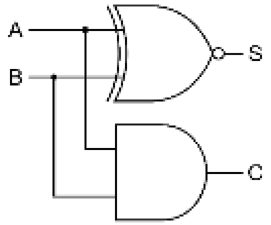
$$F = \overline{(\overline{A+B}) + (\overline{\overline{A}+\overline{B}})}$$

Simplifying the boolean expression obtained above:
ඉහත ලබාගත් බූලීය ප්‍රකාශනය සුළු කිරීම:

$$F = \overline{\overline{(A + B)} + \overline{(\overline{A} + \overline{B})}}$$

$\overline{\overline{(A + B)}}.\overline{\overline{(\overline{A} + \overline{B})}}$	De Morgans law ද මුවාර් ප්‍රමේයය
$(A + B).(\overline{A} + \overline{B})$	Double Inverse law ද්විත්ව ප්‍රතිලෝම න්‍යාය
$A\overline{A} + A\overline{B} + \overline{A}B + B\overline{B}$	Distributive law විඝටන න්‍යාය
$0 + A\overline{B} + \overline{A}B + 0$	Inverse law ප්‍රතිලෝම න්‍යාය
$A\overline{B} + \overline{A}B$	Identity law සර්වසාම්‍ය න්‍යාය

5. A student examines the output of the given circuit where A=1 at all the time .Then he gets S=0 as an output. Then he removes the AND gate and replaces it with a NOR GATE. What is the output?
සැමවිටම A=1 වන පරිපථයක , සිසුවෙකු ප්‍රතිදානය පරීක්ෂා කරයි. එවිට ඔහුට S=0 ලෙස ලැබුණි. ඉන්පසුව ඔහු එහි AND ද්වාරය ගලවා ඉවත්කර ඒ වෙනුවට NOR ද්වාරයක් යොදා පරීක්ෂා කරයි. එවිට ලැබෙන B ආදානය හා C ප්‍රතිදානය සොයන්න.



1) B= 1 and C= 0
2) B= 0 and C= 1
3) B=1 and C= 1
4) B=0 and C=0
5) None of above. / ඉහත කිසිවක් නොවේ.

Answer :4)

Truth table of XNOR
XNOR හි සත්‍යතා වගුව

ICN	A	B	S
1	0	0	1
2	0	1	0
3	1	0	0
4	1	1	1

We overlook the ICN 1 and 2 because we need only when the output of A is 1.
අපට අවශ්‍ය වන්නේ A හි අගය සැම විටම 1ව පවතින අවස්ථාව පමණක් නිසා අප ICN 1 හා 2 අවස්ථා නොසලකා හරිමු.

Values that can be use.
භාවිතා කළ හැකි අගයන්.

ICN	A	B	S
3	1	0	0
4	1	1	1

S=0 ලෙස ලැබෙන බව ප්‍රශ්නයෙහි සටහන් කර ඇති නිසා ICN 4 අවස්ථාව නොසලකා හරින්න.
Overlook the case ICN 4 as it is noted in the question that S=0.

ICN	A	B	S
3	1	0	0
~~4~~	~~1~~	~~1~~	~~1~~

Now we get the value of B as 1. Since it has been said that a NOR gate is used instead of an AND gate, so let's use NOR logic.
දැන් B හි අගය අපට 0 ලෙස ලැබේ. AND ද්වාරය වෙනුවට NOR ද්වාරයක් භාවිත කරන බව පවසා ඇති නිසා NOR තර්කය භාවිත කරමු.

ICN	A	B	S
3	1	0	0

Accordingly, since B=0 and C=0, answer number 4 is correct.
ඒ අනුව B=0 හා C=0 නිසා පිළිතුරු අංක 4 නිවැරදි වේ.

6. Which of the following is a type of database management system (DBMS)?
පහත ඒවායින් දත්ත සමුදා කළමනාකරණ පද්ධති (DBMS) වර්ගය කුමක්ද?

1) MySQL
2) PostgreSQL
3) Oracle
4) All of the above. ඉහත සියල්ලම
5) None of the above ඉහත කිසිවක් නොවේ.

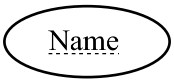
Answer : 4)

Explanation / පැහැදිලි කිරීම:

The correct answer is 4) All of the above because MySQL, PostgreSQL, and Oracle are all types of Database Management Systems (DBMS). Each of these systems is used to create, manage, and interact with databases, though they vary in features and capabilities. MySQL is an open-source relational database known for its simplicity and speed, commonly used in web applications. PostgreSQL is also open-source but is known for its advanced features and support for complex queries, making it popular in data-heavy environments. Oracle DBMS is a commercial, enterprise-grade system, known for its scalability, robustness, and support for large-scale applications.
නිවැරදි පිළිතුර වන්නේ 4) ඉහත සියල්ල MySQL, PostgreSQL සහ Oracle සියලු වර්ගවල දත්ත සමුදා කළමනාකරණ පද්ධති DBMS වන බැවිනි. මෙම සෑම පද්ධතියක්ම විශේෂාංග සහ හැකියාවන්ගෙන් වෙනස් වුවද, දත්ත සමුදායන් නිර්මාණය කිරීමට, කළමනාකරණය කිරීමට සහ අන්තර් ක්‍රියා කිරීමට භාවිතා කරයි. MySQL යනු වෙබ් යෙදුම්වල බහුලව භාවිතා වන එහි සරල බව සහ වේගය සඳහා ප්‍රසිද්ධ විවෘත මූලාශ්‍ර සම්බන්ධතා දත්ත සමුදායකි. PostgreSQL ද විවෘත මූලාශ්‍රයක් වන නමුත් එහි උසස් විශේෂාංග සහ සංකීර්ණ විමසුම් සඳහා සහාය සඳහා ප්‍රසිද්ධය, එය දත්ත බර පරිසරයන් තුළ ජනප්‍රිය කරයි. Oracle DBMS යනු වාණිජ, ව්‍යවසාය-ශ්‍රේණිගත පද්ධතියකි, එහි පරිමාණය, ශක්තිමත් බව සහ මහා පරිමාණ යෙදුම් සඳහා සහය සඳහා ප්‍රසිද්ධය.

The other options are incorrect individually, as each represents only one type of DBMS. Option 5, "None of the above," is incorrect because all three listed systems are well-established types of DBMS. Each ystem listed serves the same primary function managing databases but with different strengths, suited to various project requirements. Therefore, the correct answer is that all the given options are types of DBMS.
එක් එක් DBMS වර්ගයක් පමණක් නියෝජනය කරන බැවින් අනෙක් විකල්ප තනි තනිව වැරදිය. විකල්ප 5, ,ඉහත කිසිවක් නොවේ,, ලැයිස්තුගත පද්ධති තුනම හොඳින් ස්ථාපිත DBMS වර්ග බැවින් වැරදිය. ලැයිස්තුගත කර ඇති සෑම පද්ධතියක්ම එකම මූලික කාර්යයක් වන දත්ත සමුදා කළමනාකරණය කරයි, නමුත් විවිධ ශක්තීන් සමඟ, විවිධ ව්‍යාපෘති අවශ්‍යතාවලට ගැළපේ. එබැවින්, නිවැරදි පිළිතුර නම් ලබා දී ඇති සියලුම විකල්ප DBMS වර්ග වේ යන්නයි.

7. What type of key is depicted by the following diagram?
පහත රූප සටහනෙන් දැක්වෙන්නේ කුමන ආකාරයේ යතුරක්ද?



1) Alternate key විකල්ප යතුර 2) Partial key අර්ධ යතුර 3) Derived key ව්‍යුත්පන්න යතුර 4) Composite key සංයුක්ත යතුර 5) Candidate key අපේක්ෂක යතුර

Answer : 2)

In Entity-Relationship (ER) diagrams, partial keys are depicted using a dashed underline, instead of the solid underline used for primary keys. Therefore, the answer is 2.
භූතාර්ථ-සම්බන්ධතා (ER) සටහන්වල, ප්‍රාථමික යතුරු සඳහා භාවිතා කරන ඝන යටි ඉරි වෙනුවට, කඩ ඉරි සහිත යටි ඉරක් භාවිතයෙන් අර්ධ යතුරු නිරූපණය කෙරේ. එබැවින් පිළිතුර 2 වේ.

A partial key identifies a record in a database table, but only uniquely within the context of another related entity. Partial keys are used in database design for weak entities. A weak entity is dependent on another, stronger entity for its existence.
අර්ධ යතුරක් දත්ත සමුදා වගුවක වාර්තාවක් හඳුනා ගනී, නමුත් වෙනත් සම්බන්ධිත භූතාර්ථයක සන්දර්භය තුළ පමණක් අනන්‍ය වේ. දුර්වල භූතාර්ථ සඳහා දත්ත සමුදාය නිර්මාණය කිරීමේදී අර්ධ යතුරු භාවිතා වේ. දුර්වල භූතාර්ථයක් එහි පැවැත්ම සඳහා තවත් ශක්තිමත් භූතාර්ථයක් මත රඳා පවතී.

8. Which of the following is the most efficient way to reverse a string in Python?
පයිතන් හි string එකක් ප්‍රතිවර්තනය කිරීමේ වඩාත් කාර්යක්ෂම ක්‍රමය පහත ඒවායින් කුමක්ද?

1) Using a loop to iterate through the string and append characters to a new string in reverse order.
string එක හරහා පුනරාවර්තනය කිරීමට සහ ප්‍රතිලෝම අනුපිළිවෙලට නව string එකකට අක්ෂර එකතු කිරීමට ලූපයක් භාවිතා කිරීම.
2) Using the reversed() function and converting the result to a string.
reversed() ශ්‍රිතය භාවිතා කිරීම සහ ප්‍රතිඵලය string එකකට පරිවර්තනය කිරීම.
3) <u>Using slice notation with a step of -1.</u>
-1 step අගයක් සහිත slice අංකනය භාවිතා කිරීම.
4) Using the join() method with a reversed list of characters.
ප්‍රතිලෝම අක්ෂර ලැයිස්තුවක් සමඟ join() ක්‍රමය භාවිතා කිරීම.
5) Using a recursive function. / ආවර්තික ශ්‍රිතයක් භාවිතා කිරීම.

Answer : 3)

Here's a brief explanation of each option
මෙන්න එක් එක් විකල්පය පිළිබඳ කෙටි පැහැදිලි කිරීමක්

- **Option / විකල්ප 1)**

This method involves manually iterating through the string and constructing a new string character by character. While straightforward, it is inefficient due to the repeated creation of new strings, which can lead to increased time complexity, especially for longer strings.
මෙම ක්‍රමයට string එක හරහා අතින් පුනරාවර්තනය කිරීම සහ අක්ෂරයෙන් නව string අක්ෂරයක් ගොඩනැගීම ඇතුළත් වේ. සරල වුවත්, නව string නැවත නැවත නිර්මාණය කිරීම හේතුවෙන් එය අකාර්යක්ෂම වේ, එය කාල සංකීර්ණතාව වැඩි කිරීමට හේතු විය හැක, විශේෂයෙන් දිගු string සඳහා.

- **Option / විකල්ප 2)**

The reversed() function creates an iterator that yields characters from the original string in reverse order. However, it requires an additional step to convert this iterator back into a string using .join(), making it slightly less efficient than direct slicing.
reversed() ශ්‍රිතය මඟින් ප්‍රතිලෝම අනුපිළිවෙලින් මුල් string එකේන් අක්ෂර ලබා දෙන පුනරාවර්තකයක් නිර්මාණය කරයි. කෙසේ වෙතත්, .join() භාවිතයෙන් මෙම පුනරාවර්තකය නැවත තන්තුවක් බවට පරිවර්තනය කිරීමට අමතර පියවරක් අවශ්‍ය වේ, එය සෘජු slicing වඩා තරමක් අඩු කාර්යක්ෂම වේ.

- **Option / විකල්ප 3)**

This is the most efficient method, as it allows for reversing a string in a single line of code (reversed_string = original_string[::-1]). It leverages Python's built-in capabilities and is both concise and fast, making it the preferred approach for reversing strings.
මෙය වඩාත් කාර්යක්ෂම ක්‍රමය වන අතර, එය තනි කේත පේළියක (reversed_string = original_string[::-1]) තන්තුවක් ආපසු හැරවීමට ඉඩ සලසයි. එය පයිතන් හි ගොඩනඟන ලද හැකියාවන් උපයෝගී කර ගන්නා අතර එය සංක්ෂිප්ත සහ වේගවත් වන අතර, එය නූල් ප්‍රතිවර්තනය කිරීම සඳහා වඩාත් කැමති ප්‍රවේශය බවට පත් කරයි.

- **Option / විකල්ප 4)**

This method involves first converting the string into a list of characters, reversing that list, and then joining it back into a string. While effective, it is less efficient than slicing because it involves multiple operations (conversion to list, reversal, and joining).
මෙම ක්‍රමයට මුලින්ම string එක අක්ෂර ලැයිස්තුවක් බවට පරිවර්තනය කිරීම, එම ලැයිස්තුව ආපසු හැරවීම සහ එය නැවත string එකක් බවට සම්බන්ධ කිරීම ඇතුළත් වේ. ඵලදායි වන අතර, එය බහු මෙහෙයුම් (ලැයිස්තුවට පරිවර්තනය කිරීම, ආපසු හැරවීම සහ සම්බන්ධ වීම) ඇතුළත් වන නිසා එය පෙති කැපීමට වඩා අඩු කාර්යක්ෂම වේ.

- **Option / විකල්ප 5)**

This method defines a function that calls itself to reverse the string. While it can work for small strings, recursion can lead to high memory usage and stack overflow errors for larger strings due to the depth of recursive calls, making it inefficient compared to other methods.
මෙම ක්‍රමය මඟින් string ප්‍රතිලෝම කිරීමට තමන්ව කැඳවන ශ්‍රිතයක් නිර්වචනය කරයි. එය කුඩා string සඳහා ක්‍රියා කළ හැකි අතර, ප්‍රත්‍යාවර්තනය ඉහළ මතක භාවිතයට හේතු විය හැකි අතර ප්‍රත්‍යාවර්ති ඇමතුම්වල ගැඹුර හේතුවෙන් විශාල string සඳහා overflow දෝෂ ඇති විය හැකි අතර එය අනෙකුත් ක්‍රමවලට සාපේක්ෂව අකාර්යක්ෂම වේ.

Therefore the correct answer is 3).
එබැවින් නිවැරදි පිළිතුර 3) වේ.

9. What is the purpose of the **break** statement in a loop, particularly when used in conjunction with nested loops or control flow mechanisms?
විශේෂයෙන්ම nested ලූප හෝ පාලන ගැලීම් යාන්ත්‍රණයන් සමඟ ඒකාබද්ධව භාවිතා කරන විට, ලූපයක **break** ප්‍රකාශයේ අරමුණ කුමක්ද?

1) To exit the current iteration and move to the next iteration of the loop.
වත්මන් පුනරාවර්තනයෙන් ඉවත් වී ලූපයේ ඊළඟ පුනරාවර්තනය වෙත යාමට.

2) To terminate the entire loop, including outer loops in nested structures.
nested ව්‍යුහවල පිටත ලූප ඇතුළුව සම්පූර්ණ ලූපය අවසන් කිරීමට.

3) <u>To exit innermost loop in which break statement is encountered, and resume execution after that loop.</u>
break ප්‍රකාශය හමු වූ අභ්‍යන්තරම ලූපයෙන් පිටවීමට සහ එම ලූපයෙන් පසුව ක්‍රියාත්මක කිරීම නැවත ආරම්භ කිරීමට.

4) To raise a custom exception and halt program execution.
සම්මත විශේෂ්‍ය වාර්තා මතු කිරීම සහ වැඩසටහන් ක්‍රියාත්මක කිරීම නැවැත්වීම.

5) To temporarily suspend loop execution and resume at a specific checkpoint.
ලූපය ක්‍රියාත්මක කිරීම තාවකාලිකව අත්හිටුවීමට සහ නිශ්චිත ස්ථානයකින් නැවත ආරම්භ කිරීමට.

Answer : 3)

Explanation / පැහැදිලි කිරීම

The purpose of the break statement in a loop is to exit the innermost loop in which it is encountered, immediately stopping further iterations of that loop.
ලූපයක break ප්‍රකාශයේ පරමාර්ථය වන්නේ එය හමුවන අභ්‍යන්තරම ලූපයෙන් පිටවීම, එම ලූපයේ වැඩිදුර පුනරාවර්තනය වහාම නතර කිරීමයි.

When used in conjunction with nested loops, the break statement only affects the innermost loop, allowing the program to resume execution after the block of that loop, while outer loops remain unaffected and continue executing.
nested ලූප සමඟ ඒකාබද්ධව භාවිතා කරන විට, break ප්‍රකාශය බලපාන්නේ අභ්‍යන්තර ලූපයට පමණක් වන අතර, එම ලූපය අවහිර කිරීමෙන් පසුව ක්‍රමලේඛයට ක්‍රියාත්මක කිරීම නැවත ආරම්භ කිරීමට ඉඩ සලසයි, නමුත් පිටත ලූපයන් බලපෑමට ලක් නොවී දිගටම ක්‍රියාත්මක වේ.

This makes it particularly useful for control flow mechanisms where specific conditions require premature termination of a loop.
නිශ්චිත කොන්දේසි යටතේ ලූපයක් අකාලයේ අවසන් කිරීම අවශ්‍ය වන පාලන ගැලීම් යාන්ත්‍රණ සඳහා මෙය විශේෂයෙන් ප්‍රයෝජනවත් වේ.

Contrary to some options, it does not raise exceptions or suspend execution to resume later. Therefore, the correct option is 3) To exit the innermost loop in which the break statement is encountered, and resume execution after that loop.
සමහර විකල්ප වලට පටහැනිව, එය විශේෂ්‍ය වාර්ථා මතු නොකරයි හෝ පසුව නැවත ආරම්භ කිරීම සඳහා ක්‍රියාත්මක කිරීම අත්හිටුවන්නේ නැත. එබැවින්, නිවැරදි විකල්පය වන්නේ 3) break ප්‍රකාශය හමු වූ අභ්‍යන්තරම ලූපයෙන් පිටවීම සහ එම ලූපයෙන් පසුව ක්‍රියාත්මක කිරීම නැවත ආරම්භ කිරීමයි.

Therefore, the correct answer is 3).
එබැවින් නිවැරදි පිළිතුර 3) වේ.

10. What is the difference between a list and a tuple in Python?
පයිතන් හි ලැයිස්තුවක් සහ ටපල් එකක් අතර වෙනස කුමක්ද?

1) Lists are immutable, while tuples are mutable.
ලැයිස්තු වෙනස් කළ නොහැකි අතර ටපල් විකෘති වේ.

2) Lists are mutable, while tuples are immutable.
ලැයිස්තු විකෘති වන අතර ටපල් වෙනස් කළ නොහැක.

3) Lists can have duplicate elements, while tuples cannot.
ලැයිස්තු වල අනුපිටපත් මූලද්‍රව්‍ය තිබිය හැකි අතර ටපල් වලට නොහැක.

4) Tuples can have duplicate elements, while lists cannot.
ටපල්වලට අනුපිටපත් මූලද්‍රව්‍ය තිබිය හැකි අතර, ලැයිස්තුවලට නොහැක.

5) There is no difference between lists and tuples.
ලිස්ට් සහ ටපල් අතර වෙනසක් නැත.

Answer: 2)

According to Python rules, a list is mutable, which means its elements can be changed, added, or removed after it is created. Because of this mutability, lists are used when you need data that can be updated during the program. A tuple is the opposite it is immutable, meaning once a tuple is created, you cannot modify its values. This immutability makes tuples safer and faster for fixed collections of data.
පයිතන් රීති වලට අනුව, ලැයිස්තුවක් වෙනස් කළ හැකි ය, එනම් එය නිර්මාණය කිරීමෙන් පසු එහි මූලද්‍රව්‍ය වෙනස් කිරීමට, එකතු කිරීමට හෝ ඉවත් කිරීමට හැකිය. මෙම විකෘතිතාවය නිසා, වැඩසටහන අතරතුර යාවත්කාලීන කළ හැකි දත්ත අවශ්‍ය වූ විට ලැයිස්තු භාවිතා කරනු ලැබේ. ටපල් එකක එය ප්‍රතිවිරුද්ධයයි එය වෙනස් කළ නොහැකි ය, එනම් ටපල් එකක් නිර්මාණය කළ පසු, ඔබට එහි අගයන් වෙනස් කළ නොහැක. මෙම වෙනස් කළ නොහැකි බව ස්ථාවර දත්ත එකතු කිරීම් සඳහා ටපල් ආරක්ෂිත සහ වේගවත් කරයි.

Therefore, the statement "Lists are mutable, while tuples are immutable" correctly describes the difference between them. This is why Option 2 is the true statement. The other options are incorrect because they either swap the mutability or talk about duplicates, which both lists and tuples can have. So the only accurate and standard difference is given in Option 2.
එබැවින්, "ලැයිස්තු වෙනස් කළ හැකි වේ, නමුත් ටපල් වෙනස් කළ නොහැකම යන ප්‍රකාශය ඒවා අතර වෙනස නිවැරදිව විස්තර කරයි. මේ නිසා විකල්ප 2 සත්‍ය ප්‍රකාශයයි. අනෙක් විකල්ප වැරදියි, මන්ද ඒවා ලැයිස්තු සහ ටපල් දෙකටම තිබිය හැකි විකෘතිතාව හුවමාරු කර ගැනීම හෝ අනුපිටපත් ගැන කතා කිරීම නිසාය. එබැවින් එකම නිවැරදි සහ සම්මත වෙනස විකල්ප 2 හි දක්වා ඇත.