Selection and Insertion Sort

---

1. True Statement About Selection Sort

Selection Sort is characterized by repeatedly finding the minimum element from the unsorted portion and swapping it with the first unsorted element. The correct answers are:

a) In each iteration we find the minimum element in the unsorted part of the array.
b) In each iteration we find the index of the minimum element in the unsorted part of the array.

Option c) is incorrect because Selection Sort swaps the minimum element, not the index. Option d) is also incorrect as Selection Sort performs at most n−1n-1n−1 swaps, not O(n2)O(n^2)O(n2).

2. Worst Case Input for Insertion Sort

Insertion Sort performs worst when the array is sorted in reverse order because each insertion requires moving elements, resulting in the highest number of comparisons and shifts. The correct answer is:

d) Array sorted in reverse order

3. Number of Passes Required for Insertion Sort

In Insertion Sort, the number of passes required is always n−1n-1n−1, where nnn is the number of elements in the array. For an array of 5 elements, the number of passes required is:

c) 4

4. Minimum Possible Sum of Two Numbers Formed from Digits

```
#include <iostream>
#include <vector>
#include <algorithm>
```

```
using namespace std;

int minimumSumOfTwoNumbers(vector<int>& digits) {
    // Sort the digits in ascending order
    sort(digits.begin(), digits.end());

    int num1 = 0, num2 = 0;
    // Distribute the digits between two numbers
    for (size_t i = 0; i < digits.size(); ++i) {
        if (i % 2 == 0) {
            num1 = num1 * 10 + digits[i];
        } else {
            num2 = num2 * 10 + digits[i];
        }
    }

    return num1 + num2;
}

int main() {
    vector<int> digits = {6, 2, 5, 1, 9}; // Example
digits array
    cout << "Minimum possible sum of two numbers: " <<
minimumSumOfTwoNumbers(digits) << endl;
    return 0;
}
```

5. Sorting Strings Using Bubble Sort

```
#include <iostream>
#include <string>

using namespace std;
```

```
void bubbleSortStrings(string arr[], int n) {
    for (int i = 0; i < n - 1; ++i) {
        for (int j = 0; j < n - i - 1; ++j) {
            if (arr[j] > arr[j + 1]) {
                swap(arr[j], arr[j + 1]);
            }
        }
    }
}

int main() {
    string arr[] = {"banana", "apple", "cherry",
"date"};
    int n = sizeof(arr) / sizeof(arr[0]);

    bubbleSortStrings(arr, n);

    cout << "Sorted array of strings: \n";
    for (int i = 0; i < n; ++i) {
        cout << arr[i] << " ";
    }
    cout << endl;

    return 0;
}
```