සියලු ම හිමිකම් ඇවිරිණි /All Rights Reserved]

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 404 MARKING

2026/01/18
ictfromabc
Ravindu Bandaranayake

1. What was a major impact of Herman Hollerith’s punch card system on the 1890 U.S. Census?
   1890 එක්සත් ජනපද සංගණනයට හර්මන් හොලරිත්ගේ සිදුරු පත් ක්‍රමයේ ප්‍රධාන බලපෑම කුමක්ද?

   1) It drastically reduced the time required to process census data from years to months.
      එය වසරවල සිට මාස දක්වා සංගණන දත්ත සැකසීමට ගතවන කාලය විශාල ලෙස අඩු කළේය.
   2) It introduced the use of electronic computers for the first time in census data processing.
      එය සංගණන දත්ත සැකසීමේදී ප්‍රථම වරට ඉලෙක්ට්‍රොනික පරිගණක භාවිතය හඳුන්වා දුන්නේය.
   3) It allowed for manual counting of individual census forms by census workers.
      එය සංගණන සේවකයන් විසින් තනි පුද්ගල සංගණන ආකෘති අත්යුරුව ගණන් කිරීමට ඉඩ ලබා දුන්නේය.
   4) It eliminated the need for data collection and only involved data analysis.
      එය දත්ත රැස්කිරීමේ අවශ්‍යතාවය ඉවත් කළ අතර දත්ත විශ්ලේෂණයට පමණක් සම්බන්ධ විය.
   5) It provided a way to store census data in digital formats for the first time.
      එය ප්‍රථම වරට සංගණන දත්ත සංඛ්‍යාංක ආකෘතිවලින් ගබඩා කිරීමට ක්‍රමයක් ලබා දුන්නේය.

Answer : 1)

The major impact of Herman Hollerith’s punch card system on the 1890 U.S. Census was that it drastically reduced the time required to process census data from years to months.
1890 U.S. සංගණනයේදී හර්මන් හොලරිත්ගේ පන්ච් කාඩ් ක්‍රමයේ ප්‍රධානතම බලපෑම වූයේ සංගණන දත්ත වසරින් මාස ගණනකට සැකසීමට අවශ්‍ය කාලය විශාල ලෙස අඩු කිරීමයි.

Before Hollerith's system, census data was processed manually, which took many years to complete. However, Hollerith introduced a punch card system that automated the counting and sorting of data using machines, speeding up the process significantly. The punch cards could store data in the form of holes, which were read by machines, making the data processing much faster and more efficient. As a result, the 1890 census data, which previously took around 8 years to process, was completed in just a few months, drastically improving efficiency. This makes Option 1) the correct answer.
හොලරිත්ගේ පද්ධතියට පෙර, සංගණන දත්ත අතින් සකස් කරන ලද අතර, එය සම්පූර්ණ කිරීමට වසර ගණනාවක් ගත විය. කෙසේ වෙතත්, හොලරිත් යන්ත්‍ර භාවිතයෙන් දත්ත ගණන් කිරීම සහ වර්ග කිරීම ස්වයංක්‍රිය කරන ලද පන්ච් කාඩ් පද්ධතියක් හඳුන්වා දුන් අතර, ක්‍රියාවලිය සැලකිය යුතු ලෙස වේගවත් කරයි. පන්ච් කාඩ්පත් යන්ත්‍ර මගින් කියවන ලද සිදුරු ආකාරයෙන් දත්ත ගබඩා කළ හැකි අතර, දත්ත සැකසීම වඩා වේගවත් හා කාර්යක්ෂම කරයි. එහි ප්‍රතිඵලයක් වශයෙන්, 1890 සංගණන දත්ත, කලින් සැකසීමට වසර 8ක් පමණ ගත වූ අතර, මාස කිහිපයකින්, කාර්යඬමතාව විශාල ලෙස වැඩිදියුණු විය. මෙය විකල්පය 1) නිවැරදි පිළිතුර බවට පත් කරයි.

2. Which of the following statements about Unicode is correct?
Unicode පිළිබඳ පහත සඳහන් කුමන ප්‍රකාශය නිවැරදිද?

1) Supports only the English language.
ඉංග්‍රීසි භාෂාවට පමණක් සහය දක්වයි.

2) Unicode is only used for text representation in web pages.
යුනිකෝඩ් භාවිතා කරන්නේ වෙබ් පිටු වල අකුරු නිරූපණය සඳහා පමණි.

3) Characters are limited to a maximum of 256.
අක්ෂර උපරිම 256 කට සීමා වේ.

4) Uses a 16-bit encoding scheme.
බිටු 16 ක කේතන ක්‍රමයක් භාවිතා කරයි.

5) Assigns a unique number to every character, regardless of platform.
වේදිකාව කුමක් වුවත්, සෑම සංකේතයකටම අනන්‍ය අංකයක් පවරයි.

Answer : 5)

Explanation / පැහැදිලි කිරීම

The correct statement about Unicode is that it "assigns a unique number to every character, regardless of platform." Unicode is a standardized encoding system developed to ensure that every character from nearly every language and symbol set worldwide can be represented in computers and digital systems. Unlike older encoding schemes, Unicode covers a vast range of languages, scripts, and symbols by assigning each character a unique numeric code, called a "code point," which remains the same across different devices, platforms, and programs.
යුනිකෝඩ් පිළිබඳ නිවැරදි ප්‍රකාශය නම් එය වේදිකාව කුමක් වුවත්, සෑම අක්ෂරයකටම අනන්‍ය අංකයක් පවරයි යන්නයි. යුනිකෝඩ් යනු ලොව පුරා පිහිටුවා ඇති සෑම භාෂාවකින්ම සහ සංකේතයකින්ම සෑම අක්ෂරයක්ම පරිගණක සහ ඩිජිටල් පද්ධති තුළ නිරූපණය කළ හැකි බව සහතික කිරීම සඳහා සකස් කරන ලද ප්‍රමිතිගත කේතීකරණ පද්ධතියකි. පැරණි කේතීකරණ ක්‍රම මෙන් නොව, යුනිකෝඩ් විවිධ උපාංග, වේදිකා සහ වැඩසටහන් හරහා එකම ලෙස පවතින කේත ලක්ෂ්‍යයක් ලෙස හැඳින්වෙන, එක් එක් අක්ෂරයකට අනන්‍ය සංඛ්‍යාත්මක කේතයක් ලබා දීමෙන් විශාල පරාසයක භාෂා, ස්ක්‍රිප්ට් සහ සංකේත ආවරණය කරයි.

This universal approach allows text to be accurately displayed and understood no matter the operating system, application, or language being used. It overcomes the limitations of previous encoding systems that could only support specific languages or character sets, ensuring that global digital communication remains consistent and accessible.
මෙම විශ්වීය ප්‍රවේශය මඟින් මෙහෙයුම් පද්ධතිය, යෙදුම හෝ භාෂාව භාවිතා කළත් අකුරු නිවැරදිව ප්‍රදර්ශනය කිරීමට සහ තේරුම් ගැනීමට ඉඩ සලසයි. ගෝලීය ඩිජිටල් සන්නිවේදනය ස්ථාවර සහ ප්‍රවේශ විය හැකි බව සහතික කරමින්, විශේෂිත භාෂා හෝ අක්ෂර කට්ටල සඳහා පමණක් සහය විය හැකි පෙර කේතීකරණ පද්ධතිවල සීමාවන් එය ජය ගනී.

Therefore, the correct answer is option 5).
එබැවින්, නිවැරදි පිළිතුර විකල්ප 5) වේ.

3. Consider the following statements regarding combinational and sequential logic circuits.
සංයෝජන සහ අනුක්‍රමික තාර්කික පරිපථ සම්බන්ධයෙන් පහත ප්‍රකාශයන් සලකා බලන්න.

A. Sequential circuits are mainly used for arithmetic and Boolean operations.
අනුක්‍රමික පරිපථ ප්‍රධාන වශයෙන් අංක ගණිතමය සහ බූලියානු මෙහෙයුම් සඳහා භාවිතා වේ.

B. Combinational circuits are faster and better in performance than sequential circuits.
අනුක්‍රමික පරිපථවලට වඩා සංයෝජන පරිපථ වේගවත් සහ කාර්ය සාධනය වඩා හොඳය.

C. Combinational circuits are less complex than sequential circuit.
සංයුක්ත පරිපථ අනුක්‍රමික පරිපථවලට වඩා සංකීර්ණත්වයෙන් අඩු වේ.

Which of the above statements is/are True?
ඉහත ප්‍රකාශවලින් සත්‍ය මොනවාද?

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

Answer: 3)

Explanation of statements / ප්‍රකාශයන් පැහැදිලි කිරීම

Statement / ප්‍රකාශය A

This is incorrect. Sequential circuits are primarily used for storing data, managing states, and performing time-dependent operations. Arithmetic and Boolean operations are typically the domain of combinational circuits, not sequential circuits.
මෙය වැරදියි. අනුක්‍රමික පරිපථ මූලික වශයෙන් දත්ත ගබඩා කිරීම, තත්වයන් කළමනාකරණය කිරීම සහ කාලය මත යැපෙන මෙහෙයුම් සිදු කිරීම සඳහා භාවිතා වේ. අංක ගණිතය සහ බූලියන් මෙහෙයුම් සාමාන්‍යයෙන් සංයෝජන පරිපථවල වසම මිස අනුක්‍රමික පරිපථ නොවේ.

Statement / ප්‍රකාශය B

This is correct. Combinational circuits do not depend on previous states and have no memory element, making them inherently faster. Sequential circuits rely on clock cycles and are thus slower in comparison.
මෙය නිවැරදියි. සංයෝජන පරිපථ පෙර පැවති තත්වයන් මත රඳා නොපවතින අතර මතක මූලද්‍රව්‍යයක් නොමැති අතර ඒවා නෛසර්ගිකව වේගවත් කරයි. අනුක්‍රමික පරිපථ ඔරලෝසු චක්‍ර මත රඳා පවතින අතර සංසන්දනය කිරීමේදී මන්දගාමී වේ.

Statement / ප්‍රකාශය C

This is correct. Combinational circuits do not include memory elements, and their design is generally simpler compared to sequential circuits, which require additional components like flip-flops for state retention.
මෙය නිවැරදියි. සංයුක්ත පරිපථවල මතක මූලද්‍රව්‍ය ඇතුළත් නොවන අතර, ඒවායේ සැලසුම සාමාන්‍යයෙන් අනුක්‍රමික පරිපථවලට සාපේක්ෂව සරල වන අතර, තත්වය රඳවා තබා ගැනීම සඳහා flip-flops වැනි අමතර සංරචක අවශ්‍ය වේ.

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

4. Normalization in database design helps to,
දත්ත සමුදා නිර්මාණයේ ප්‍රමතකරණය උපකාරී වන්නේ,

1) Increase redundancy in data / දත්තවල අතිරික්තය වැඩි කිරීම.
2) Reduce data integrity / දත්ත අඛණ්ඩතාව අඩු කිරීම.
3) Organize data into efficient structures / කාර්යක්ෂම ව්‍යුහයන් බවට දත්ත සංවිධානය කිරීමට.
4) Speed up database queries / දත්ත සමුදා විමසුම් වේගවත් කිරීමට.
5) None of the above / ඉහත කිසිවක් නොවේ.

Answer : 3)

Normalization in database design helps to organize data into efficient structures. By reducing redundancy and ensuring data integrity, it helps create a well-structured database that minimizes repetition and ensures that data is stored logically. This improves the overall organization of the database, making it easier to manage and maintain.
දත්ත සමුදා නිර්මාණයේ ප්‍රමතකරණය දත්ත කාර්යක්ෂම ව්‍යුහයන් බවට සංවිධානය කිරීමට උපකාරී වේ. අතිරික්තය අඩු කිරීමෙන් සහ දත්ත අඛණ්ඩතාව සහතික කිරීමෙන්, එය පුනරාවර්තනය අවම කරන සහ දත්ත තාර්කිකව ගබඩා කර ඇති බව සහතික කරන හොඳින් ව්‍යුහගත දත්ත සමුදායක් නිර්මාණය කිරීමට උපකාරී වේ. මෙය දත්ත සමුදායේ සමස්ත සංවිධානය වැඩි දියුණු කරයි, කළමනාකරණය සහ නඩත්තු කිරීම පහසු කරයි.

The other options are incorrect for the following reasons: Option 1 is incorrect because normalization reduces redundancy, not increases it. Option 2 is wrong because normalization improves data integrity by eliminating anomalies. Option 4 is not accurate because normalization focuses on structuring data, not directly speeding up queries, though it can improve query efficiency in the long run.
පහත සඳහන් හේතූන් මත අනෙකුත් විකල්ප වැරදි වේ: විකල්ප 1 වැරදියි මන්ද ප්‍රමතකරණය අතිරික්තය අඩු කරයි, එය වැඩි නොකරයි. ප්‍රමතකරණය විෂමතා ඉවත් කිරීමෙන් දත්ත අඛණ්ඩතාව වැඩි දියුණු කරන නිසා විකල්ප 2 වැරදියි. ප්‍රමතකරණය දත්ත ව්‍යුහගත කිරීම කෙරෙහි අවධානය යොමු කරයි, විමසුම් සෘජුවම වේගවත් නොකරන බැවින් විකල්ප 4 නිවැරදි නොවේ, නමුත් එය දිගුකාලීනව විමසුම් කාර්යක්ෂමතාව වැඩි දියුණු කළ හැකිය.

5. Which of the following SQL commands can be used for unspecified value search patterns?
සඳහන් නොකළ අගය සෙවුම් රටා සඳහා භාවිතා කළ හැක්කේ පහත SQL විධානවලින් කවරක්ද?

1) IN 2) BETWEEN 3) OR 4) AND 5) LIKE

Answer : 5)

Explanation / පැහැදිලි කිරීම

Option / විකල්ප 1)
The IN operator is used to specify a list of possible values for a column. It allows you to match a column against a set of values.
තිරුවක් සඳහා විය හැකි අගයන් ලැයිස්තුවක් නියම කිරීමට IN ක්‍රියාකරු භාවිතා කරයි. අගයන් සමූහයකට එරෙහිව තිරුවක් ගැලපීමට එය ඔබට ඉඩ සලසයි.

Option / විකල්ප 2)
The BETWEEN operator is used to filter results within a specific range, inclusive of the boundaries (e.g., WHERE column BETWEEN value1 AND value2). It is not used for unspecified value search patterns but for range-based searches.
BETWEEN ක්‍රියාකරු මායිම් ඇතුළුව නිශ්චිත පරාසයක් තුළ ප්‍රතිඵල පෙරීමට භාවිතා කරයි (උදා. WHERE column BETWEEN value1 AND value2). එය නියම නොකළ අගය සෙවුම් රටා සඳහා නොව පරාසය මත පදනම් වූ සෙවුම් සඳහා භාවිතා වේ.

Option / විකල්ප 3)
The OR operator is used to combine multiple conditions where at least one condition must be true (e.g., WHERE column = value1 OR column = value2). It does not specify patterns but can be used to check for multiple exact values.
OR ක්‍රියාකරු අවම වශයෙන් එක් කොන්දේසියක්වත් සත්‍ය විය යුතු කොන්දේසි කිහිපයක් ඒකාබද්ධ කිරීමට භාවිතා කරයි (උදා: WHERE column = value1 OR column = value2). එය රටා සඳහන් නොකරන නමුත් නිවැරදි අගයන් කිහිපයක් සඳහා පරීක්ෂා කිරීමට භාවිතා කළ හැක.

Option / විකල්ප 4)
The AND operator is used to combine multiple conditions where all conditions must be true (e.g., WHERE column1 = value1 AND column2 = value2). It's used for exact matches, not for search patterns.
AND ක්‍රියාකරු සියලු කොන්දේසි සත්‍ය විය යුතු බහු කොන්දේසි ඒකාබද්ධ කිරීමට භාවිතා කරයි (උදා: WHERE column1 = value1 AND column2 = value2). එය සෙවුම් රටා සඳහා නොව නිශ්චිත ගැලපීම් සඳහා භාවිතා වේ.

Option / විකල්ප 5)
The LIKE operator is used for pattern matching, especially when the exact value is not known. It allows for the use of wildcards (% for multiple characters, _ for a single character) to match unspecified values (e.g., WHERE column LIKE 'value%').
LIKE ක්‍රියාකරු රටා ගැලපීම සඳහා භාවිතා කරයි, විශේෂයෙන් නිශ්චිත අගය නොදන්නා විට. එය නිශ්චය නොකළ අගයන් (උදා: WHERE column LIKE 'value%') ගැලපීම සඳහා (අනුලකුණු කිහිපයක් සඳහා % _ තනි අක්ෂරයක් සඳහා) Wildcards භාවිතයට ඉඩ සලසයි.

Therefore the Correct answer is 5)
එබැවින් නිවැරදි පිළිතුර 5)

6. Which rows are returned when you execute the following SQL query?
පහත SQL විමසුම ක්‍රියාත්මක කරන විට ආපසු ලබා දෙන පේළි මොනවාද?

```
SELECT *
FROM Products A
INNER JOIN Categories B
ON A.category_id = B.category_id;
```

1) All rows from both tables / වගු දෙකෙන්ම සියලුම පේළි.
2) Only rows with NULL in either table / එක් වගුවක NULL සහිත පේළි පමණි.
3) Only the rows where category_id matches in both tables.
   වගු දෙකෙහිම category_id ගැළපෙන පේළි පමණි.
4) All rows from the Products table and NULLs for unmatched rows from Categories.
   Products වගුවෙන් සියලුම පේළි සහ Categories වලින් නොගැළපෙන පේළි සඳහා NULL අගයන්.
5) All rows from the Categories table and NULLs for unmatched rows from Products.
   Categories වගුවෙන් සියලුම පේළි සහ Products වලින් නොගැළපෙන පේළි සඳහා NULL අගයන්.

Answer: 3)

Explanation / පැහැදිලි කිරීම:

`INNER JOIN` returns only the rows where there is a match in both tables based on the condition `A.category_id = B.category_id`.
`A.category_id = B.category_id` යන කොන්දේසිය මත පදනම්ව, `INNER JOIN` මඟින් වගු දෙකෙහිම ගැළපීමක් ඇති පේළි පමණක් ආපසු ලබා දේ.

It does not return rows with `NULL` values or unmatched rows.
එය `NULL` අගයන් හෝ නොගැලපෙන පේළි සහිත පේළි ආපසු ලබා නොදේ.

The correct answer is / නිවැරදි පිළිතුර වන්නේ:

3) Only the rows where category_id matches in both tables.
   වගු දෙකෙහිම category_id ගැළපෙන පේළි පමණි.

Other options / වෙනත් විකල්ප:

Option / විකල්ප 1)
This would be true for a `CROSS JOIN`, not an `INNER JOIN`.
මෙය `CROSS JOIN` සඳහා සත්‍ය වනු ඇත, `INNER JOIN` සඳහා නොවේ.

Option / විකල්ප 2)
`INNER JOIN` excludes rows with `NULL` in the joining column (`category_id`).
`INNER JOIN` මඟින් සම්බන්ධ වන තීරුවේ (`category_id`) `NULL` සහිත පේළි බැහැර කරයි.

Option / විකල්ප 4)
This describes a `LEFT JOIN`.
මෙය `LEFT JOIN` එකක් විස්තර කරයි.

Option / විකල්ප 5)
This describes a `RIGHT JOIN`.
මෙය `RIGHT JOIN` එකක් විස්තර කරයි.

7.
```
Total = 0
for i in range(1, 20):
    if i % 3 == 0 and i % 5 != 0:
        Total += i
print(Total)
```

What will be the output of the code above?
ඉහත පයිතන් කේතයේ ප්‍රතිදානය කුමක් වේද?

1) 35 2) 38 3) 40 4) 45 5) <u>48</u>

Answer : 5)

We need to find the sum of all numbers between 1 and 19 that meet two conditions:
කොන්දේසි දෙකක් සපුරාලන 1 සහ 19 අතර සියලුම සංඛ්‍යාවල එකතුව සොයාගත යුතුය:

- Multiples (ගුණාකාර) of 3 (`i % 3 == 0`)
- Not multiples (ගුණාකාර නොවන) of 5 (`i % 5 != 0`)

<u>Step 1 / පියවර 1</u>
Find multiples of 3 up to 19.
3 සිට 19 දක්වා ගුණාකාර සොයන්න

Numbers that are multiples of 3 between 1 and 19 are:
1 සහ 19 අතර 3 ගුණාකාර සංඛ්‍යා වේ:

3, 6, 9, 12, 15, 18

<u>Step 2 / පියවර 2</u>
Exclude numbers that are also multiples of 5.
5 හි ගුණාකාර වන සංඛ්‍යා බැහැර කරමු

Among the multiples of 3 listed above, we exclude those that are also divisible by 5:
ඉහත ලැයිස්තුගත කර ඇති 3 හි ගුණාකාර අතර, 5න් බෙදිය හැකි ඒවා ද බැහැර කරමු:

3 — not divisible by 5 / 5 න් බෙදිය නොහැක ✔
6 — not divisible by 5 / 5 න් බෙදිය නොහැක ✔
9 — not divisible by 5 / 5 න් බෙදිය නොහැක ✔
12 — not divisible by 5 / 5 න් බෙදිය නොහැක ✔
15 — divisible by 5 / 5 න් බෙදිය නොහැක ❌ exclude / බැහැර කරන්න
18 — not divisible by 5 / 5 න් බෙදිය නොහැක ✔

So, we have the numbers 3, 6, 9, 12, and 18.
එබැවින් අපට 3, 6, 9, 12 සහ 18 අංක ඇත.

Step 3 / පියවර 3
Sum the valid numbers
වලංගු සංඛ්‍යා එකතු කරමු

Now, let's add these numbers together.
දැන්, මෙම සංඛ්‍යා එකට එකතු කරමු.

3 + 6 + 9 +12 + 18 = 48

Therefore, the correct answer is 5).
එබැවින්, නිවැරදි පිළිතුර 5) වේ.

8. What is the correct output of the following python code?
පහත පයිතන් ක්‍රමලේඛයේ ප්‍රතිදානය කුමක්ද?

```
dictionary = {'one': 'two', 'three': 'one', 'two': 'three'}
v = dictionary['one']
for k in range(len(dictionary)):
      v = dictionary[v]
print(v)
```

1) `one`   2) `two`   3) `three`   4) `one three two`   5) `two one three`

Answer : 2)

`dictionary = {'one': 'two', 'three': 'one', 'two': 'three'}`
The dictionary has the following key-value pairs:
dictionary එකෙහි පහත යතුරු අගය යුගල ඇත:

'one' → 'two'
'three' → 'one'
'two' → 'three'

`v = dictionary['one']`

Here, `dictionary['one']` is `'two'`. / මෙහි `dictionary['one']`, `'two'` වේ.
So, `v = 'two'`. / එබැවින්, `v = 'two'` වේ.

Iteration 1 / පුනරාවර්තනය 1:
From the dictionary, `'two'` maps to `'three'`. /
So, `v = 'three'`. /
dictionary එකෙහි සිට `'two'`, `'three'` ට අනුරූප වේ.
එබැවින්, `v = 'three'` වේ.

Iteration 2 / පුනරාවර්තනය 2:
From the dictionary, `'three'` maps to `'one'`. /
So, `v = 'one'`. /
dictionary එකෙහි සිට `'three'`, `'one'` ට අනුරූප වේ.
එබැවින්, `v = 'one'` වේ.

Iteration 3 / පුනරාවර්තනය 3:
From the dictionary, `'one'` maps to `'two'`. /
So, `v = 'two'`. /
dictionary එකෙහි සිට `'one'`, `'two'` ට අනුරූප වේ.
එබැවින්, `v = 'two'` වේ.

After the loop completes, the value of `v` is `'two'`. / ලූපය සම්පූර්ණ වූ පසු, `v` හි අගය `'two'` වේ.
The `print(v)` statement outputs `two`. / `print(v)` ප්‍රකාශය `two` ප්‍රතිදානය කරයි.

So, the correct answer number is 2.
එමනිසා, නිවැරදි පිළිතුරු අංකය 2 වේ.

9. What does the `len()` function do in Python?
Pythonහි `len()` ශ්‍රිතය කරන්නේ කුමක්ද?

1) Returns the last element of a list.
ලැයිස්තුවක අවසාන මූලාංගය ලබා දෙයි.
2) Returns the number of elements in an iterable.
පුනරාවර්තනයක ඇති මූලාංග ගණන ආපසු ලබා දෙයි.
3) Returns the type of an object.
වස්තුවක වර්ගය ආපසු ලබා දෙයි.
4) Returns the memory address of an object.
වස්තුවක මතක ලිපිනය ආපසු ලබා දෙයි.
5) None of the above.
ඉහත කිසිවක් නොවේ.

Answer: 2)

Explanation of `len()` / `len()` හි පැහැදිලි කිරීම:
The len() function in Python is used to determine the number of elements in an iterable object, such as a list, tuple, string, dictionary, set, or range.
Python හි `len()` ශ්‍රිතය ලැයිස්තුවක්, tuple එකක්, string එකක්, dictionary එකක්, කට්ටලයක් හෝ පරාසයක් වැනි පුනරාවර්තනය කළ හැකි වස්තුවක ඇති මූලාංග ගණන තීරණය කිරීමට භාවිතා කරයි.

```
my_list = [1, 2, 3, 4, 5]
print(len(my_list))  # Output: 5

my_string = "Python"
print(len(my_string))  # Output: 6
```

Explanation of each option / එක් එක් විකල්පය පැහැදිලි කිරීම:

Option / විකල්ප 1)
False. The `len()` function does not return the last element of a list. To access the last element, you can use negative indexing (`my_list[-1]`).
අසත්‍ය වේ. `len()` ශ්‍රිතය ලැයිස්තුවක අවසාන මූලාංගය ආපසු ලබා නොදේ. අවසාන මූලාංගය වෙත ප්‍රවේශ වීමට, ඔබට සෘණ සුචිගත කිරීම (`my_list[-1]`) භාවිතා කළ හැක.

Option / විකල්ප 2)
True. This is the correct behavior of the `len()` function.
සත්‍ය වේ. `len()` ශ්‍රිතයේ නිවැරදි හැසිරීම මෙයයි.

Option / විකල්ප 3)
False. To get the type of an object, you use the `type()` function.
අසත්‍ය වේ. වස්තුවක වර්ගය ලබා ගැනීමට, ඔබ `type()` ශ්‍රිතය භාවිතා කරයි.

Option / විකල්ප 4)
False. The `id()` function returns the memory address of an object.
අසත්‍ය වේ. `id()` ශ්‍රිතය වස්තුවක මතක ලිපිනය ලබා දෙයි.

Option / විකල්ප 5)
False. Option 2 is the correct description of the `len()` function.
අසත්‍ය වේ. විකල්ප 2 යනු `len()` ශ්‍රිතයේ නිවැරදි විස්තරයයි.

Therefore, the correct answer is 2).
එබැවින්, නිවැරදි පිළිතුර 2) වේ.

10. What attribute can be used in an ordered list to specify a different starting number?
වෙනත් ආරම්භක අංකයක් සඳහන් කිරීමට ordered ලැයිස්තුවක භාවිතා කළ හැකි ගුණාංගය කුමක්ද?

1) start
2) type
3) value
4) begin
5) count

Answer : 1)

To specify a different starting number in an ordered list, you can use the **start** attribute. This attribute allows you to define the starting number of the list, and it is supported in HTML5
Ordered ලැයිස්තුවක වෙනස් ආරම්භක අංකයක් සඳහන් කිරීමට, ඔබට **start** ගුණාංගය භාවිතා කළ හැක. මෙම ගුණාංගය ඔබට ලැයිස්තුවේ ආරම්භක අංකය නිර්වචනය කිරීමට ඉඩ සලසයි, එය HTML5 හි සහාය දක්වයි.

Here is an example / මෙන්න උදාහරණයක්

```
<ol start="5">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Item 3</li>
</ol>
```

In this example, the list will start with the number 5.
මෙම උදාහරණයේ දී, ලැයිස්තුව අංක 5 න් ආරම්භ වේ.

The other options are not correct because,
අනෙක් විකල්ප නිවැරදි නොවන්නේ,

type attribute: This is used to specify the type of numbering (e.g., numbers, letters, Roman numerals)
මෙය අංක කිරීමේ වර්ගය සඳහන් කිරීමට භාවිතා කරයි (උදා: ඉලක්කම්, අකුරු, රෝම ඉලක්කම්)

value attribute : This was used in older versions of HTML to set the value of a specific list item, but it is not typically used for setting the starting number of the entire list. මෙය විශේෂිත ලැයිස්තු අයිතමයක අගය සැකසීමට HTML හි පැරණි අනුවාද වල භාවිතා කර ඇත, නමුත් එය සම්පූර්ණ ලැයිස්තුවේ ආරම්භක අංකය සැකසීම සඳහා සාමාන්‍යයෙන් භාවිතා නොවේ.

begin attribute : There is no such attribute for ordered lists in HTML. HTML හි ordered ලැයිස්තු සඳහා එවැනි ගුණාංගයක් නොමැත

count attribute : There is no such attribute for ordered lists in HTML.
HTML හි ordered ලැයිස්තු සඳහා එවැනි ගුණාංගයක් නොමැත

Therefore, the correct answer is 1).
එබැවින් නිවැරදි පිළිතුර 1) වේ.