HackerRank For Work

Spring_2018_INFO6205_Se... 40 minutes

Question - 1

SCORE: **5 points**

Question 1

Given the shuffled array [J, L, E, P, A, C, T, O], what would the array look like after the first partition method call?

- [A, C, E, J, L, O, P, T]
- [A, C, E, J, P, L, T, O]
- [J, C, E, A, P, L, T, O]
- [J, A, C, E, L, O, P, T]

Question - 2

SCORE: **5 points**

Partition

Which of the following code performs the partition operation in QuickSort?

```
A.
private static int partition(int[] arr, int low,
int high)
{
        int left, right, pivot_item = arr[low];
        left = low;
        right = high;
        while(left > right)
        {
                while(arr[left] <= pivot_item)
                {
                        left++;
                }
                while(arr[right] > pivot_item)
                {
                        right--;
                }
                if(left < right)
                {
                        swap(arr, left, right);
                }
        }
        arr[low] = arr[right];
        arr[right] = pivot_item;
        return right;
}
```

```
B.
private static int partition(int[] arr, int low,
int high)
{
        int left, right, pivot_item = arr[low];
        left = low;
        right = high;
        while(left <= right)
        {
                while(arr[left] > pivot_item)
                {
```

```
                        left++;
                }
                while(arr[right] <= pivot_item)
                {
                        right--;
                }
                if(left < right)
                {
                        swap(arr, left, right);
                }
        }
        arr[low] = arr[right];
        arr[right] = pivot_item;
        return right;
}
```

```
C.
private static int partition(int[] arr, int low,
int high)
{
        int left, right, pivot_item = arr[low];
        left = low;
        right = high;
        while(left > right)
        {
                while(arr[left] > pivot_item)
                {
                        left++;
                }
                while(arr[right] <= pivot_item)
                {
                        right--;
                }
                if(left < right)
                {
                        swap(arr, left, right);
                }
        }
        arr[low] = arr[right];
        arr[right] = pivot_item;
        return right;
}
```

```
D.
private static int partition(int[] arr, int low,
int high)
{
        int left, right, pivot_item = arr[low];
        left = low;
        right = high;
        while(left <= right)
        {
                while(arr[left] <= pivot_item)
                {
                        left++;
                }
                while(arr[right] > pivot_item)
                {
                        right--;
                }
                if(left < right)
                {
                        swap(arr, left, right);
                }
        }
        arr[low] = arr[right];
        arr[right] = pivot_item;
        return right;
}
```

- A
- B
- C
- D

Question - 3

SCORE: **5 points**

Question 3

Which scenario will lead to the worst-case performance of quicksort?

- A. The array is sorted.
- B. The array is reverse sorted.
- C. All values in the array are the same.
- A, B and C.
- Only A and B
- Only B and C.
- Only A and C.

Question - 4

SCORE: **5 points**

Question 4

Which sorting methods perform better with partially-sorted arrays? Check all for which it is true.

- selection sort
- insertion sort
- mergesort
- quicksort

Question - 5

SCORE: **30 points**

Merge Sort

Implement merge sort.