Information and Communication Technology
තොරතුරු හා සන්නිවේදන තාක්ෂණය

2026 QUIZ NO 315 MARKING

2025/10/21
ictfromabc
Ravindu Bandaranayake

1. Which of the following storage devices has the smallest data capacity?
පහත ආවයන උපාංග අතරින් කුඩාම දත්ත ධාරිතාවක් ඇති උපාංගය කුමක්ද?

   1) Compact Disk. / සංයුක්ත තැටිය.
   2) Random Access Memory. / සසම්භාවී ප්‍රවේශ මතකය.
   3) Flash Memory. / සැනෙලි මතකය.
   4) Cache Memory. / වාරක මතකය.
   5) Register Memory. / රෙජිස්තර මතකය.

Answer : 5)

Different storage devices have different capacities. The functions performed by these devices are also different. The data capacity of the devices is listed below in ascending order.
විවිධ ආවයන උපාංගවලට එකිනෙකට වෙනස් ධාරිතාවන් ඇත. මෙම උපාංග වලින් කෙරෙන කාර්යයන්ද එකිනෙකට වෙනස් ය. උපාංග වල දත්ත ධාරිතාව අඩු එකේ සිට වැඩි වන පිළිවෙලට පහත දක්වා ඇත.

Register Memory / රෙජිස්තර මතකය
Cache Memory / වාරක මතකය
Compact Disk (CD) / සංයුක්ත තැටිය
Digital Versatile Disc (DVD) / සංඛ්‍යාංක බහුවිධි තැටිය
Random Access Memory / සසම්භාවී ප්‍රවේශ මතකය
Read Only Memory / පඨන මාත්‍ර මතකය
Flash Memory / සැනෙලි මතකය
Hard Disk / දෘඪ තැටිය
Magnetic Tape / චුම්භක පටිය

So the device with the smallest data capacity is the register memory.
එබැවින් කුඩාම දත්ත ධාරිතාවයක් ඇති උපාංගය වන්නේ රෙජිස්තර මතකයයි.

2. This volatile device is a part of the CPU that acts as a buffer between the physical memory and the processor when running applications of the computer and it improves performance. What is this device?
මෙම විද්‍යුත් උපාංගය CPU උපාංගයේ කොටසක් වන අතර එය පරිගණකයේ යෙදුම් ධාවනය කිරීමේදී භෞතික මතකය සහ ප්‍රොසෙසරය අතර බෆරයක් ලෙස ක්‍රියා කරන අතර එය කාර්ය සාධනය වැඩි දියුණු කරයි, මෙම උපාංගය කුමක්ද?

1) RAM 2) ROM 3) L1 cache 4) Registers 5) Virtual memory

Answer : 3)

1) **RAM** (Random Access Memory)
RAM is a type of volatile memory used to temporarily store data and machine code currently being used by the computer. It allows for fast read and write access to a storage medium that is essential for the system's performance. While RAM significantly improves the performance of a computer by providing quick access to data for the CPU, it is not located within the CPU itself. Instead, RAM is part of the system memory and serves as a broader workspace for the entire operating system and applications, not specifically as a buffer between the CPU and the physical memory.
RAM යනු දැනට පරිගණකය භාවිතා කරන දත්ත සහ යන්ත්‍ර කේතය තාවකාලිකව ගබඩා කිරීමට භාවිතා කරන නශ්‍ය මතක වර්ගයකි. එය පද්ධතියේ ක්‍රියාකාරීත්වය සඳහා අත්‍යවශ්‍ය වන ගබඩා මාධ්‍යයකට වේගයෙන් කියවීමට සහ ලිවීමට ප්‍රවේශ වීමට ඉඩ සලසයි. CPU සඳහා දත්ත වෙත ඉක්මන් ප්‍රවේශයක් ලබා දීමෙන් RAM පරිගණකයේ ක්‍රියාකාරීත්වය සැලකිය යුතු ලෙස වැඩිදියුණු කරන අතර, එය CPU තුළම පිහිටා නොමැත. ඒ වෙනුවට, RAM යනු පද්ධති මතකයේ කොටසක් වන අතර, විශේෂයෙන්ම CPU සහ භෞතික මතකය අතර බෆරයක් ලෙස නොව, සම්පූර්ණ මෙහෙයුම් පද්ධතිය සහ යෙදුම් සඳහා පුළුල් වැඩ ඉඩක් ලෙස සේවය කරයි.

2) **ROM** (Read-Only Memory)
ROM is a type of non-volatile memory that retains its contents even when the computer is powered off. It is primarily used to store firmware, which includes the basic instructions for booting the computer and performing hardware initialization. Since ROM is read-only and not designed for the high-speed data transactions that occur during regular application processing, it does not serve as a performance-enhancing buffer between the CPU and physical memory. Therefore, ROM does not fit the description provided in the question.
ROM යනු පරිගණකය ක්‍රියා විරහිත වූ විට පවා එහි අන්තර්ගතය රඳවා තබා ගන්නා වාෂ්පශීලී නොවන මතක වර්ගයකි. එය මූලික වශයෙන් ෆර්මිවෙයාර් ගබඩා කිරීම සඳහා භාවිතා කරයි, පරිගණකය ආරම්භ කිරීම සහ දෘඩාංග ආරම්භ කිරීම සඳහා මූලික උපදෙස් ඇතුළත් වේ. ROM කියවීමට පමණක් වන අතර සාමාන්‍ය යෙදුම් සැකසීමේදී සිදුවන අධිවේගී දත්ත ගනුදෙනු සඳහා නිර්මාණය කර නොමැති බැවින්, එය CPU සහ භෞතික මතකය අතර කාර්ය සාධනය වැඩි කරන බෆරයක් ලෙස ක්‍රියා නොකරයි. එබැවින්, ප්‍රශ්නයේ දක්වා ඇති විස්තරයට ROM නොගැලපේ.

3) **L1 Cache**
The L1 cache is a small, extremely fast type of volatile memory located directly within the CPU. Its primary function is to act as a buffer between the CPU and the physical memory (RAM), storing the most frequently accessed data and instructions. This arrangement allows the CPU to quickly retrieve this data without the delays associated with accessing the main memory. By reducing the time needed for data retrieval, the L1 cache significantly enhances the overall performance of the CPU. Given its role and location, the L1 cache matches the description provided in the question, making it the correct answer.
L1 cache යනු CPU තුළ සෘජුවම පිහිටා ඇති කුඩා, අතිශය වේගවත් නශ්‍ය මතක වර්ගයකි. එහි මූලික කාර්යය වන්නේ CPU සහ භෞතික මතකය (RAM) අතර බෆරයක් ලෙස ක්‍රියා කිරීම, නිතර ප්‍රවේශ වන දත්ත සහ උපදෙස් ගබඩා කිරීමයි. මෙම විධිවිධානය ප්‍රධාන මතකයට ප්‍රවේශ වීම හා සම්බන්ධ ප්‍රමාදයකින් තොරව මෙම දත්ත ඉක්මනින් ලබා ගැනීමට CPU හට ඉඩ සලසයි. දත්ත ලබා ගැනීම සඳහා අවශ්‍ය කාලය අඩු කිරීමෙන්, L1 හැඹිලිය CPU හි සමස්ත ක්‍රියාකාරිත්වය සැලකිය යුතු ලෙස වැඩි කරයි. එහි භූමිකාව සහ ස්ථානය අනුව, L1 cache ප්‍රශ්නයේ සපයා ඇති විස්තරයට ගැලපේ, එය නිවැරදි පිළිතුර බවට පත් කරයි.

4) **Registers**
Registers are small storage locations within the CPU that hold data, instructions, and addresses temporarily during processing. They are the fastest type of memory within the CPU, used to execute instructions and perform calculations. While registers are crucial for CPU operations, they do not act as a buffer between the CPU and the physical memory. Instead, they are part of the CPU's immediate processing unit and handle specific, short-term tasks rather than improving performance by buffering data from the main memory.
රෙජිස්ටර් යනු CPU තුළ දත්ත, උපදෙස් සහ ලිපින සැකසීමේදී තාවකාලිකව රඳවා තබා ගන්නා කුඩා ගබඩා ස්ථාන වේ. ඒවා CPU තුළ ඇති වේගවත්ම මතක වර්ගය වන අතර, උපදෙස් ක්‍රියාත්මක කිරීමට සහ ගණනය කිරීම් සිදු කිරීමට භාවිතා කරයි. CPU මෙහෙයුම් සඳහා රෙජිස්ටර් ඉතා වැදගත් වන අතර, ඒවා CPU සහ භෞතික මතකය අතර බෆරයක් ලෙස ක්‍රියා නොකරයි. ඒ වෙනුවට, ඒවා CPU හි ක්ෂණික සැකසුම් ඒකකයේ කොටසක් වන අතර ප්‍රධාන මතකයෙන් දත්ත බෆර කිරීමෙන් කාර්ය සාධනය වැඩි දියුණු කරනවාට වඩා විශේෂිත, කෙටි කාලීන කාර්යයන් හසුරුවයි.

5) **Virtual Memory**
Virtual memory is a memory management technique that allows the computer to compensate for physical memory shortages by temporarily transferring data from RAM to disk storage. This process allows for the execution of larger applications than would otherwise fit in physical memory. While virtual memory helps manage memory more efficiently and can improve system performance in low-memory situations, it is not a small, fast buffer located within the CPU. Thus, it does not fit the description provided in the question.
අතථ්‍ය මතකය යනු RAM සිට තැටි ගබඩාව වෙත දත්ත තාවකාලිකව මාරු කිරීම මගින් භෞතික මතක හිඟය පියවා ගැනීමට පරිගණකයට ඉඩ සලසන මතක කළමනාකරණ ක්‍රමයකි. මෙම ක්‍රියාවලිය භෞතික මතකයට ගැලපෙන ප්‍රමාණයට වඩා විශාල යෙදුම් ක්‍රියාත්මක කිරීමට ඉඩ සලසයි. අතථ්‍ය මතකය මතකය වඩාත් කාර්යක්ෂමව කළමනාකරණය කිරීමට උපකාරී වන අතර අඩු මතක අවස්ථා වලදී පද්ධති ක්‍රියාකාරිත්වය වැඩිදියුණු කළ හැකි අතර, එය CPU තුළ පිහිටා ඇති කුඩා, වේගවත් බෆරයක් නොවේ. මේ අනුව, එය ප්‍රශ්නයේ දක්වා ඇති විස්තරයට නොගැලපේ.

So, the L1 cache is the device that acts as a buffer between the CPU and physical memory, improving performance by providing quick access to frequently used data and instructions, making it the correct answer.

එබැවින්, L1 cache යනු CPU සහ භෞතික මතකය අතර බෆරයක් ලෙස ක්‍රියා කරන උපාංගය වන අතර, නිතර භාවිතා කරන දත්ත සහ උපදෙස් වෙත ඉක්මන් ප්‍රවේශයක් ලබා දීමෙන් කාර්ය සාධනය වැඩි දියුණු කරයි, එය නිවැරදි පිළිතුර බවට පත් කරයි.

3. What is the correct answer when the boolean expression $(1B + (0D + 1A)B)A + D\overline{D}$ is trivialized?
$(1B + (0D + 1A)B)A + D\overline{D}$ යන බූලියානු ප්‍රකාශනය සුළු කළ විට නිවැරදි පිළිතුර කුමක්ද?

1) 1 2) 0 3) $(B+AB).\overline{A}$ 4) $\underline{B.A}$ 5) D.A

Answer : 4)

This boolean expression can be simplfied as follows.
මෙම බූලියානු ප්‍රකාශනය පහත ආකාරයට සුළු කළ හැකිය.

$= (1B + (0D + 1A)B)A + D\overline{D}$
$= (1B + (0 + 1A)B)A + D\overline{D}$ Identity law සර්වසාම්‍ය න්‍යාය
$= (1B + 1AB)A + D\overline{D}$ Identity law සර්වසාම්‍ය න්‍යාය
$= (B + 1AB)A + D\overline{D}$ Identity law සර්වසාම්‍ය න්‍යාය
$= (B + AB)A + D\overline{D}$ Identity law සර්වසාම්‍ය න්‍යාය
$= (B + AB)A + 0$ Inverse law ප්‍රතිලෝම න්‍යාය
$= (B + AB)A$ Identity law සර්වසාම්‍ය න්‍යාය
$= B.A$ Redunclancy law සමනිරික්ත න්‍යාය

Therefore the correct answer is **4).**
එබැවින් නිවැරදි පිළිතුර **4)** වේ.

4. What is the result of trivializing the boolean expression $1A + D(BB + \overline{DB})$?
$1A + D(BB + \overline{DB})$ යන බූලිය ප්‍රකාශනය සුළු කළ විට ලැබෙන ප්‍රතිඵලය කුමක්ද?

1) 1 2) 0 3) $(B+AB).\overline{A}$ 4) $\underline{A+D}$ 5) D.A

Answer : 4)

This can be done as follows,
මෙය පහත පරිදි සිදුකර හැක,

$= 1A + D(BB + \overline{DB})$
$= 1A + D(BB + \overline{D} + \overline{B})$ Demorgan law / ද මුවාර් ප්‍රමේය
$= A + D(BB + \overline{D} + \overline{B})$ Identity law / සර්වසාම්‍ය න්‍යාය
$= A + D(B + \overline{D} + \overline{B})$ Idempotent law / තද්වභාවී න්‍යාය
$= A + D(\overline{D} + 1)$ Complement law / ප්‍රතිලෝම න්‍යාය
$= A + D1$ Identity law / සර්වසාම්‍ය න්‍යාය
$= A + D$ Identity law / සර්වසාම්‍ය න්‍යාය

Therefore the correct answer is **4).**
එබැවින් නිවැරදි පිළිතුර **4)** වේ.

5. $G\overline{A}M + \overline{MK}$ What is the simplified answer? / සරල කළ පිළිතුර වන්නේ කුමක්ද ?

1) $\underline{G\overline{A} + \overline{M} + \overline{K}}$ 2) $\overline{M} + \overline{K}$ 3) $\overline{A}$ 4) A 5) G

Answer : 1)

This can be achieved as follows, / මෙය පහත පරිදි ලබාගත හැක,

$G\overline{A}M + \overline{MK}$
$G\overline{A}M + \overline{M} + \overline{K}$ De morgan Law
$G\overline{A} + \overline{M} + \overline{K}$ Idempotent Law

Therefore, the answer is $G\overline{A} + \overline{M} + \overline{K}$. / එමනිසා, පිළිතුර වන්නේ $G\overline{A} + \overline{M} + \overline{K}$ වේ.

6. Why would you avoid storing derived attributes directly in a table?
ව්‍යුත්පන්න උපලක්ෂණ සෘජුවම වගුවක ගබඩා කිරීමෙන් ඔබ වැළකී සිටින්නේ ඇයි?

   1) To improve query performance.
   විමසුම් කාර්ය සාධනය වැඩි දියුණු කිරීම සඳහා.

   2) To avoid redundancy and inconsistency.
   අතිරික්තය සහ නොගැළපීම වැළැක්වීම සඳහා.

   3) To save storage space in the database.
   දත්ත ගබඩාවේ ගබඩා ඉඩ ඉතිරි කර ගැනීමට.

   4) To allow derived attributes to be queried independently.
   ව්‍යුත්පන්න උපලක්ෂණ ස්වාධීනව විමසීමට ඉඩ දීම.

   5) To enforce referential integrity.
   යොමු අඛණ්ඩතාව බලාත්මක කිරීමට.

Answer : 2)

Explanation / පැහැදිලි කිරීම

Storing derived attributes directly in a table is avoided primarily to prevent redundancy and inconsistency, which is why the correct answer is Option 2. Derived attributes are values that can be calculated or derived from other data in the database. If they are stored directly, any change in the base data might not automatically update the derived values, leading to inconsistencies. Additionally, storing these attributes unnecessarily duplicates data, increasing redundancy. This practice conflicts with database normalization principles aimed at ensuring efficient and reliable data storage.
අතිරික්තය සහ නොගැලපීම වැලැක්වීම සඳහා ව්‍යුත්පන්න උපලක්ෂණ සෘජුවම වගුවක ගබඩා කිරීම වැළකෙයි, එබැවින් නිවැරදි පිළිතුර විකල්ප 2 වේ. ව්‍යුත්පන්න උපලක්ෂණ යනු දත්ත සමුදායේ ඇති අනෙකුත් දත්ත වලින් ගණනය කළ හැකි හෝ ව්‍යුත්පන්න කළ හැකි අගයන් වේ. ඒවා සෘජුවම ගබඩා කර ඇත්නම්, මූලික දත්තවල යම් වෙනසක් ව්‍යුත්පන්න අගයන් ස්වයංක්‍රියව යාවත්කාලීන නොකරනු ඇත, එය නොගැලපීම් වලට තුඩු දෙයි. අතිරේකව, මෙම උපලක්ෂණ ගබඩා කිරීම අනවශ්‍ය ලෙස දත්ත අනුපිටපත් කරයි, අතිරික්තය වැඩි කරයි. මෙම පරිවය කාර්යක්ෂම සහ විශ්වාසනීය දත්ත ගබඩා කිරීම සහතික කිරීම අරමුණු කරගත් දත්ත සමුදාය ප්‍රමාණකරණ මූලධර්ම සමඟ ගැටේ.

Other options are incorrect because they do not align with the primary reason for avoiding the storage of derived attributes. For instance, while improving query performance Option 1 and saving storage space Option 3 are important considerations, these are not the main reasons for avoiding derived attribute storage. Allowing derived attributes to be queried independently Option 4 is irrelevant, as derived attributes can be recalculated when needed. Enforcing referential integrity Option 5 applies to maintaining consistent relationships between tables, which is unrelated to storing derived attributes.
ව්‍යුත්පන්න උපලක්ෂණ ගබඩා කිරීම වැළැක්වීමේ මූලික හේතුව සමඟ නොගැලපෙන නිසා වෙනත් විකල්ප වැරදිය. උදාහරණයක් ලෙස, විමසුම් කාර්ය සාධනය වැඩි දියුණු කිරීම විකල්ප 1 සහ ගබඩා ඉඩ ඉතිරි කිරීම විකල්ප 3 වැදගත් සලකා බැලිම් වන අතර, ව්‍යුත්පන්න උපලක්ෂණ ආවයනය වළක්වා ගැනීමට ප්‍රධාන හේතු මේවා නොවේ. ව්‍යුත්පන්න උපලක්ෂණ ස්වාධීනව විමසීමට ඉඩ දීම විකල්ප 4 අදාල නොවේ, මන්ද ව්‍යුත්පන්න උපලක්ෂණ අවශ්‍ය විට නැවත ගණනය කළ හැක. යොමු කිරීමේ අඛණ්ඩතාව බලාත්මක කිරීම විකල්ප 5 ව්‍යුත්පන්න ගුණාංග ගබඩා කිරීමට සම්බන්ධ නොවන වගු අතර ස්ථාවර සම්බන්ධතා පවත්වා ගැනීමට අදාළ වේ.

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

7. In the context of relational databases, which of the following characteristics is unique to many-to-many relationships?
සම්බන්ධතා දත්ත සමුදායන්හි සන්දර්භය තුළ, බහු-බහු සම්බන්ධතා සඳහා අනන්‍ය වන පහත සඳහන් ලක්ෂණ මොනවාද?

   1) Requires each table to have a unique identifier.
   සෑම වගුවකටම අනන්‍ය හඳුනාගැනීමක් තිබීම අවශ්‍ය වේ.
   2) Requires two foreign keys in one of the original tables.
   මුල් වගු වලින් එකකට ආගන්තුක යතුරු දෙකක් අවශ්‍ය වේ.
   3) Needs a separate table to establish the relationship.
   සම්බන්ධතාවය ස්ථාපිත කිරීම සඳහා වෙනම වගුවක් අවශ්‍ය වේ.
   4) Cannot exist without primary keys.
   ප්‍රාථමික යතුරු නොමැතිව පැවතිය නොහැක.
   5) Allows records in both tables to remain independent.
   වගු දෙකෙහිම වාර්තා ස්වාධීනව පැවතීමට ඉඩ සලසයි.

Answer : 3)

Explanation / පැහැදිලි කිරීම

In the context of relational databases, many-to-many relationships require a separate table, known as a junction or bridge table, to establish the relationship between the two original tables. This table typically contains foreign keys referencing the primary keys of the related tables, enabling the connection of multiple records from one table to multiple records in the other. Option 3 is correct as it accurately describes this characteristic.
සම්බන්ධතා දත්ත සමුදායේ සන්දර්භය තුළ, බොහෝ සම්බන්ධතා සඳහා මුල් වගු දෙක අතර සම්බන්ධතාවය ස්ථාපිත කිරීම සඳහා junction හෝ bridge වගුවක් ලෙස හැඳින්වෙන වෙනම වගුවක් අවශ්‍ය වේ. මෙම වගුවේ සාමාන්‍යයෙන් අදාළ වගුවල ප්‍රාථමික යතුරු යොමු කරන ආගන්තුක යතුරු අඩංගු වන අතර, එක් වගුවක සිට අනෙක් වාර්තා කිහිපයකට බහු වාර්තා සම්බන්ධ කිරීම සක්‍රිය කරයි. මෙම ලක්ෂණය නිවැරදිව විස්තර කරන බැවින් විකල්ප 3 නිවැරදි වේ.

The other options are not unique to many-to-many relationships. Option 1 refers to unique identifiers, which are necessary for any table in a relational database. Option 2 describes foreign keys, but many-to-many relationships require foreign keys in the junction table, not the original tables. Option 4 is incorrect because primary keys are a requirement for all tables, not specifically for many-to-many relationships. Option 5 is incorrect as many-to-many relationships inherently involve interdependencies between tables.
වෙනත් විකල්ප බොහෝ සම්බන්ධතා සඳහා අනන්‍ය නොවේ. විකල්ප 1 යනු සම්බන්ධතා දත්ත ගබඩාවක ඕනෑම වගුවක් සඳහා අවශ්‍ය වන අනන්‍ය හඳුනාගැනීම් වලට යොමු වේ. විකල්ප 2 විදේශීය යතුරු විස්තර කරයි, නමුත් බොහෝ සම්බන්ධතා සඳහා මුල් වගු නොව හන්දි වගුවේ විදේශීය යතුරු අවශ්‍ය වේ. ප්‍රාථමික යතුරු සියලු වගු සඳහා අවශ්‍ය වන බැවින්, විශේෂයෙන් බොහෝ සම්බන්ධතා සඳහා නොව, විකල්ප 4 වැරදිය. බොහෝ සම්බන්ධතා වල සහජයෙන්ම වගු අතර අන්තර් යැපීම් සම්බන්ධ වන බැවින් විකල්ප 5 වැරදි වේ.

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

8. What will be the output of the following Python code?
පහත දැක්වෙන පයිතන් කේතයේ ප්‍රතිදානය කුමක්ද?

```
def process_list(lst):
    for i in range(len(lst)):
        if i % 2 == 0:
            lst[i] = lst[i] * 3
        else:
            lst[i] = lst[i] - 2
    return lst

data = [2, 4, 6, 8, 10]
processed = process_list(data.copy())

print(processed)
print(data)
```

1) `[6, 2, 18, 6, 30]`
   `[6, 2, 18, 6, 30]`

2) `[6, 2, 18, 6, 10]`
   `[2, 4, 6, 8, 10]`

3) `[6, 2, 18, 6, 10]`
   `[6, 4, 6, 8, 10]`

4) `[6, 2, 18, 6, 30]`
   `[2, 4, 6, 8, 10]`

5) `[6, 2, 18, 6, 10]`
   `[2, 2, 6, 6, 10]`

Answer : 4)

The code processes a copy of the list `data` by iterating through each element. For even indices, the corresponding elements are multiplied by 3, and for odd indices, 2 is subtracted. The processed list `[6, 2, 18, 6, 30]` is then printed, while the original list `data` remains unchanged as `[2, 4, 6, 8, 10]` since the function works on a copy of the list, not modifying the original.
කේතය එක් එක් මූලාංග හරහා පුනරාවර්තනය කිරීමෙන් ලැයිස්තුවේ `data` පිටපතක් සකසයි. ඉරට්ටේ දර්ශක සඳහා, අනුරූප මූලාංග 3 න් ගුණ කරනු ලබන අතර, ඔත්තේ දර්ශක සඳහා, 2 අඩු කරනු ලැබේ. පසුව සැකසූ ලැයිස්තුව `[6, 2, 18, 6, 30]` මුද්‍රණය කරනු ලබන අතර, ශ්‍රිතය පිටපතක ක්‍රියා කරන බැවින් මුල් `data` ලැයිස්තුව `[2, 4, 6, 8, 10]` ලෙස නොවෙනස්ව පවතී. ලැයිස්තුවේ, මුල් පිටපත වෙනස් නොකරයි.

9. What will be the output of the following code?
පහත කේතයේ ප්‍රතිදානය කුමක්ද?

```
result = []
for i in range(6):
    if i % 2 == 0:
        result.append(i * 2)
    else:
        result.append(i ** 2)

print(result)
```

1) `[0, 1, 4, 9, 8, 25]`
2) `[0, 1, 4, 9, 16, 25]`
3) `[0, 2, 4, 9, 8, 25]`
4) `[0, 2, 8, 9, 16, 25]`
5) `[0, 2, 4, 16, 8, 25]`

Answer : 1)

This Python code initializes an empty list called `result` and then iterates over a range of numbers from 0 to 5 using a for loop. Inside the loop, it checks if the current number i is even by using the condition i % 2 == 0.
මෙම පයිතන් කේතය `result` නමින් හිස් ලැයිස්තුවක් ආරම්භ කරන අතර පසුව for loop භාවිතා කරමින් 0 සිට 5 දක්වා සංඛ්‍යා පරාසයක් හරහා පුනරාවර්තනය වේ. ලූපය තුළ, එය i % 2 == 0 කොන්දේසිය භාවිතයෙන් වත්මන් අංකය i ඉරට්ටේ දැයි පරීක්ෂා කරයි.

If i is even, the code appends i * 2 to the `result` list. If i is odd, the code appends i ** 2 to the `result` list. This process continues for each iteration of the loop.
i ඉරට්ටේ නම්, කේතය i * 2 `result` ලැයිස්තුවට එකතු කරයි. i ඔත්තේ නම්, කේතය i ** 2 `result` ලැයිස්තුවට එකතු කරයි. ලූපයේ එක් එක් පුනරාවර්තනය සඳහා මෙම ක්‍රියාවලිය දිගටම පවතී.

The even numbers (0, 2, 4) are multiplied by 2, resulting in 0, 4, and 8, respectively. The odd numbers (1, 3, 5) are squared, resulting in 1, 9, and 25 respectively.
ඉරට්ටේ (0, 2, 4) 2 න් ගුණ කර, පිළිවෙළින් 0, 4, සහ 8 ලැබේ. ඔත්තේ සංඛ්‍යා (1, 3, 5) වර්ග කර ඇති අතර, එහි ප්‍රතිඵලය ලෙස පිළිවෙලින් 1, 9 සහ 25 ලැබේ.

Finally, the code prints the `result` list, which will output `[0,1,4,9,8,25].`
අවසාන වශයෙන්, කේතය `result` ලැයිස්තුව මුද්‍රණය කරයි, එය `[0,1,4,9,8,25]` ප්‍රතිදානය කරයි.

Therefore the correct answer is **1)** `[0,1,4,9,8,25]`
එබැවින් නිවැරදි පිළිතුර **1)** `[0,1,4,9,8,25]` වේ.

10. Consider the following Python code.
පහත පයිතන් කේතය සලකා බලන්න.

```
def func(n):
    if n == 1:
        return 1
    else:
        return n + func(n - 1)

print(func(4))
```

Which of the following statements about this code is true?
මෙම කේතය පිළිබඳ පහත සඳහන් කුමන ප්‍රකාශය සත්‍ය වන්නේද?

1) The function `func(n)` calculates the sum of all integers from `n` to 1 and returns the result.
   `func(n)` ශ්‍රිතය n සිට 1 දක්වා සියලුම නිඛිලවල එකතුව ගණනය කර ප්‍රතිඵලය ලබා දෙයි.
2) The function `func(n)` will return the value 4 when called with the argument 4.
   `func(n)` ශ්‍රිතය 4 තර්කය(argument) සමගින් ඇමතූ විට අගය 4 ලබා දෙනු ඇත.
3) The base case of the recursion is `n == 0`, so the function will return 0 when `n` is 0.
   පුනරාවර්තනයේ පාදක(base) අවස්ථාව `n == 0` වේ, එබැවින් ශ්‍රිතය `n,` 0 වන විට 0 ලබාදෙයි.
4) The function `func(n)` will return 15 when called with the argument 4.
   `func(n)` ශ්‍රිතය 4 තර්කය(argument) සමගින් ඇමතූ විට 15 ලබා දෙනු ඇත.
5) The function `func(n)` will work correctly if `n` is less than 1.
   `func(n)` ශ්‍රිතය `n,` 1ට වඩා අඩු නම් නිවැරදිව ක්‍රියා කරයි.

Answer : 1)

Let's review code.
අපි කේතය සමාලෝචනය කරමු.

The function **`func(n)`** checks if **`n == 1`**. If true, it returns **`1`**. Otherwise, it returns **`n + func(n - 1)`**, which means the function calls itself, reducing `n` by `1` each time, until `n` becomes `1`.
`func(n)` ශ්‍රිතය **`n == 1`** නම් පරීක්ෂා කරයි. එය සත්‍ය නම්, 1 ලබා දෙයි. එසේ නොමැති නම්, එය **`n + func(n - 1)`** ලබා දෙයි, එනම් ශ්‍රිතය තමාටම අමතා, `n,` 1 බවට පත් වන තෙක් සෑම අවස්ථාවකම n වලින් 1 ක් අඩු කරයි.

`If n = 4:`

A. `func(4) = 4 + func(3)`
B. `func(3) = 3 + func(2)`
C. `func(2) = 2 + func(1)`
D. `func(1) = 1 (base case)`

After the recursive calls, all the return values are added together.
පුනරාවර්තන ඇමතුම් වලින් පසුව, සියලු ආපසු ලබා දෙන අගයන් එකට එකතු කරයි.

```
4 + 3 + 2 + 1 = 10
```

So, `func(4)` will return 10.
ඉතින්, `func(4)` **-** `10` **ලබා දෙයි.**

Let's review each option.
අපි එක් එක් විකල්පය සමාලෝචනය කරමු.

- **Option / විකල්පය 1)**
  This is valid option. The function sums up the integers from `n` down to `1`.
  මෙය වලංගු විකල්පයකි. ශ්‍රිතය මගින් නිඛිල `n` **සිට** `1` **දක්වා එකතු කරයි.**

- **Option / විකල්පය 2)**
  This is invalid option. As seen above, it returns `10`, not `4`.
  මෙය වලංගු නොවන විකල්පයකි. ඉහත දැක්වෙන පරිදි, එය `4` **නොව** `10` **ලබා දෙයි.**

- **Option / විකල්පය 3)**
  This is invalid option. The base case is `n == 1`, not `n == 0`.
  මෙය වලංගු නොවන විකල්පයකි. මූලික අවස්ථාව `n == 1` **වේ,** `n == 0` **නොවේ.**

- **Option / විකල්පය 4)**
  This is invalid option. The correct result is `10`, not 15.
  මෙය වලංගු නොවන විකල්පයකි. නිවැරදි ප්‍රතිඵලය `15` **නොව** `10` **වේ.**

- **Option / විකල්පය 5)**
  This is invalid option. If `n` is less than 1, the function will not terminate correctly as there is no base case for `n < 1`.
  මෙය වලංගු නොවන විකල්පයකි. `n,` 1 **ට වඩා අඩු නම්,** `n < 1` **සඳහා** base **අවස්ථාවක් නොමැති බැවින් ශ්‍රිතය නිවැරදිව අවසන් නොවේ.**

Correct Answers is Option **1).**
නිවැරදි පිළිතුර විකල්ප 1) **වේ.**