Microsoft Interview Questions

These are one of the Microsoft Interview Questions

Round 1:

1. What are your interests? (as in academics/topics)

2. Given a string, search it in a set of strings (say among 1000s of string). What data structure would you use to store those 1000 strings and get the results fastest?
Answer:I answered hash tables but he said suggest a better
one.He said suggest a better one and then gave me one
Tree sort of DS and then asked me to compare the two.

3. Reverse a linked list?

4. Find if there is a loop in a linked List?

5. Given two arrays of numbers, find if each of the two arrays have the same set of integers ? Suggest an algo which can run faster than NlogN ?

6. Validate a Binary search tree? (as in the left- right child follow the property)
Well i gave a the some weird eg where the struct was not a Binary tree but if passed through the test will give positive results.then he asked me to solve for that too.

Round 2:

The interviewer gets a bit serious with each stage. He will test ur work for all possible set of inputs.

Prologue: Well in my case he started with how they require not only a programmer but a designer and coder who writes perfect code.

1. Write a routine that takes input as a string such as

"aabbccdef" and o/p "a2b2c2def"
or
"a4bd2g4" for "aaaabddgggg"

write it perfectly as if it should ready to be shipped after you code it.

2. In the same Question (q1) why will u o/p "abc" for the i/p "abc" instead of "a1b1c1" ?

3. Given a NxN matrix with 0s and 1s. now whenever you encounter a 0 make the corresponding row and column elements 0.

Flip 1 to 0 and 0 remains as they are.

for example

1 0 1 1 0
0 1 1 1 0
1 1 1 1 1
1 0 1 1 1
1 1 1 1 1

results in
0 0 0 0 0
0 0 0 0 0
0 0 1 1 0
0 0 0 0 0
0 0 1 1 0

Round 3:

1. Some Questions on the projects listed on your resume?

2. Some Qs on DB Lock Manager?

3. Given 2 set of arrays of size N(sorted +ve integers) find the median of the resultent array of size 2N.(dont even think of sorting the two arrays in a third array , though u can sort them. Try something better than order N ..order LogN)

4. Given 1000 bottles of juice, one of them contains poison and tastes bitter. Spot the spoiled bottle in minimum sips?

5. Whats the difference b/w a thread and a process? are Word and PowerPoint different processes or threads of a single process?

6. How does a spell checker routine (common to both, word and PowerPoint) used? I mean is the code copied 2 times for each of the processes in the main memory, if they are different processes or how is it used if they are threads.

7. How could you determine if a linked list contains a cycle in it, and, at what node the cycle starts?

There are a number of approaches. The approach I shared is in time N (where N is the number of nodes in your linked list). Assume that the node definition contains a boolean flag, bVisited.

```
struct Node
{
...
bool bVisited;
};
```

Then, to determine whether a node has a loop, you could first set this flag to false for all of the nodes:

```
// Detect cycle
// Note: pHead points to the head of the list (assume already exists)
Node *pCurrent = pHead;
while (pCurrent)
```

```
{
pCurrent->bVisited = false;
pCurrent = pCurrent->pNext;
}
```

A much better approach was submitted by 4Guys visitor George R., a Microsoft interviewer/employee. He recommended using the following technique, which is in time O(N) and space O(1).

Use two pointers.

```
// error checking and checking for NULL at end of list omitted
p1 = p2 = head;

do {
p1 = p1-gt;next;
p2 = p2-gt;next->next;
}while (p1 != p2);
```

p2 is moving through the list twice as fast as p1. If the list is circular, (i.e. a cycle exists) it will eventually get around to that sluggard, p1.

8. How would you reverse a doubly-linked list?

This problem isn't too hard. You just need to start at the head of the list, and iterate to the end. At each node, swap the values of pNext and pPrev. Finally, set pHead to the last node in the list.

```
Node * pCurrent = pHead, *pTemp;
while (pCurrent)
{ pTemp = pCurrent-gt;pNext;
pCurrent-gt;pNext = pCurrent->pPrev;
pCurrent-gt;pPrev = temp;

pHead = pCurrent;

pCurrent = temp;
}
```

9. Assume you have an array that contains a number of strings (perhaps char * a[100]). Each string is a word from the dictionary. Your task, described in high-level terms, is to devise a way to determine and display all of the anagrams within the array (two words are anagrams if they contain the same characters; for example, tales and slate are anagrams.)

Begin by sorting each element in the array in alphabetical order. So, if one element of your array was slate, it would be rearranged to form aelst (use some mechanism to know that the particular instance of aelst maps to slate). At this point, you slate and tales would be identical: aelst.
Next, sort the entire array of these modified dictionary words. Now, all of the anagrams are grouped together. Finally, step through the array and display duplicate terms, mapping the sorted letters (aelst) back to the word (slate or tales).

10. Given the following prototype:
int compact(int * p, int size);

write a function that will take a sorted array, possibly with duplicates, and compact the array, returning the new length of the array. That is, if p points to an array containing: 1, 3, 7, 7, 8, 9, 9, 9, 10, when the function returns, the contents of p should be: 1, 3, 7, 8, 9, 10, with a length of 5 returned.

A single loop will accomplish this.

```
int compact(int * p, int size)
{
int current, insert = 1;
for (current=1; current < size; current++)
if (p[current] != p[insert-1])
{
p[insert] = p[current];
current++;
insert++;
}else
current++;
}
```