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 231 MARKING

2025/07/29
ictfromabc
Ravindu Bandaranayake

1. Which output device produces a hard copy output by spraying tiny ink droplets onto paper?
කඩදාසි මත කුඩා තින්ත බිංදු ඉසිමෙන් දෘඩ පිටපත් ප්‍රතිදානයක් නිපදවන ප්‍රතිදාන උපාංගය කුමක්ද?

1) Laser printer
2) Dot matrix printer
3) Thermal printer
4) Inkjet printer
5) 3D printer

Answer 4)

Explanation/ පැහැදිලි කිරීම

An inkjet printer works by propelling very small droplets of liquid ink directly onto paper to create an image or text. It is known for its ability to produce high-quality color prints and is commonly used for home and office printing. The droplets are so tiny that they can create detailed images with smooth color transitions.
Inkjet මුද්‍රණ යන්ත්‍රයක් ක්‍රියා කරන්නේ රූපයක් හෝ පාඨ නිර්මාණය කිරීම සඳහා ඉතා කුඩා ද්‍රව තින්ත බිංදු කෙලින්ම කඩදාසි මතට තල්ලු කිරීමෙනි. එය උසස් තත්ත්වයේ වර්ණ මුද්‍රණ නිපදවීමේ හැකියාව සඳහා ප්‍රසිද්ධ වන අතර එය නිවාස සහ කාර්යාල මුද්‍රණය සඳහා බහුලව භාවිතා වේ. බිංදු ඉතා කුඩා බැවින් ඒවාට සුමට වර්ණ සංක්‍රාන්ති සහිත සවිස්තරාත්මක රූප නිර්මාණය කළ හැකිය.

Other options differ in their printing mechanisms:
වෙනත් විකල්ප ඒවායේ මුද්‍රණ යාන්ත්‍රණයන්හි වෙනස් වේ:

- **Option/ විකල්ප 1)**

Laser printers use a laser beam and toner powder, not ink droplets, to produce prints. They are faster and more suited for high-volume printing but use a different technology.
ලේසර් මුද්‍රණ යන්ත්‍ර මුද්‍රණ නිපදවීම සඳහා ලේසර් කදම්භයක් සහ ටෝනර් කුඩු භාවිතා කරයි, තින්ත බිංදු නොවේ. ඒවා ඉහළ පරිමාවකින් යුත් මුද්‍රණය සඳහා වේගවත් හා වඩාත් සුදුසු නමුත් වෙනස් තාක්ෂණයක් භාවිතා කරයි.

- **Option/ විකල්ප 2)**

Dot matrix printers produce characters and images by striking an ink-soaked ribbon with tiny pins, creating dots on paper. This is an impact printing method, quite different from inkjet spraying.
Dot matrix මුද්‍රණ යන්ත්‍ර කුඩා අල්පෙනෙති සහිත තින්ත පොඟවා ගත් පීත්ත පටියකට පහර දීමෙන් අක්ෂර සහ රූප නිපදවයි, කඩදාසි මත තිත් නිර්මාණය කරයි. මෙය inkjet ඉසීමට වඩා බෙහෙවින් වෙනස් බලපෑම් මුද්‍රණ ක්‍රමයකි.

- **Option/ විකල්ප 3)**

Thermal printers use heat to produce images on heat-sensitive paper; they do not use ink droplets.
තාප මුද්‍රණ යන්ත්‍ර තාප සංවේදී කඩදාසි මත රූප නිපදවීමට තාපය භාවිතා කරයි ඔවුන් තින්ත බිංදු භාවිතා නොකරයි.

- **Option/ විකල්ප 5)**

3D printers create three-dimensional objects layer by layer using materials like plastic, not paper prints.
ත්‍රිමාණ මුද්‍රණ යන්ත්‍ර කඩදාසි මුද්‍රණ නොව ප්ලාස්ටික් වැනි ද්‍රව්‍ය භාවිතයෙන් ස්ථරයෙන් ස්ථරයට ත්‍රිමාණ වස්තූන් නිර්මාණය කරයි.

Therefore, the inkjet printer is the correct answer because it specifically sprays tiny ink droplets to produce the hard copy output.
එබැවින්, සබනවැළු මුද්‍රණ යන්ත්‍රය නිවැරදි පිළිතුර වන්නේ එය දෘඩ පිටපත් ප්‍රතිදානය නිපදවීම සඳහා විශේෂයෙන් කුඩා තින්ත බිංදු ඉසින බැවිනි.

Therefore the answer is 4) / එබැවින් පිළිතුර 4) වේ

2. Which of the following statements about the history of computing is correct?
පරිගණකකරණයේ ඉතිහාසය පිළිබඳ පහත සඳහන් ප්‍රකාශවලින් නිවැරදි කුමක්ද?

1) ENIAC – First transistor-based general-purpose computer
ENIAC – පළමු ට්‍රාන්සිස්ටරය මත පදනම් වූ පොදු කාර්ය පරිගණකය
2) Jacquard Loom – Introduced punched cards for automation
ජැකාර්ඩ් ලූම් - ස්වයංක්‍රියකරණය සඳහා සිදුරු කළ කාඩ්පත් හඳුන්වා දෙන ලදී
3) IBM 1401 – First mechanical calculator used in education
IBM 1401 – අධ්‍යාපනයේ භාවිතා කරන ලද පළමු යාන්ත්‍රික කැල්කියුලේටරය
4) Pascaline – First programmable mechanical computer
පැස්කලිනය - පළමු වැඩසටහන්ගත කළ හැකි යාන්ත්‍රික පරිගණකය
5) UNIVAC I – First computer to use vacuum tubes
UNIVAC I – රික්තක නල භාවිතා කළ පළමු පරිගණකය

Answer 2)

Explanation/ පැහැදිලි කිරීම

- **Option/ විකල්ප 1)**

ENIAC was a electronic computer built in the 1940s that used thousands of vacuum tubes to perform calculations. However, it was not transistor-based; transistors were invented later and replaced vacuum tubes in computers for better efficiency and reliability.
ENIAC යනු 1940 ගණන්වල ඉදිකරන ලද ඉලෙක්ට්‍රොනික පරිගණකයක් වන අතර එය ගණනය කිරීම් සිදු කිරීම සඳහා දහස් ගණනක් රික්ත නල භාවිතා කළේය. කෙසේ වෙතත්, එය ට්‍රාන්සිස්ටර මත පදනම් වූවක් නොවේ ට්‍රාන්සිස්ටර පසුව සොයා ගන්නා ලද අතර වඩා හොඳ කාර්යක්ෂමතාව සහ විශ්වසනීයත්වය සඳහා පරිගණකවල රික්ත නල ප්‍රතිස්ථාපනය කරන ලදී.

- **Option/ විකල්ප 2)**

The Jacquard Loom, created in the early 19th century, was a mechanical loom that used punched cards to control weaving patterns automatically. This concept of using punched cards directly influenced early computing machines, marking an important step towards programmable devices.
19 වන සියවසේ මුල් භාගයේදී නිර්මාණය කරන ලද ජැකාර්ඩ් ලූම්, රෙදි විවිම රටා ස්වයංක්‍රියව පාලනය කිරීම සඳහා සිදුරු කරන ලද කාඩ්පත් භාවිතා කරන යාන්ත්‍රික රෙදි වියනක් විය. සිදුරු කරන ලද කාඩ්පත් භාවිතා කිරීමේ මෙම සංකල්පය මුල් පරිගණක යන්ත්‍රවලට සෘජුවම බලපෑ අතර එය වැඩසටහන්ගත කළ හැකි උපාංග සඳහා වැදගත් පියවරක් සනිටුහන් කළේය.

- **Option/ විකල්ප 3)**

The IBM 1401 was an early digital computer introduced in the late 1950s, primarily used for business data processing, not as a mechanical calculator or for educational purposes.
IBM 1401 යනු 1950 ගණන්වල අගභාගයේදී හඳුන්වා දුන් මුල් ඩිජිටල් පරිගණකයක් වන අතර එය ප්‍රධාන වශයෙන් ව්‍යාපාරික දත්ත සැකසීම සඳහා භාවිතා කරන ලදී, යාන්ත්‍රික කැල්කියුලේටරයක් ලෙස හෝ අධ්‍යාපනික අරමුණු සඳහා නොවේ.

- **Option/ විකල්ප 4)**

Pascaline, invented by Blaise Pascal in the 17th century, was one of the first mechanical calculators designed to add and subtract numbers. However, it was not programmable like later computers.
17 වන සියවසේදී බ්ලේස් පැස්කල් විසින් සොයා ගන්නා ලද පැස්කලයින්, සංඛ්‍යා එකතු කිරීම සහ අඩු කිරීම සඳහා නිර්මාණය කරන ලද පළමු යාන්ත්‍රික කැල්කියුලේටර වලින් එකකි. කෙසේ වෙතත්, එය පසුකාලීන පරිගණක මෙන් වැඩසටහන්ගත කළ නොහැකි විය.

- **Option/ විකල්ප 5)**

UNIVAC I was the first commercially available computer in the United States, using vacuum tubes for processing. While it played a big role in computing history, ENIAC was actually the first to use vacuum tubes, so UNIVAC I was not the first.
UNIVAC I යනු එක්සත් ජනපදයේ රික්තක නල භාවිතා කරමින් සැකසීම සඳහා වාණිජමය වශයෙන් ලබා ගත හැකි පළමු පරිගණකයයි. පරිගණක ඉතිහාසයේ එය විශාල කාර්යභාරයක් ඉටු කළද, ඇත්ත වශයෙන්ම රික්තක නල පළමුවෙන්ම භාවිතා කළේ ENIAC වන බැවින් UNIVAC I පළමුවැන්නා නොවේ.

Therefore the answer is 2) / එබැවින් පිළිතුර 2) වේ.

3. What happens when you add two numbers in two's complement representation and the result produces a carry-out of the most significant bit?
ඔබ දෙකේ අනුපූරක නිරූපණයට සංඛ්‍යා දෙකක් එකතු කළ විට සහ ප්‍රතිඵලය වැඩිම වෙසෙසි බිටුවෙන් ගෙනයාමක් නිපදවන විට කුමක් සිදුවේ ද?

1) The carry bit affects the sum / රැගෙන යන බිටුව එකතුවට බලපායි.
2) The carry bit is added back to the sum / රැගෙන යන බිටු එකතුවට නැවත එකතු වේ.
3) The carry bit is inverted / රැගෙන යන බිටුව ප්‍රතිලෝම කර ඇත.
4) The carry bit is stored separately / රැගෙන යාමේ බිටුව වෙන වෙනම ගබඩා කර ඇත.
5) The carry bit is discarded / රැගෙන යාමේ බිටුව ඉවත දමනු ලැබේ.

Answer : 5)

When performing addition using two's complement representation, the carry-out of the most significant bit (MSB) is an important indicator but does not directly affect the sum. Let's explore the meaning behind the carry-out and the correct answer:
දෙකෙහි අනුපූරක නිරූපණය භාවිතයෙන් එකතු කිරීම සිදු කරන විට, most significant bit (MSB) රැගෙන යාම වැදගත් දර්ශකයක් වන නමුත් එකතුවට සෘජුවම බලපාන්නේ නැත. රැගෙන යාම පිටුපස ඇති අර්ථය සහ නිවැරදි පිළිතුර ගවේෂණය කරමු:

Two's complement is a method used to represent positive and negative integers in binary. When adding two numbers in this representation, the carry-out of the MSB can indicate whether an overflow has occurred, but the carry itself is discarded, and it does not affect the resulting sum.
දෙකෙහි අනුපූරකය යනු ද්විමය තුළ ධන සහ සෘණ නිඛිල නියෝජනය කිරීමට භාවිතා කරන ක්‍රමයකි. මෙම නිරූපණයෙහි සංඛ්‍යා දෙකක් එකතු කරන විට, MSB රැගෙන යාමෙන් පිටාර ගැලීමක් සිදුවී ඇත්ද යන්න දැක්විය හැක, නමුත් රැගෙන යාමම ඉවත දමනු ලබන අතර, එය ප්‍රතිඵලයක් ලෙස ලැබෙන එකතුවට බලපාන්නේ නැත.

So, the correct answer 5) The carry bit is discarded.
එබැවින්, නිවැරදි පිළිතුර 5) රැගෙන යාමේ බිටි ඉවත දමනු ලැබේ.

In two's complement arithmetic, the carry bit that results from the most significant bit (MSB) is discarded. This is because the sum already reflects the correct result in two's complement form, and the carry-out is only used to detect overflow, not to adjust the sum itself.
දෙකෙහි අනුපූරක අංක ගණිතයේදී, most significant bit (MSB) වලින් ලැබෙන කැරි බිටි ඉවත දමනු ලැබේ. මක්නිසාද යත්, එකතුවෙන් දැනටමත් නිවැරදි ප්‍රතිඵලය දෙකෙහි අනුපූරක ආකාරයෙන් පිළිබිඹු වන අතර, රැගෙන යාම භාවිතා කරන්නේ පිටාර ගැලීම හඳුනා ගැනීමට මිස එකතුව සකස් කිරීමට නොවේ.

Other options / අනෙකුත් විකල්ප:

Option / විකල්පය 1)
Incorrect because the carry bit does not alter the sum in two's complement; it's simply discarded.
රැගෙන යාමේ බිටු දෙකෙහි අනුපූරකයේ එකතුව වෙනස් නොකරන නිසා වැරදි වේ. එය සරලව ඉවත දමනු ලැබේ.

Option / විකල්පය 2)
Incorrect. Unlike in some other binary arithmetic methods (like unsigned binary addition), in two's complement, the carry-out from the MSB is not added back.
වැරදි වේ. වෙනත් සමහර ද්විමය අංක ගණිත ක්‍රමවල මෙන් නොව (අත්සන් නොකළ ද්විමය එකතු කිරීම් වැනි), දෙකෙහි අනුපූරකයේදී, MSB වෙතින් ගෙනයාම නැවත එකතු නොවේ.

Option / විකල්පය 3)
Incorrect. The carry bit is not inverted; it's discarded after the addition process.
වැරදි වේ. රැගෙන යාමේ බිට් ප්‍රතිලෝම නොවේ. එකතු කිරීමේ ක්‍රියාවලියෙන් පසු එය ඉවතලනු ලැබේ.

Option / විකල්පය 4)
Incorrect. In two's complement arithmetic, there's no need to store the carry bit separately since it does not impact the final sum.
වැරදි වේ. දෙකෙහි අනුපූරක අංක ගණිතයේදී, එය අවසන් එකතුවට බලපාන්නේ නැති නිසා රැගෙන යාමේ බිට් එක වෙනම ගබඩා කිරීමට අවශ්‍ය නොවේ.

4. The IEEE 754 32-bit representation of the decimal number -6.5 has which of the following as its sign bit and exponent?
-6.5 දශම සංඛ්‍යාවේ IEEE 754 32-bit නිරූපණයෙහි පහත සඳහන් ඒවායින් ලකුණ බිට් සහ ඝාතකය ලෙස ඇත්තේ කුමක්ද?

1) Sign = 1, Exponent = 10000001
2) Sign = 0, Exponent = 10000001
3) Sign = 1, Exponent = 10000010
4) Sign = 1, Exponent = 01111110
5) Sign = 0, Exponent = 10000010

Answer 1

Explanation/ පැහැදිලි කිරීම

IEEE 754 is a standard way to store decimal numbers in binary. In the 32-bit format, the number is split into three parts.
IEEE 754 යනු ද්විමය ආකාරයෙන් දශම සංඛ්‍යා ගබඩා කිරීමේ සම්මත ක්‍රමයකි. 32-බිට් ආකෘතියෙන්, අංකය කොටස් තුනකට බෙදා ඇත.

Part	**Size (bits)**
Sign	1
Exponent	8
Mantissa	23

The sign bit simply indicates whether the number is positive or negative. A **0** means the number is **positive**, while a **1** indicates it is **negative.**
ලකුණු බිට් එක සරලව දක්වන්නේ සංඛ්‍යාව ධන ද සෘණ ද යන්නයි. A **0** යනු සංඛ්‍යාව **ධන** වන අතර, **1** යනු එය **සෘණ** බව දක්වයි.

In the case of -6.5, the number is negative, so the sign bit is 1.
-6.5 අවස්ථාවෙහිදී, සංඛ්‍යාව සෘණ වන බැවින්, ලකුණු බිටුව 1 වේ.

Next, we need to convert the absolute value of the number (6.5) into binary.
ඊළඟට, අපි සංඛ්‍යාවේ නිරපේක්ෂ අගය (6.5) ද්විමය බවට පරිවර්තනය කළ යුතුයි.

Base 10	6.5					
	4	2	1	1	2	4
	1	1	0	1	0	1
Base 2	110.101_2					

Now normalize the binary number/ දැන් ද්විමය අංකය සාමාන්‍යකරණය කරන්න

NOTE

Normalization means adjusting a binary number so that it's written in a standard format:
This is just like scientific notation in decimal, where we write numbers like:
සාමාන්‍යකරණය යනු ද්විමය සංඛ්‍යාවක් සම්මත ආකෘතියකින් ලිවීමට සකස් කිරීමයි:
මෙය දශමයේ විද්‍යාත්මක අංකනයට සමානයි, එහිදී අපි සංඛ්‍යා ලියන්නේ

Ex:-5,200 = 5.2×10^3

$110.1 = 1.101 \times 2^2$.

Now we compute the biased exponent. IEEE 754 uses a bias of **127** for the exponent in 32-bit format.
Since the actual exponent here is 2, we add the bias to it
දැන් අපි exponent **ගණනය කරමු.** IEEE 754 32-bit **ආකෘතියෙන්** exponent **සඳහා** 127 **ක**
පක්ෂග්‍රාහිත්වයක් භාවිතා කරයි. මෙහි සැබෑ ඝාතකය 2 **වන බැවින්, අපි එයට පක්ෂග්‍රාහිත්වය එකතු කරමු.**

Biased Exponent = 2+127 = 129

We then convert this decimal value (129) into binary
ඉන්පසු අපි මෙම දශම අගය 129 **ද්විමය බවට පරිවර්තනය කරමු.**

Base 10	129							
	128	64	32	16	8	4	2	1
	1	0	0	0	0	0	0	1
Base 2	10000001_2							

So, the 8-bit exponent field becomes 10000001.
ඉතින්, 8-**බිට් ඝාතක ක්ෂේත්‍රය** 10000001 **බවට පත්වේ.**

Therefore the answer is 1)/ **එබැවින් පිළිතුර** 1) **වේ**

5. Which of the following logic expressions simplifies to $A \oplus B$?
පහත සඳහන් තාර්කික ප්‍රකාශනවලින් $A \oplus B$ **දක්වා සරල කරන්නේ කුමක්ද?**

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

Answer: 1)

Explanation/ **පැහැදිලි කිරීම**

XOR (exclusive OR) outputs 1 only when exactly one input is 1 and the other is 0.
XOR (exclusive OR) 1 **ප්‍රතිදානය කරන්නේ හරියටම එක් ආදානයක්** 1 **වන විට සහ අනෙක** 0 **වන විට පමණි.**
- A = 0, B = 0 → Output = 0
- A = 0, B = 1 → Output = 1
- A = 1, B = 0 → Output = 1
- A = 1, B = 1 → Output = 0

So XOR is true only when inputs are different.
එබැවින් XOR **සත්‍ය වන්නේ ආදාන වෙනස් වූ විට පමණි.**

Breaking down Option 1 / **විකල්ප** 1 **බිඳ දැමීම:**

A	B	$A.\overline{B}$	$\overline{A}.B$	$A.\overline{B}+\overline{A}.B$
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

So, $A.\overline{B}+\overline{A}.B$ is the standard Boolean form of the XOR operation, that's why it's correct.
එබැවින්, $A.\overline{B}+\overline{A}.B$ යනු XOR මෙහෙයුමේ සම්මත බූලියන් ආකාරයයි, ඒ නිසා එය නිවැරදියි.

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

6. $A.\overline{B} \oplus A.B.\overline{C}$

Solve this boolean expression.
මෙම බූලියානු ප්‍රකාශනය සුළු කරන්න.

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

Answer : 4)

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

$A\overline{B} \oplus AB\overline{C}$

$A\overline{B}.\overline{AB\overline{C}}+A\overline{B}.\overline{AB\overline{C}}$	$A \oplus B = \overline{A}B+A\overline{B}$
$(\overline{A}+\overline{\overline{B}}).AB\overline{C}+A\overline{B}.\overline{AB\overline{C}}$	De Morgan's law ද මුවාර් ප්‍රමේයය
$(\overline{A}+B).AB\overline{C}+A\overline{B}.\overline{AB\overline{C}}$	Double inverse law ද්විත්ව ප්‍රතිලෝම න්‍යාය
$(\overline{A}+B).AB\overline{C}+A\overline{B}.(\overline{A}+\overline{B}+\overline{\overline{C}})$	De Morgan's law ද මුවාර් ප්‍රමේයය
$(\overline{A}+B).AB\overline{C}+A\overline{B}.(\overline{A}+\overline{B}+C)$	Double inverse law ද්විත්ව ප්‍රතිලෝම න්‍යාය
$\overline{A}AB\overline{C}+BAB\overline{C}+A\overline{B}\overline{A}+A\overline{B}\overline{B}+A\overline{B}C$	Distributive law විෂටන න්‍යාය
$0B\overline{C}+BAB\overline{C}+A\overline{B}\overline{A}+A\overline{B}\overline{B}+A\overline{B}C$	Inverse law ප්‍රතිලෝම න්‍යාය
$0+BAB\overline{C}+A\overline{B}\overline{A}+A\overline{B}\overline{B}+A\overline{B}C$	Identity law සර්වසාම්‍ය න්‍යාය
$BAB\overline{C}+A\overline{B}\overline{A}+A\overline{B}\overline{B}+A\overline{B}C$	Identity law සර්වසාම්‍ය න්‍යාය
$AB\overline{C}+A\overline{B}\overline{A}+A\overline{B}\overline{B}+A\overline{B}C$	Idempotent law තද්දේවභාවී න්‍යාය
$AB\overline{C}+\overline{B}0+A\overline{B}\overline{B}+A\overline{B}C$	Inverse law ප්‍රතිලෝම න්‍යාය
$AB\overline{C}+0+A\overline{B}\overline{B}+A\overline{B}C$	Identity law සර්වසාම්‍ය න්‍යාය
$AB\overline{C}+A\overline{B}\overline{B}+A\overline{B}C$	Identity law සර්වසාම්‍ය න්‍යාය
$AB\overline{C}+A\overline{B}+A\overline{B}C$	Idempotent law තද්දේවභාවී න්‍යාය
$AB\overline{C}+A\overline{B}$	Redundancy law සමතිරික්ත න්‍යාය
$A(B\overline{C}+\overline{B})$	Distributive law විෂටන න්‍යාය
$A(\overline{C}+\overline{B})$	Redundancy law සමතිරික්ත න්‍යාය
$A\overline{C}+A\overline{B}$	Distributive law විෂටන න්‍යාය

So, the correct answer number here is 4).
එමනිසා, මෙහි නිවැරදි පිළිතුරු අංකය 4) වේ.

7. How many NAND gates are required to implement only the Sum output of a Full Adder?
Full Adder එකතුව ප්‍රතිදානය පමණක් ක්‍රියාත්මක කිරීමට NAND ද්වාර කියක් අවශ්‍ය වේද?

1) 4 2) 8 3) 9 4) 10 5) 12

Answer: 4)

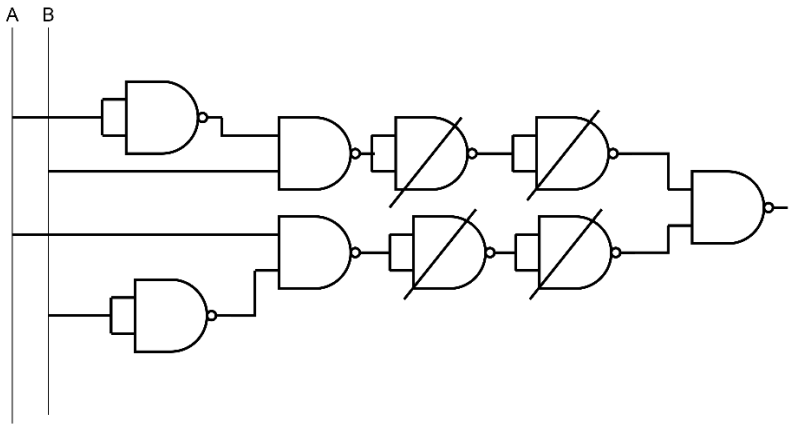
Explanation / පැහැදිලි කිරීම:

The Sum output of a full adder is given by/ එකතුව ප්‍රතිදානය ලබා දෙන්නේ:
Sum = A ⊕ B ⊕ Cin

This means the sum is the XOR of three inputs, which we usually break down into two XOR operations.
මෙහි තේරුම එකතුව යනු ආදාන තුනක XOR වන අතර, එය අපි සාමාන්‍යයෙන් XOR මෙහෙයුම් දෙකකට බෙදා දෙමු.

First - A⊕BA
Then - (A⊕B) ⊕ Cin

How many NAND gates are required to implement the XOR function for inputs A and B?
A සහ B ආදාන සඳහා XOR ශ්‍රිතය ක්‍රියාත්මක කිරීමට NAND ද්වාර කියක් අවශ්‍ය වේද?



Since the sum requires two XOR gates connected in series,
එකතුවට ශ්‍රේණිගතව සම්බන්ධ කර ඇති XOR ද්වාර දෙකක් අවශ්‍ය වන බැවින්,

The first XOR (A⊕B) uses 5 NAND gates.
පළමු (A⊕B), NAND ද්වාර 5ක් භාවිතා කරයි.

The second XOR (result of first XOR ⊕) uses another 5 NAND gates.
දෙවන XOR (result of first XOR ⊕) තවත් NAND ද්වාර 5ක් භාවිතා කරයි.

Adding them together gives a total of / ඒවා එකට එකතු කළ විට ලැබෙන්නේ:
5 + 5 = 10 NAND gates.

Therefore, the answer is 4).
එබැවින්, පිළිතුර 4) වේ.

8. What will be the output of this code?
මෙම කේතයේ ප්‍රතිදානය කුමක් වේද?

```
x = 10
if x > 5:
    if x < 15:
        print("A")
    else:
        print("B")
else:
    print("C")
```

1) A 2) B 3) C 4) A C 5) Error

Answer 1)

Explanation/ **පැහැදිලි කිරීම**

- **`x = 10`**

This line assigns the value `10` to the variable `x`. Wherever `x` appears, it means the number `10`.
මෙම පේළිය `x` විචල්‍යයට `10` අගය පවරයි. `x` දිස්වන ඕනෑම තැනක, එයින් අදහස් වන්නේ `10` අංකයයි.

- **`if x > 5:`**

Here, we check if the value of `x` (which is 10) is greater than `5`. Since `10 > 5` is True, the program now enters this if block and checks the next condition nested inside it.
මෙහි, අපි `x` හි අගය (එය 10) `5` ට වඩා වැඩි දැයි පරීක්ෂා කරමු. `10 > 5` සත්‍ය බැවින්, වැඩසටහන දැන් මෙම if බ්ලොක් එකට ඇතුළු වී එය තුළ ඇති ඊළඟ කොන්දේසිය පරීක්ෂා කරයි.

- **`if x < 15:`**

Inside the previous if, we now check whether `x` is less than `15`. Since `10 < 15` is also True, this block will now execute the code inside it.
පෙර if ඇතුළත, අපි දැන් `x 15` ට වඩා අඩු දැයි පරීක්ෂා කරමු. `10 < 15` ද සත්‍ය බැවින්, මෙම බ්ලොක් එක දැන් එය තුළ කේතය ක්‍රියාත්මක කරයි.

- **`print("A")`**

Since both conditions are true, the message "A" will be printed to the screen.
කොන්දේසි දෙකම සත්‍ය බැවින්, "A" පණිවිඩය තිරයට මුද්‍රණය කෙරේ.

Therefore the answer is 1) / **එබැවින් පිළිතුර 1) වේ.**

9. Which of the following statements about tuples in Python is false?
පයිතන් හි ටපල් පිළිබඳ පහත සඳහන් ප්‍රකාශවලින් අසත්‍ය කුමක්ද?

1) Tuples are immutable data structures / **ටපල් යනු වෙනස් කළ නොහැකි දත්ත ව්‍යුහයන් වේ.**
2) Tuples can contain mutable objects like lists / **ටපල් වලට ලැයිස්තු වැනි විකෘති වස්තු අඩංගු විය හැකිය.**
3) Tuples support item assignment to change elements.
මූලාංග වෙනස් කිරීම සඳහා අයිතම පැවරුමට ටපල් සහාය දක්වයි.
4) Tuples support indexing and slicing operations
ටපල් සුවිගත කිරීම සහ slicing මෙහෙයුම් සඳහා සහාය දක්වයි.
5) Tuples can be used as keys in dictionaries / Dictionary **යතුරු ලෙස ටපල් භාවිතා කළ හැකිය.**

Answer 3)

Explanation/ **පැහැදිලි කිරීම**

- **Option/ විකල්ප 1)**

This means that once a tuple is created, you cannot modify, add, or remove items from it. For example, if you have `t = (1, 2, 3)`, you cannot do `t[0] = 10`. Attempting this will raise a `TypeError`.
මෙයින් අදහස් කරන්නේ ටපල් එකක් නිර්මාණය කළ පසු, ඔබට එයින් අයිතම වෙනස් කිරීමට, එකතු කිරීමට හෝ ඉවත් කිරීමට නොහැකි බවයි. උදාහරණයක් ලෙස, ඔබට `t = (1, 2, 3)` තිබේ නම්, ඔබට `t[0] = 10` කළ නොහැක. මෙය උත්සාහ කිරීමෙන් `TypeError` එකක් මතු වේ.

- **Option/ විකල්ප 2)**

Even though the tuple itself is immutable, it can contain elements like lists, which are mutable.
Example: `t = ([1, 2], 3)` – Here, you can do `t[0].append(4)` and it becomes `([1, 2, 4], 3)`.
ටපල් එක වෙනස් කළ නොහැකි වුවද, එහි ලැයිස්තු වැනි මූලාංග අඩංගු විය හැකි අතර ඒවා mutable වේ.
උදාහරණය: `t = ([1, 2], 3)` - මෙහිදී, ඔබට `t[0].append(4)` කළ හැකි අතර එය `([1, 2, 4], 3)` බවට පත් වේ.

So while you can't change the tuple structure, you can change the content inside a mutable element.
එබැවින් ඔබට ටපල් ව්‍යුහය වෙනස් කළ නොහැකි වුවද, ඔබට mutable මූලාංගයක් තුළ අන්තර්ගතය වෙනස් කළ හැකිය.

- **Option/ විකල්ප 3)**

This is the wrong statement. Since tuples are immutable, you cannot reassign any of its elements.
මෙය වැරදි ප්‍රකාශයකි. ටපල් වෙනස් කළ නොහැකි බැවින්, ඔබට එහි කිසිදු මූලාංගයක් නැවත පැවරිය නොහැක.

- **Option/ විකල්ප 4)**

You can access elements in a tuple just like a list:
ඔබට ලැයිස්තුවක් මෙන් ටපල් එකක මූලාංග වෙත ප්‍රවේශ විය හැකිය:

Example:
`t = (10, 20, 30, 40)`
`t[2]` gives 30 and `t[1:3]` gives (20, 30)
`t[2]` මගින් 30 ලබා දෙන අතර `t[1:3]` මගින් (20, 30) ලබා දෙයි.

- **Option/ විකල්ප 5)**

Tuples are hashable as long as they don’t contain unhashable (mutable) items like lists. That’s why they are valid as keys in dictionaries:
ලැයිස්තු වැනි හැෂ් කළ නොහැකි (වෙනස් කළ හැකි) අයිතම අඩංගු නොවන තාක් කල් ටපල් හැෂ් කළ හැකිය. ඒ නිසා ඒවා dictionary යතුරු ලෙස වලංගු වේ:
`d = { (1, 2): "value" }`

Therefore the answer is 3) / එබැවින් පිළිතුර 3) වේ.

10. What is the output of this code?
මෙම කේතයේ ප්‍රතිදානය කුමක් ද?

```
for i in range(1, 6):
    print('{:<10}'.format(i * '( ʘ ‿ ʘ)  '))
```

1) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)

2) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)

3) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)

4) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)

5) (ʘ ‿ ʘ)
(ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)

Answer : 1)

The provided Python code generates a pattern of repeated text, which increases with each iteration of the loop. Specifically, the loop runs five times, and in each iteration, the string `'( ʘ ‿ ʘ)'` is repeated a number of times corresponding to the current loop index, i. The `'{:<10}'` format ensures the output is left-aligned. As a result, the output pattern forms a pyramid-like structure where each line contains an increasing number of `'( ʘ ‿ ʘ)'` strings, from one instance in the first line to five instances in the last line.
සපයා ඇති පයිතන් කේතය පුනරාවර්තන පෙළ රටාවක් ජනනය කරයි, එය ලූපයේ එක් එක් පුනරාවර්තනය සමඟ වැඩි වේ. විශේෂයෙන්ම, ලූපය පස් වතාවක් ධාවනය වන අතර, එක් එක් පුනරාවර්තනය තුළ, `'( ʘ ‿ ʘ)'` තන්තුව වත්මන් ලූප දර්ශකයට අනුරූප වන වාර ගණනක් පුනරාවර්තනය වේ, `'{:<10}'` ආකෘතිය මඟින් ප්‍රතිදානය වමට පෙළගස්වා ඇති බව සහතික කරයි. ප්‍රතිඵලයක් වශයෙන්, ප්‍රතිදාන රටාව පිරමිඩ වැනි ව්‍යුහයක් සාදයි, එහිදී එක් එක් පේළියේ `'( ʘ ‿ ʘ)'` තන්තු වැඩි ගණනක් අඩංගු වේ, පළමු පේළියේ එක් අවස්ථාවක සිට අවසාන පේළියේ අවස්ථා පහක් දක්වා.

Iteration (`i`)	Output
1	(ʘ ‿ ʘ)
2	(ʘ ‿ ʘ) (ʘ ‿ ʘ)
3	(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
4	(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)
5	(ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ) (ʘ ‿ ʘ)

The correct option that matches this output structure is Option 1.
මෙම ප්‍රතිදාන ව්‍යුහයට ගැලපෙන නිවැරදි විකල්පය වන්නේ විකල්ප 1 යි.