Sheet2

Merge sort uses which of the following technique to implement sorting?

a) Backtracking

b) greedy algorithm

c) divide and conquer

d) dynamic programming

What is the worst case time complexity of merge sort?

a) O(n log n)

b) O(n2)

c) O(n2 log n)

d) O(n log n2)

Which of the following stable sorting algorithm takes the least time when applied to an almost sorted array?

a)Quick sort

b)Insertion sort

c)Selection sort

d)Merge sort

What is the auxiliary space complexity of merge sort?

a) O(1)

b) O(log n)

c) O(n)

d) O(n log n)

Choose the incorrect statement about merge sort from the following?

a) it is a comparison based sort

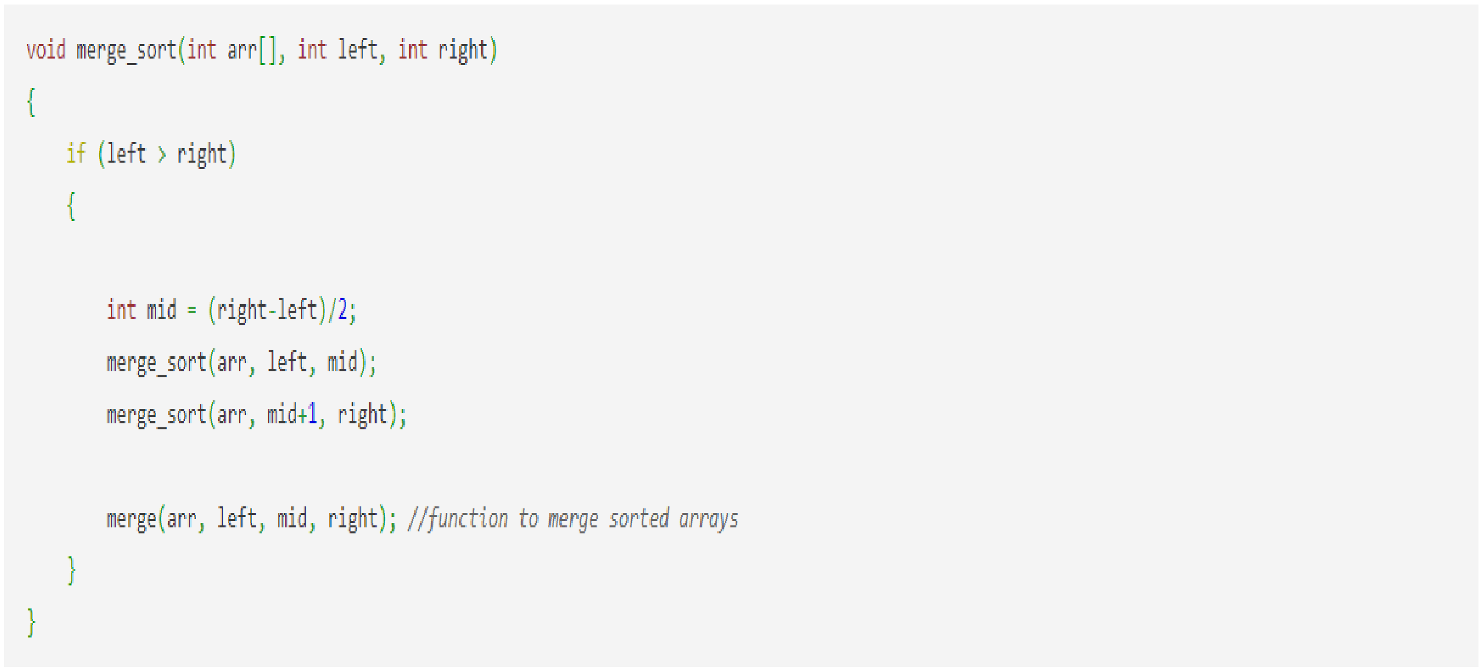
b) it is an adaptive algorithm

c) it is not an in place algorithm

d) it is stable algorithm

Choose the correct code for merge sort.

a)



```
void merge_sort(int arr[], int left, int right)
{
    if (left > right)
    {

        int mid = (right-left)/2;
        merge_sort(arr, left, mid);
        merge_sort(arr, mid+1, right);

        merge(arr, left, mid, right); //function to merge sorted arrays
    }
}
```

b)

```
void merge_sort(int arr[], int left, int right)
{
    if (left < right)
    {

        int mid = left+(right-left)/2;
        merge_sort(arr, left, mid);
        merge_sort(arr, mid+1, right);

        merge(arr, left, mid, right); //function to merge sorted arrays
    }
}
```

c)

```
void merge_sort(int arr[], int left, int right)
{
    if (left < right)
    {

        int mid = left+(right-left)/2;
merge(arr, left, mid, right); //function to merge sorted arrays
        merge_sort(arr, left, mid);
        merge_sort(arr, mid+1, right);


    }
}
```

d)

```
void merge_sort(int arr[], int left, int right)
{
    if (left < right)
    {

        int mid = (right-left)/2;
merge(arr, left, mid, right); //function to merge sorted arrays
        merge_sort(arr, left, mid);
        merge_sort(arr, mid+1, right);



    }
}
```