සියලු ම හිමිකම් ඇවිරිණි / *All Rights Reserved*]

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

Information and Communication Technology

තොරතුරු හා සන්නිවේදන තාක්ෂණය

2026 QUIZ NO 241 MARKING

2025/08/08
ictfromabc
Ravindu Bandaranayake

1. Which of the following is located outside the Microprocessor?
ක්ෂුද්‍ර සකසනයෙන් පිටත පිහිටා ඇත්තේ පහත ඒවායින් කුමක් ද?

   1) ALU / අංක ගණිතමය හා තාර්කික ඒකකය.
   2) Regiseters / රෙජිස්තර.
   3) RAM / සසම්භාවී ප්‍රවේශ මතකය.
   4) Level 1 Cache Memory / පළමු මට්ටමේ වාරක මතකය.
   5) Control Unit / පාලන ඒකකය.

   Answer : 3)

   Option / විකල්ප 1)
   The Arithmetic Logic Unit (ALU) is a key component located inside the microprocessor. It performs fundamental arithmetic operations, such as addition and subtraction, as well as logical operations like comparisons. The ALU is essential for executing instructions and processing data, making it a core part of the CPU's functionality.
   අංක ගණිත තාර්කික ඒකකය (ALU) යනු මයික්‍රොප්‍රොසෙසරය තුළ පිහිටා ඇති ප්‍රධාන අංගයකි. එය එකතු කිරීම සහ අඩු කිරීම වැනි මූලික ගණිතමය මෙහෙයුම් මෙන්ම සංසන්දනය වැනි තාර්කික මෙහෙයුම් සිදු කරයි. උපදෙස් ක්‍රියාත්මක කිරීමට සහ දත්ත සැකසීමට ALU අත්‍යවශ්‍ය වන අතර එය CPU හි ක්‍රියාකාරීත්වයේ මූලික කොටසක් බවට පත් කරයි.

   Option / විකල්ප 2)
   Registers are small, high-speed storage locations that are also found inside the microprocessor. They temporarily hold data and instructions that the CPU is currently processing. Because registers are faster than other types of memory, they help optimize performance by allowing the CPU to access data quickly without delays from slower memory sources.
   රෙජිස්ටර් යනු මයික්‍රොප්‍රොසෙසරය තුළ ද ඇති කුඩා, අධිවේගී ගබඩා ස්ථාන වේ. CPU දැනට සකසන දත්ත සහ උපදෙස් ඔවුන් තාවකාලිකව රඳවා තබා ගනී. රෙජිස්ටර් අනෙකුත් මතක වර්ග වලට වඩා වේගවත් බැවින්, මන්දගාමී මතක ප්‍රභවයන්ගෙන් ප්‍රමාදයකින් තොරව ඉක්මනින් දත්ත වෙත ප්‍රවේශ වීමට ඍජු හට ඉඩ දීමෙන් කාර්ය සාධනය ප්‍රශස්ත කිරීමට ඒවා උපකාරී වේ.

   Option / විකල්ප 3)
   Random Access Memory (RAM) is a type of volatile memory located outside the microprocessor. It serves as the main working memory for the CPU, storing data and instructions that are actively in use. RAM is essential for running programs and processes, and its speed and capacity directly impact overall system performance.
   සසම්භාවී ප්‍රවේශ මතකය (RAM) යනු මයික්‍රොප්‍රොසෙසරයෙන් පිටත පිහිටි වාෂ්පශීලී මතක වර්ගයකි. එය CPU සඳහා ප්‍රධාන ක්‍රියාකාරී මතකය ලෙස ක්‍රියා කරයි, සක්‍රීයව භාවිතයේ පවතින දත්ත සහ උපදෙස් ගබඩා කරයි. වැඩසටහන් සහ ක්‍රියාවලි ක්‍රියාත්මක කිරීම සඳහා RAM අත්‍යවශ්‍ය වන අතර එහි වේගය සහ ධාරිතාව සමස්ත පද්ධතියේ ක්‍රියාකාරිත්වයට සෘජුවම බලපායි.

   Option / විකල්ප 4)
   Level 1 Cache Memory is a small, ultra-fast memory cache located inside the microprocessor. It stores frequently accessed data and instructions to reduce latency and speed up processing. By providing the CPU with quick access to critical information, Level 1 cache enhances overall system efficiency and performance.
   Level 1 Cache Memory යනු මයික්‍රොප්‍රොසෙසරය තුළ පිහිටා ඇති කුඩා, අතිශය වේගවත් මතක හැඹිලියකි. එය ප්‍රමාදය අඩු කිරීමට සහ සැකසීම වේගවත් කිරීමට නිතර ප්‍රවේශ වන දත්ත සහ උපදෙස් ගබඩා කරයි. CPU වෙත තීරණාත්මක තොරතුරු වෙත ඉක්මන් ප්‍රවේශයක් ලබා දීමෙන්, Level 1 cache මඟින් සමස්ත පද්ධතියේ කාර්යක්ෂමතාව සහ කාර්ය සාධනය වැඩි දියුණු කරයි.

Option / විකල්ප 5)
The Control Unit is another vital component located within the microprocessor. It orchestrates the operations of the CPU by directing how instructions are executed and coordinating the activities of the ALU, registers, and memory. The Control Unit interprets instructions fetched from memory and generates control signals, ensuring that all parts of the microprocessor work together seamlessly.
Control Unit යනු මයික්‍රොප්‍රොසෙසරය තුළ පිහිටා ඇති තවත් වැදගත් අංගයකි. උපදෙස් ක්‍රියාත්මක වන ආකාරය සහ ALU රෙජිස්ටර් සහ මතකයේ ක්‍රියාකාරකම් සම්බන්ධීකරණය කිරීම මගින් එය CPU හි මෙහෙයුම් සංවිධානය කරයි. පාලන ඒකකය මතකයෙන් ලබාගත් උපදෙස් අර්ථකථනය කර පාලන සංඥා ජනනය කරයි, මයික්‍රොප්‍රොසෙසරයේ සියලුම කොටස් බාධාවකින් තොරව එකට ක්‍රියා කරන බව සහතික කරයි.

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

2. A memory that synchronizes with the computers system clock,
පරිගණක පද්ධතියේ ඔරලෝසුව සමඟ සමමුහුර්තනය වන මතකයක් වන්නේ,

1) DRAM 2) SRAM 3) SDRAM 4) CMOS 5) EPROM

Answer : 3)

The question is asking about a type of memory that synchronizes with the computer's system clock, meaning the memory operates in sync with the clock cycle of the processor, ensuring smooth data transfers and better performance.
ප්‍රශ්නය අසන්නේ පරිගණකයේ පද්ධති ඔරලෝසුව සමඟ සමමුහුර්ත වන මතක වර්ගයක් ගැන ය, එනම් මතකය ප්‍රොසෙසරයේ ඔරලෝසු චක්‍රය සමඟ සමමුහුර්තව ක්‍රියාත්මක වන අතර සුමට දත්ත හුවමාරුවක් සහ වඩා හොඳ ක්‍රියාකාරිත්වයක් සහතික කරයි.

- Among the given options, DRAM (Dynamic Random Access Memory) is a type of volatile memory, but it does not explicitly synchronize with the system clock.
  ලබා දී ඇති විකල්ප අතර, DRAM (ගතික සසම්භාවී ප්‍රවේශ මතකය) යනු නශ්‍ය මතක වර්ගයකි, නමුත් එය පද්ධති ඔරලෝසුව සමඟ පැහැදිලිව සමමුහුර්ත නොවේ.

- SRAM (Static Random Access Memory) is faster but also not tied to the clock in the same way.
  SRAM (ස්ථිතික සසම්භාවී ප්‍රවේශ මතකය) වේගවත් නමුත් ඔරලෝසුව සමඟ එකම ආකාරයකින් බැඳී නොමැත.

- SDRAM (Synchronous Dynamic Random Access Memory) is the correct answer because it synchronizes with the system clock, allowing data to be processed in sync with the CPU's operations.
  SDRAM (Synchronous Dynamic Random Access Memory) නිවැරදි පිළිතුර වන්නේ එය පද්ධති ඔරලෝසුව සමඟ සමමුහුර්ත වන නිසා, CPU මෙහෙයුම් සමඟ දත්ත සමමුහුර්ත කිරීමට ඉඩ සලසයි.

- CMOS and EPROM are types of memory used for different purposes, such as storing BIOS settings or data that can be erased and reprogrammed, but they don't synchronize with the system clock.
  CMOS සහ EPROM යනු BIOS සැකසුම් ගබඩා කිරීම හෝ මැකීමට සහ නැවත ක්‍රමලේඛනය කළ හැකි දත්ත වැනි විවිධ අරමුණු සඳහා භාවිතා කරන මතක වර්ග වේ, නමුත් ඒවා පද්ධති ඔරලෝසුව සමඟ සමමුහුර්ත නොවේ.

Therefore, the correct answer is Option 3) SDRAM, as it operates in sync with the system clock to ensure efficient memory access.
එබැවින්, නිවැරදි පිළිතුර වන්නේ විකල්ප 3) SDRAM එය කාර්යක්ෂම මතක ප්‍රවේශය සහතික කිරීම සඳහා පද්ධති ඔරලෝසුව සමඟ සමමුහුර්තව ක්‍රියාකරයි.

3. Which of the following represents the bitwise XOR operation of two binary numbers 10111100_2 and 01000111_2 ?
පහත කවරක් 10111100_2 සහ 01000111_2 යන ද්විමය සංඛ්‍යා දෙකෙහි බිටු අනුසාරිත XOR මෙහෙයුම නිරූපණය කරයි ද?

1) 11100111_2 2) $\underline{11111011_2}$ 3) 11001100_2 4) 10101011_2 5) 11111111_2

Answer : 2)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Above is the truth table of the XOR operation.
ඉහතින් දක්වා ඇත්තේ XOR මෙහෙයුමෙහි සත්‍යතා වගුව වේ.

10111100 Bitwise XOR 01000111
$1\ 0\ 1\ 1\ 1\ 1\ 0\ 0_2$ $\underline{0\ 1\ 0\ 0\ 0\ 1\ 1\ 1_2}$ $\underline{\underline{1\ 1\ 1\ 1\ 1\ 0\ 1\ 1_2}}$

So the correct answer is 11111011_2.
එබැවින් නිවැරදි පිළිතුර 11111011_2 වේ.

4. Which of the following statements about 1's complement is true?
එකෙහි අනුපූරකය පිළිබඳ පහත සඳහන් කුමන ප්‍රකාශය සත්‍ය ද?

1) It represents negative numbers by inverting the bits and adding 1.
එය බිටු පෙරළීමෙන් සහ 1ක් එකතු කිරීමෙන් සෘණ සංඛ්‍යා නියෝජනය කරයි.

2) It has two representations for zero.
එහි ශූන්‍යයට නිරූපණයන් දෙකක් ඇත.

3) It's the standard method used in most computers for signed numbers.
බොහෝ පරිගණකවල ලකුණුවත් සංඛ්‍යා සඳහා භාවිතා කරන සම්මත ක්‍රමය එයයි.

4) It requires an additional sign bit to indicate positive or negative.
ධන හෝ සෘණ බව දැක්වීමට අමතර ලකුණු බිටුවක් අවශ්‍ය වේ.

5) It always results in unique binary patterns for positive and negative numbers.
එය සෑම විටම ධන සහ සෘණ සංඛ්‍යා සඳහා අනන්‍ය ද්විමය රටා ඇති කරයි.

Answer : 2)

Option / විකල්පය 1
This describes 2's complement, not 1's complement. In 1's complement, negative numbers are represented by inverting all bits without adding 1.
මෙය විස්තර කරන්නේ 2 හි අනුපූරකය මිස 1 අනුපූරකය නොවේ. 1 හි අනුපූරකයේ, 1 එකතු නොකර සියලුම බිටු ප්‍රතිලෝම කිරීමෙන් සෘණ සංඛ්‍යා නිරූපණය කෙරේ.

Option / විකල්පය 2
True. In 1's complement, positive zero is represented as all 0s, and negative zero is represented as all 1s. This is a unique feature of 1's complement systems.
සත්‍යයි. 1 හි අනුපූරකයේ, ධන ශූන්‍යය සියල්ල 0 ලෙසත්, සෘණ ශූන්‍යය සියල්ල 1 ලෙසත් නිරූපණය කෙරේ. මෙය 1 හි අනුපූරක පද්ධතිවල සුවිශේෂී ලක්ෂණයකි.

Option / විකල්පය 3
This is false. Most modern computers use 2's complement because it eliminates the issue of two zeros and simplifies arithmetic operations.
අසත්‍යයි. බොහෝ නවීන පරිගණක 2 හි අනුපූරකය භාවිතා කරන්නේ එය ශූන්‍ය දෙකක ගැටළුව ඉවත් කර අංක ගණිත ක්‍රියාකාරකම් සරල කරන බැවිනි.

Option / විකල්පය 4
This is partially true but misleading. While 1's complement uses the most significant bit (MSB) as a sign bit (0 for positive, 1 for negative), this is also true for other systems like 2's complement. The "additional" bit phrasing makes it incorrect here.
මෙය අර්ධ වශයෙන් සත්‍ය නමුත් කොටසක් අසත්‍යයි. 1 හි අනුපූරකය ලකුණු බිටුව ලෙස වැඩිම වෙසෙසි බිටුව (MSB) භාවිතා කරන අතර (0 ධන සඳහා, 1 සෘණ සඳහා), 2 හි අනුපූරකය වැනි අනෙකුත් පද්ධති සඳහාද මෙය සත්‍ය වේ. "අතිරේක" බිටු වාක්‍ය ඛණ්ඩය මෙහි වැරදියි.

Option / විකල්පය 5
False. Due to the dual representation of zero, 1's complement does not have unique patterns for every value, as both positive zero and negative zero exist.
අසත්‍යයි. ශූන්‍යයේ ද්විත්ව නිරූපණය හේතුවෙන්, ධන ශූන්‍ය සහ සෘණ ශූන්‍ය යන දෙකම පවතින බැවින්, 1 හි අනුපූරකයට සෑම අගයක් සඳහාම අනන්‍ය රටා නොමැත.

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

5. Which of the following answer is equivalent to this Boolean statement?
පහත සඳහන් කුමන පිළිතුර මෙම බූලියානු ප්‍රකාශයට සමානද?

$DA + (\overline{A1} + \overline{CC})D$

1) 1 2) 0 3) $\underline{D}$ 4) $\overline{C}$ 5) D.A

Answer : 3)

$= DA + (\overline{A1} + \overline{CC})D$
$= DA + (\overline{A} + \overline{1} + \overline{C} + \overline{C})D$ De Morgan's Law (ඩ මුවාර් ප්‍රමේයය)
$= DA + (\overline{A} + 0 + \overline{C} + \overline{C})D$ $\overline{1} = 0$
$= DA + (\overline{A} + 0 + \overline{C})D$ Idempotent Law (තද්වභාවී න්‍යාය)
$= DA + (\overline{A} + \overline{C})D$ Identity Law (සර්වසාම්‍ය න්‍යාය)
$= DA + D\overline{A} + D\overline{C}$ Distributive Law (විෂටන න්‍යාය)
$= D(A + \overline{A}) + D\overline{C}$ Distributive Law (විෂටන න්‍යාය)
$= D(1) + D\overline{C}$ Inverse Law (ප්‍රතිලෝම න්‍යාය)
$= D + D\overline{C}$ Identity Law (සර්වසාම්‍ය න්‍යාය)
$= D$ Redundancy Law (සමනිරීක්ත න්‍යාය)

So, the correct answer number is 3).
එමනිසා, නිවැරදි පිළිතුරු අංකය 3) වේ.

6. Which of the following statements about full adders and half adders is true?
පූර්ණ ආකලකය සහ අර්ධ ආකලකය පිළිබඳ පහත සඳහන් කුමන ප්‍රකාශය සත්‍ය ද?

   1) A half adder can add two binary digits and includes a carry in input.
   අර්ධ ආකලකයට ද්විමය ඉලක්කම් දෙකක් එකතු කළ හැකි අතර carry in ආදානයක් ඇතුළත් වේ.

   2) A full adder can add two binary digits but does not include a carry in input.
   පූර්ණ ආකලකයට ද්විමය ඉලක්කම් දෙකක් එකතු කළ හැකි නමුත් carry in ආදානයක් ඇතුළත් නොවේ.

   3) A half adder produces a sum and a carry out output but cannot handle a carry in input.
   අර්ධ ආකලකය එකතුවක් සහ carry out ප්‍රතිදානයක් නිපදවන නමුත් carry in ආදානයක් හැසිරවිය නොහැක.

   4) A full adder can only add two binary digits and cannot handle a carry in input.
   පූර්ණ ආකලකයට ද්විමය ඉලක්කම් දෙකක් පමණක් එකතු කළ හැකි අතර carry in ආදානයක් හැසිරවිය නොහැක.

   5) Both full adders and half adders require three inputs: two significant bits and a carry in bit.
   පූර්ණ ආකලකය සහ අර්ධ ආකලකය යන දෙකටම ආදාන තුනක් අවශ්‍ය වේ. (සැලකිය යුතු බිටු දෙකක් සහ carry in බිට් එකක්)

Answer : 3)

Let's consider each option one by one.
එක් එක් විකල්පය එකින් එක සලකා බලමු.

- **Statement / ප්‍රකාශය 1**

This is incorrect. A half adder adds two binary digits but does not include a carry in input. It only produces a sum and a carry out.
මෙය නිවැරදි නොවේ. අර්ධ ආකලකයක් ද්විමය ඉලක්කම් දෙකක් එකතු කරන නමුත් ආදානයේ රැගෙන යාමක් ඇතුළත් නොවේ. එය නිෂ්පාදනය කරන්නේ එකතුවක් සහ රැගෙන යාමක් පමණි.

- **Statement / ප්‍රකාශය**

This is incorrect. A full adder can add two binary digits and does include a carry in input, which allows it to handle carry-over from previous additions.
මෙය නිවැරදි නොවේ. පූර්ණ ආකලකයකට ද්විමය ඉලක්කම් දෙකක් එකතු කළ හැකි අතර ආදානයේ රැගෙන යාමක් ඇතුළත් වේ, එය පෙර එකතු කිරීම් වලින් රැගෙන යාම හැසිරවීමට ඉඩ සලසයි.

- **Statement / ප්‍රකාශය 3**

This is correct. A half adder takes two binary inputs, produces a sum and a carry out, but does not have a carry in input. It is only suitable for adding two single bits without considering any carry from previous additions.
මෙය නිවැරදි වේ. අර්ධ ආකලකයක් ද්විමය ආදාන දෙකක් ගනී, එකතුවක් සහ ගෙනයාමක් නිපදවයි, නමුත් ආදානයේ රැගෙන යාමක් නොමැත. එය යෝග්‍ය වන්නේ පෙර එකතු කිරීම් වලින් කිසිදු රැගෙන යාමක් නොසලකා තනි බිටු දෙකක් එකතු කිරීම සඳහා පමණි.

- **Statement / ප්‍රකාශය 4**

This is incorrect. A full adder can handle two binary digits and also includes a carry in input, making it capable of handling multi-bit binary addition by taking into account carry-over from previous additions.
මෙය නිවැරදි නොවේ. පූර්ණ ආකලකයකට ද්විමය ඉලක්කම් දෙකක් හැසිරවිය හැකි අතර ආදානයේ රැගෙන යාමක් ද ඇතුළත් වේ, පෙර එකතු කිරීම් වලින් රැගෙන යාම සැලකිල්ලට ගනිමින් බහු-බිටු ද්විමය එකතු කිරීම හැසිරවීමේ හැකියාව එයට ඇත.

- **Statement / ප්‍රකාශය 5**

This is incorrect. While a full adder requires three inputs (two significant bits and a carry in), a half adder only requires two inputs (the two binary digits) and does not have a carry in input.
මෙය නිවැරදි නොවේ. පූර්ණ ආකලකයකට ආදාන තුනක් (සැලකිය යුතු බිටු දෙකක් සහ රැගෙන යාමක්) අවශ්‍ය වන අතර, අර්ධ ආකලකයකට අවශ්‍ය වන්නේ ආදාන දෙකක් (ද්විමය ඉලක්කම් දෙක) පමණක් වන අතර ආදානය තුළ රැගෙන යාමක් නොමැත.

So, the correct answer is indeed **3).**
එබැවින්, නිවැරදි පිළිතුර **3)** වේ.

7. Which operator is overloaded by the `__add__()` method?
`__add__()` ක්‍රමය ලෙස දක්වා ඇත්තේ කුමන මෙහෙයවනයද?

1) `-` 2) `*` 3) `/` 4) `+` 5) `%`

Answer : 4)

Explanation of the options / **විකල්ප පැහැදිලි කිරීම:**

Option / විකල්පය 1)
The `-` operator is overloaded by the `__sub__()` method. This method defines the behavior for subtraction between objects of a class.
`-` මෙහෙයවනය `__sub__()` ක්‍රමය මඟින් දක්වා ඇත. මෙම ක්‍රමය පන්තියක (class) වස්තු අතර අඩුකිරීමේ හැසිරීම නිර්වචනය කරයි.

Example:

```
class Number:
    def __init__(self, value):
        self.value = value
    def __sub__(self, other):
        return self.value - other.value
```

Option / විකල්පය 2)
The `*` operator is overloaded by the `__mul__()` method. This method defines the behavior for multiplication between objects of a class.
`*` ක්‍රියාකරු `__mul__()` ක්‍රමය මඟින් දක්වා ඇත. මෙම ක්‍රමය පන්තියක (class) වස්තූන් අතර ගුණ කිරීම සඳහා හැසිරීම නිර්වචනය කරයි.

Example:

```
class Number:
    def __init__(self, value):
        self.value = value
    def __mul__(self, other):
        return self.value * other.value
```

Option / විකල්පය 3)
The `/` operator is overloaded by the `__truediv__()` method. This method defines the behavior for true division between objects of a class.
`/` ක්‍රියාකරු `__truediv__()` ක්‍රමය මඟින් දක්වා ඇත. මෙම ක්‍රමය පන්තියක (class) වස්තූන් අතර සැබෑ බෙදීම සඳහා හැසිරීම නිර්වචනය කරයි.

Example:

```
class Number:
    def __init__(self, value):
        self.value = value
    def __truediv__(self, other):
        return self.value / other.value
```

Option / විකල්පය 4)
The `+` operator is overloaded by the `__add__()` method. This method defines the behavior for addition between objects of a class.
`__add__()` ක්‍රමය මඟින් `+` ක්‍රියාකරු දක්වා ඇත. මෙම ක්‍රමය පන්තියක (class) වස්තූන් අතර එකතු කිරීම සඳහා හැසිරීම නිර්වචනය කරයි.

Example:

```
class Number:
    def __init__(self, value):
        self.value = value
    def __add__(self, other):
        return self.value + other.value
num1 = Number(10)
num2 = Number(20)
print(num1 + num2)  # Output: 30
```

Option / විකල්පය 5)

The `%` operator is overloaded by the `__mod__()` method. This method defines the behavior for the modulus operation between objects of a class.

`%` ක්‍රියාකරු `__mod__()` ක්‍රමය මඟින් දක්වා ඇත. මෙම ක්‍රමය පන්තියක (class) වස්තු අතර මොඩියුල ක්‍රියාකාරිත්වය සඳහා හැසිරීම නිර්වචනය කරයි.

Example:

```
class Number:
    def __init__(self, value):
        self.value = value
    def __mod__(self, other):
        return self.value % other.value
```

Therefore, the correct answer for the operator that is overloaded by the `__add__()` method in Python is 4) +.

එමනිසා, __add__() ක්‍රමය මඟින් දක්වා ඇති ක්‍රියාකරු සඳහා නිවැරදි පිළිතුර වන්නේ 4) +.

8. What is the output of the following Python code?
පහත පයිතන් කේතයේ ප්‍රතිදානය කුමක්ද?

```
x = [1, 2, 3]
x.append([4, 5])
print(len(x)+24//7)
```

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

Answer : 2)

Let's consider the process of this python code step by step.
මෙම පයිතන් කේතයේ ක්‍රියාවලිය පියවරෙන් පියවර සලකා බලමු.

`x = [1, 2, 3]`

A list `x` is created with three elements, `[1, 2, 3]`.
මූලාංග තුනකින් සමන්විත `x` ලැයිස්තුවක් `[1, 2, 3]` ලෙස සාදනු ලැබේ.

`x.append([4, 5])`

The `.append()` method is used to add a new element to the list `x`. The element added is a list `[4, 5]`, so after this operation, `x` becomes `[1, 2, 3, [4, 5]]`. This means that the list `[4, 5]` is nested within `x` as a single element.

`x` ලැයිස්තුවට නව මූලාංගයක් එක් කිරීමට `.append()` ශ්‍රිතය භාවිතා කරයි. එකතු කරන ලද මූලාංගය `[4, 5]` ලෙස ලැයිස්තුවකි , එබැවින් මෙම මෙහෙයුමෙන් පසුව `x`, `[1, 2, 3, [4, 5]]` බවට පත්වේ. මෙයින් අදහස් කරන්නේ `[4, 5]` ලැයිස්තුව තනි මූලාංගයක් ලෙස `x` තුළ තැන්පත් කර ඇති බවයි.

`print(len(x)+24//7)`

- `len(x)`: The `len()` function returns the number of elements in the list `x`. After the `append()`, `x` contains 4 elements (1, 2, 3, and `[4, 5]`), so `len(x)` returns 4.
  `len()` ශ්‍රිතය `x` ලැයිස්තුවේ ඇති මූලාංග ගණන ලබා දෙයි. `append()`, `x` හි මූලාංග 4 ක් (1, 2, 3, සහ `[4, 5]`), අඩංගු වේ, එබැවින් `len(x)` මගින් 4 ලබා දෙයි.

- `24//7`: This is integer division of 24 by 7, which results in 3 (since 24 ÷ 7 equals 3 with a remainder of 3, but integer division discards the remainder).
  මෙය 24 න් 7 කින් නිබිල බෙදීමකි, එහි ප්‍රතිඵලය 3 වේ (24 ÷ 7, 3 ට සමාන වන අතර 3 ක් ඉතිරි වේ, නමුත් පූර්ණ සංඛ්‍යා බෙදීම ඉතිරිය ඉවතලයි).

- `len(x) + 24//7`: This adds 4 (from `len(x)`) and 3 (from `24//7`), resulting in 7.
  මෙය 4 (`len(x)`) සහ 3 (`24//7`) එකතු කරයි, එහි ප්‍රතිඵලය 7 වේ.

So, the final output of the code is 7.
එබැවින්, කේතයේ අවසාන ප්‍රතිදානය 7 වේ.

The correct answer number is 2).
නිවැරදි පිළිතුරු අංකය 2) වේ.

9. What will be the output of the following code?
පහත පයිතන් කේතයේ ප්‍රතිදානය කුමක් වේවිද?

```
set1 = {1, 3, 5}
set2 = {2, 4, 6}
print(len(set1 + set2))
```

1) 3 2) 6 3) 0 4) 5 5) Error

Answer : 5)

Sets in Python / Python හි Sets:
Sets are unordered collections of unique elements. They are defined using curly braces {}. Operations that can be performed on sets include union, intersection, difference, etc., but not direct addition.
Sets යනු unique elements වල unordered collection එකකි. ඒවා curly braces {} භාවිතයෙන් අර්ථ දක්වා ඇත. Sets මත සිදු කළ හැකි මෙහෙයුම් වලට union, intersection, difference ආදිය ඇතුළත් වේ, නමුත් direct addition කිරීමක් නොවේ.

The + operator with sets / Sets සහිත + ක්‍රියාකරු (Operator):
The + operator is not defined for sets in Python. Attempting to use the + operator with sets will raise a `TypeError`.
Python හි sets සඳහා + මෙහෙයවනය (operator) අර්ථ දක්වා නැත. + මෙහෙයවනය (operator) sets සමඟ භාවිතා කිරීමට උත්සාහ කිරීම `TypeError` එකක් මතු කරයි.

When the code `print(len(set1 + set2))` is executed, Python will raise a '`TypeError`' because you cannot use the '`+`' operator to concatenate or add sets. The correct way to combine sets would be to use the union method or the '`|`' operator.
කේතය `print(len(set1 + set2))` ක්‍රියාත්මක කළ විට, ඔබට sets එකතු කිරීමට හෝ එකතු කිරීමට '+' මෙහෙයවනය භාවිත කළ නොහැකි නිසා '`TypeError`' එකක් මතු කරයි. sets ඒකාබද්ධ කිරීමේ නිවැරදි ක්‍රමය වනුයේ යුනියන් ක්‍රමය හෝ '`|`' මෙහෙයවනය (operator) භාවිතා කිරීමයි.

Therefore, the correct answer for the output of the given code is 5) Error.
ලබා දී ඇති කේතයේ ප්‍රතිදානය සඳහා නිවැරදි පිළිතුර වන්නේ 5) Error.

Correct way to combine sets / Sets ඒකාබද්ධ කිරීමට නිවැරදි මාර්ගය:
If you wanted to combine the sets and then find the length of the combined set, you could do:
ඔබට sets ඒකාබද්ධ කර ඒකාබද්ධ set එකේ දිග සොයා ගැනීමට අවශ්‍ය නම්, ඔබට කළ හැක්කේ:

```
set1 = {1, 3, 5}
set2 = {2, 4, 6}
combined_set = set1 | set2  # or set1.union(set2)
print(len(combined_set))  # This would output 6
```

10. What is the output of the following Python code?
පහත පයිතන් කේතයේ ප්‍රතිදානය කුමක් ද?

```
x = 5
y = 3
z = 4
if x > y:
    if y > z:
        result = "A"
    elif x > z:
        result = "B"
    else:
        result = "C"
elif y > x:
    if z > x:
        result = "D"
    else:
        result = "E"
else:
    result = "F"
print(result)
```

1) A 2) B 3) C 4) D 5) E

Answer : 2)

The purpose of this code is to demonstrate how conditional logic using `if-elif-else` statements works, particularly with nested conditions.
මෙම කේතයේ පරමාර්ථය වන්නේ `if-elif-else` ප්‍රකාශයන් භාවිතා කරමින් කොන්දේසි සහිත තර්කනය ක්‍රියා කරන ආකාරය, විශේෂයෙන්ම nested conditions සමඟ නිරූපණය කිරීමයි.

The following table can further explain what happened in this code.
මෙම කේතයේ සිදුවූයේ කුමක්ද යන්න පහත වගුවෙන් තවදුරටත් පැහැදිලි කළ හැක.

Step	x	y	z	Condition	Result
1	5	3	4	x > y (True)	
2	5	3	4	y > z (False)	
3	5	3	4	x > z (True)	result = "B"
4	5	3	4		Output: B

Accordingly, the output is "B". So, the answer number is 2.
ඒ අනුව, ප්‍රතිදානය වන්නේ "B" වේ. එමනිසා, පිළිතුරු අංකය 2 වේ.