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 194 MARKING

2025/06/22
ictfromabc
Ravindu Bandaranayake

1. Which of the following has the fastest data access?
   වේගවත්ම දත්ත ප්‍රවේශය ඇත්තේ පහත ඒවායින් කුමකද?

   1) Virtual memory / අතථ්‍ය මතකය
   2) Read only memory / පඨන මාත්‍ර මතකය
   3) Cache memory / වාරක මතකය
   4) Hard disk / දෘඩ තැටිය
   5) Random access memory / සසම්භාවී ප්‍රවේශ මතකය

   Answer: 3)

   Explanation / පැහැදිලි කිරීම:

   Cache memory has the fastest data access because it is a small, high-speed memory located close to the CPU, designed to store frequently used data and instructions. Unlike other types of memory, cache reduces the time it takes for the processor to access data by acting as a buffer between the main memory (RAM) and the CPU. This accelerates computing tasks by providing rapid data access, outperforming other storage types like virtual memory or hard disks.
   වාරක මතකයට වේගවත්ම දත්ත ප්‍රවේශය ඇත්තේ එය CPU වෙත සමීපව පිහිටා ඇති, නිතර භාවිතා කරන දත්ත සහ උපදෙස් ගබඩා කිරීමට නිර්මාණය කර ඇති කුඩා, අධිවේගී මතකයක් වන බැවිනි. අනෙකුත් මතක වර්ග මෙන් නොව, වාරක මතකය මඟින් ප්‍රධාන මතකය (RAM) සහ CPU අතර බෆරයක් ලෙස ක්‍රියා කිරීමෙන් සකසනයට දත්ත ප්‍රවේශ වීමට ගතවන කාලය අඩු කරයි. මෙය වේගවත් දත්ත ප්‍රවේශයක් ලබා දීමෙන් පරිගණක කාර්යයන් වේගවත් කරන අතර අථත්‍ය මතකය හෝ දෘඩ තැටි වැනි අනෙකුත් ගබඩා වර්ග අභිබවා යයි.

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

2. BCD equivalent of 93_{10} and 39_{10}?
   BCD 93_{10} සහ 39_{10} ට සමාන වන්නේ?

   1) 1001 0011, 0011 1001
   2) 1011 0011, 0011 1001
   3) 1001 1010, 1001 0001
   4) 1110 0001, 0001 1100
   5) 1001 0011, 1001 0111

   Answer: 1)

   Explanation / පැහැදිලි කිරීම:

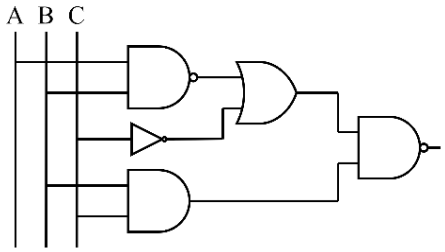
   Convert 93_{10} and 39_{10} to BCD (Binary-Coded Decimal).
   93_{10} සහ 39_{10} BCD (Binary-Coded Decimal) බවට පරිවර්තනය.

93_{10}							
9				3			
2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0
8	4	2	1	8	4	2	1
1	0	0	1	0	0	1	1
1001				0011			
$1001\ 0011_{BCD}$							

39_{10}							
3				9			
2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0
8	4	2	1	8	4	2	1
0	0	1	1	1	0	0	1
0011				1001			
$0011\ 1001_{BCD}$							

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

3. What is the answer after simplifying the logic circuit shown below?
පහත දක්වා ඇති තාර්කික පරිපථය සුළු කර පසු පිළිතුර කුමක්ද?



1) A+B 2) A+$\bar{C}$.B 3) 1 4) 0 5) $\underline{A + \bar{B} + \bar{C}}$

Answer: 5)

Explanation / පැහැදිලි කිරීම:

The output from the logic circuit here is,
මෙහි ඇති තාර්කික පරිපථය මගින් ලැබෙන ප්‍රතිඵලය වනුයේ,

$= \overline{(\overline{AB} + \bar{C}).(BC)}$

Then, it should be simplified as below / **පසුව, එය පහත පරිදි සුළු කළ යුතුය.**

$= \overline{(\overline{AB} + \bar{C}).(BC)}$
$= \overline{\overline{AB}} + \bar{C} + \bar{B} + \bar{C}$ De Morgan's law / **ද මුවාර් ප්‍රමේයය**
$= \overline{\overline{AB}}.\bar{\bar{C}} + \bar{B} + \bar{C}$ De Morgan's law / **ද මුවාර් ප්‍රමේයය**
$= AB.\bar{\bar{C}} + \bar{B} + \bar{C}$ Double complement law / **ද්විත්ව ප්‍රතිලෝම න්‍යාය**
$= ABC + \bar{B} + \bar{C}$ Double complement law / **ද්විත්ව ප්‍රතිලෝම න්‍යාය**
$= AC + \bar{B} + \bar{C}$ Redundancy law / **සමතිරික්ත න්‍යාය**
$= A + \bar{B} + \bar{C}$ Redundancy law / **සමතිරික්ත න්‍යාය**

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

4. What is output of the following python code?
පහත පයිතන් කේතයේ ප්‍රතිදානය කුමක්ද?

```
print ([1,2,3,4,5,6] [-1:2] [::-2])
```

1) `[1,5]` 2) `[5,1]` 3) `[1,3,5]` 4) `[5,3,1]` 5) None of the above.
ඉහත කිසිවක් නොවේ.

Answer: 5)

Explanation / පැහැදිලි කිරීම:

The output of `print([1,2,3,4,5,6][-1:2][::-2])` is `[]` because the slice `[-1:2]` attempts to retrieve elements starting from index `-1` (the last element) up to index `2`, but Python slicing works from left to right, so this range is invalid and results in an empty list. When `[::-2]` is applied to this empty list, it still returns an empty list, since there are no elements to reverse or step through.
`print([1,2,3,4,5,6][-1:2][::-2])` හි ප්‍රතිදානය `[]` වේ. මන්ද, slice `[-1:2]` මගින් `-1` දර්ශකය (අවසාන මූලාංගය) සිට `2` දර්ශකය දක්වා මූලාංග ලබා ගැනීමට උත්සාහ කරයි. නමුත් පයිතන් හි slice වමේ සිට දකුණට ක්‍රියා කරයි, එබැවින් මෙම පරාසය වලංගු නොවන අතර හිස් ලැයිස්තුවක් ඇති කරයි. මෙම හිස් ලැයිස්තුවට `[::-2]` යෙදූ විට, ආපසු හැරවීමට හෝ පියවර හරහා යාමට මූලාංග නොමැති බැවින් එය තවමත් හිස් ලැයිස්තුවක් ප්‍රතිදානය ලෙස ලබා දෙයි.

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

5. Which list method is used to sort the elements of a list in place?
ලැයිස්තුවක මූලාංග, වර්ග කිරීමට භාවිතා කරන ලැයිස්තු ක්‍රමවේදය (method) කුමක්ද?

1) `sort()` 2) `sorted()` 3) `order()` 4) `arrange()` 5) `Sort()`

Answer: 1)

Explanation of each option / එක් එක් විකල්පය පැහැදිලි කිරීම:

Option / විකල්ප 1)
The `sort()` method is a built-in Python list method used to sort the elements of the list in place. This means that the method rearranges the elements of the original list itself and does not create a new list. This method modifies the original list and does not return any value (returns `None`). It sorts the list in ascending order by default, but you can specify the `reverse = True` parameter to sort in descending order.
`sort()` method එක යනු ලැයිස්තුවේ ඇති මූලාංග ස්ථානගත කිරීම සඳහා භාවිතා කරන built-in පයිතන් ලැයිස්තු method එකකි. මෙයින් අදහස් කරන්නේ method එක මුල් ලැයිස්තුවේ මූලාංග නැවත සකස් කරන අතර නව ලැයිස්තුවක් නිර්මාණය නොකරන බවයි. මෙම ක්‍රමය මුල් ලැයිස්තුව වෙනස් කරන අතර කිසිදු අගයක් ලබා නොදේ (`None` නැවත ලබා දෙයි). එය පෙරනිමියෙන් ලැයිස්තුව ආරෝහණ අනුපිළිවෙලට සකසයි, නමුත් ඔබට අවරෝහණ අනුපිළිවෙලින් වර්ග කිරීමට `reverse = True` පරාමිතිය නියම කළ හැක.

Option / විකල්ප 2)
The `sorted()` function, unlike `sort()` does not modify the original list. Instead, it returns a new list that is sorted. This allows you to keep the original list unchanged while getting a sorted version of it. This function can be used with any iterative, not just lists, and also supports an optional reverse parameter to sort in descending order.
`sorted()` ශ්‍රිතය `sort()` මෙන් නොව, මුල් ලැයිස්තුව වෙනස් නොකරයි. ඒ වෙනුවට, එය වර්ග කළ නව ලැයිස්තුවක් ලබා දෙයි. මෙමගින් ඔබට මුල් ලැයිස්තුවේ අනුපිළිවෙළට සකස් කරන ලද අනුවාදයක් ලබා ගන්නා අතරතුර නොවෙනස්ව තබා ගැනීමට ඉඩ සලසයි. මෙම ශ්‍රිතය ලැයිස්තු පමණක් නොව, ඕනෑම පුනරාවර්තනයක් සමඟ භාවිතා කළ හැකි අතර, අවරෝහණ අනුපිළිවෙලට වර්ග කිරීමට විකල්ප ප්‍රතිලෝම පරාමිතියකට ද සහය දක්වයි.

Option / විකල්ප 3)
The `order()` method does not exist in Python for sorting lists. It is not a standard function or method in Python's list operations. If you come across `order()` in other contexts, it might be a custom method, or a misinterpretation of the sorting operations provided by Python.
ලැයිස්තු වර්ග කිරීම සඳහා `order()` method එක පයිතන් හි නොමැත. එය පයිතන්හි ලැයිස්තු මෙහෙයුම් වල සම්මත ශ්‍රිතයක් හෝ method එකක් නොවේ. ඔබට වෙනත් සන්දර්භයන්හිදී `order()` හමු වුවහොත්, එය අභිරුචි ක්‍රමයක් හෝ පයිතන් විසින් සපයනු ලබන වර්ග කිරීමේ මෙහෙයුම් අසත්‍ය ලෙස අර්ථ දැක්වීමක් විය හැකිය.

Option / විකල්ප 4)
Similar to `order()`, `arrange()` is not a standard Python method or function for sorting lists. It does not exist in the Python language or its standard libraries. It might be confused with other methods or could be a term used in other programming contexts or custom implementations.
`order()`, `arrange()` සමාන ලැයිස්තු වර්ග කිරීම සඳහා සම්මත පයිතන් method එකක් හෝ ශ්‍රිතයක් නොවේ. එය පයිතන් භාෂාවේ හෝ එහි සම්මත libraries වල නොපවතී. එය වෙනත් ක්‍රම සමඟ ව්‍යාකුල විය හැකිය, නැතහොත් වෙනත් ක්‍රමලේඛන සන්දර්භයන් හෝ අභිරුචි ක්‍රියාත්මක කිරීම් වලදී භාවිතා කරන යෙදුමක් විය හැකිය.

Option / විකල්ප 5)
This method also does not exist in Python for list sorting. It is not a built-in function or method in Python for sorting lists. The term might be misleading or used in other programming languages or libraries, but it is not part of Python's standard list operations.
මෙම method එකද ලැයිස්තු වර්ග කිරීම සඳහා පයිතන්හි නොපවතී. එය ලැයිස්තු වර්ග කිරීම සඳහා පයිතන්හි ගොඩනඟන ලද ශ්‍රිතයක් හෝ method එකක් නොවේ. මෙම යෙදුම නොමඟ යවන හෝ වෙනත් ක්‍රමලේඛන භාෂා හෝ libraries වල භාවිතා කළ හැක, නමුත් එය පයිතන්හි සම්මත ලැයිස්තු මෙහෙයුම් වල කොටසක් නොවේ.

In summary, when dealing with list sorting in Python, you would use `sort()` to modify the original list in place and `sorted()` to obtain a new sorted list without altering the original. The other terms are not part of Python's standard list operations for sorting.
සාරාංශයක් ලෙස, පයිතන්හි ලැයිස්තු වර්ග කිරීම සමඟ කටයුතු කරන විට, ඔබ මුල් ලැයිස්තුව වෙනස් කිරීමට `sort()` භාවිතා කරන අතර මුල් ලැයිස්තුව වෙනස් නොකර නව වර්ග කළ ලැයිස්තුවක් ලබා ගැනීමට `sorted()` භාවිතා කරයි. අනෙකුත් නියමයන්, වර්ග කිරීම සඳහා පයිතන්හි සම්මත ලැයිස්තු මෙහෙයුම්වල කොටසක් නොවේ.

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