**Array related problems (total 15 questions)**

| **SL** | **Problem statement** | **Difficulty levels** |
| --- | --- | --- |
|  | WAP that will take n integer numbers into an array, and then print all the integers into reverse order (from the last valid index to index 0).   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5 | 5 4 3 2 1 | | 6  2 8 3 9 0 1 | 1 0 9 3 8 2 | | \* |
|  | WAP that will take n integer numbers into an array, and then sum up all the integers in that array.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5 | 15 | | 6  2 8 3 9 0 1 | 23 | | \* |
|  | WAP that will take n integer numbers into an array, and then sum up all the even integers in that array.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5 | 6 | | 6  2 8 3 9 0 1 | 10 | | \* |
|  | WAP that will take n integer numbers into an array, and then sum up all the even indexed integers in that array.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5 | 9 | | 6  2 8 3 9 0 1 | 5 | | \* |

|  | WAP that will take n integer numbers into an array, and then reverse all the integers within that array. Finally print them all from 0 index to last valid index.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5 | 5 4 3 2 1 | | 6  2 8 3 9 0 1 | 1 0 9 3 8 2 | | \*\* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WAP that will take n integer numbers into an array, and then find the maximum -minimum among them with its index position.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5 | Max: 5, Index: 4  Min: 1, Index: 0 | | 6  2 8 3 9 0 1 | Max: 9, Index: 3  Min: 0, Index: 4 | | \*\* |
|  | WAP that will take n alphabets into an array, and then count number of vowels in that array.   | **Sample input** | **Sample output** | | --- | --- | | 7  AKIOUEH | Count: 5 | | 29  UNITEDINTERNATIONALUNIVERSITY | Count: 13 | | \* |
|  | WAP that will take n integers into an array, and then search a number into that array. If found then print its index. If not found then print “NOT FOUND”.   | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 3 2 6 4 3  3 | FOUND at index position: 3, 7 | | 8  7 8 1 3 2 6 4 3  5 | NOT FOUND | | \* |

|  | WAP that will take n integers into an array A, and then copy all numbers in reverse order from array A to another array B. Finally show all elements of both array A and B.   | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 3 2 6 4 3 | Array A : 7 8 1 3 2 6 4 3  Array B : 3 4 6 2 3 1 8 7 | | 3  3 2 1 | Array A : 3 2 1  Array B : 1 2 3 | | \* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WAP that will first take n integers into an array A and then m integers into array B. Now swap all elements between array A and B. Finally show all elements of both array A and B.   | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 3 2 6 4 3  3  3 2 1 | Array A : 3 2 1  Array B : 7 8 1 3 2 6 4 3 | | \*\* |
|  | WAP that will take n positive integers into an array A. Now find all the integers that are divisible by 3 and replace them by -1 in array A. Finally show all elements of array A.   | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 3 2 6 4 3 | 7 8 1 -1 2 -1 4 -1 | | 3  3 2 1 | -1 2 1 | | \* |
|  | WAP that will take n integers into an array A. Now sort them in ascending order within that array. Finally show all elements of array A.  Reference: <http://en.wikipedia.org/wiki/Bubble_sort>   | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 3 2 6 4 3 | 1 2 3 3 4 6 7 8 | | 3  3 2 1 | 1 2 3 | | \*\*\* |

|  | WAP that will take n integers into an array A. Now remove all duplicates numbers from that array. Finally print all elements from that array.   | **Sample input** | **Sample output** | | --- | --- | | 8  2 8 1 3 2 6 4 3 | 2 8 1 3 6 4 | | 3  3 3 3 | 3 | | 4  6 7 8 9 | 6 7 8 9 | | \*\* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WAP that will take n integers into an array A and m positive integers into array B. Now find the intersection (set operation) of array A and B.     | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 5 2 6 4 3  6  1 3 6 0 9 2 | 1 2 6 3 | | 3  1 2 3  2  4 5 | Empty set | | \*\* |
|  | WAP that will take n integers into an array A and m positive integers into array B. Now find the union (set operation) of array A and B.     | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 5 2 6 4 3  6  1 3 6 0 9 2 | 7 8 1 5 2 6 4 3 0 9 | | 3  1 2 3  2  4 5 | 1 2 3 4 5 | | \*\* |

|  | WAP that will take n integers into an array A and m positive integers into array B. Now find the difference (set operation) of array A and B or (A-B).     | **Sample input** | **Sample output** | | --- | --- | | 8  7 8 1 5 2 6 4 3  6  1 3 6 0 9 2 | 7 8 5 4 | | 3  1 2 3  2  4 5 | 1 2 3 | | \*\* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |

|  | WAP that will take (n x n) integer inputs into a square matrix of dimension n (where n must be an odd number). Then calculate sum of the integers based on following position pattern (consider only the boxed position during the sum). Please see the input-output.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5  2 3 4 1 6  3 4 9 6 7  4 2 6 7 8  5 4 3 2 1 | 71 | | 7  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1 | 25 | | \*\* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WAP that will take (n x n) integer inputs into a square matrix of dimension n (where n must be an odd number). Then calculate sum of the integers based on following position pattern (consider only the boxed position during the sum). Please see the input-output.   | **Sample input** | **Sample output** | | --- | --- | | 5  1 2 3 4 5  2 3 4 1 6  3 4 9 6 7  4 2 6 7 8  5 4 3 2 1 | 65 | | 7  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1  1 1 1 1 1 1 1 | 33 | | \*\* |

|  | WAP that will take (m x n) integer inputs into a matrix of dimension m x n. Now reverse that matrix within itself and display it. Reversal means swap 1st column with the nth column, swap 2nd column with the (n-1)th column and so on…   | **Sample input** | **Sample output** | | --- | --- | | 3 3  1 2 3  4 5 6  2 9 2 | 3 2 1  6 5 4  2 9 2 | | 2 6  1 2 3 4 5 6  9 8 7 6 5 4 | 6 5 4 3 2 1  4 5 6 7 8 9 | | \*\* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | WAP that will take (n x n) integer inputs into a square matrix of dimension n. Now determine whether the matrix is symmetric or not.  Reference: <http://en.wikipedia.org/wiki/Symmetric_matrix>   | **Sample input** | **Sample output** | | --- | --- | | 3  1 7 3  7 4 5  3 5 6 | Yes | | 2  1 3  4 2 | No | | \*\* |
|  | WAP that will take (m x n) positive integer inputs into a matrix of dimension m x n. Now replace all the duplicate integers by -1 in that matrix. Finally display it.   | **Sample input** | **Sample output** | | --- | --- | | 3 3  1 7 3  7 4 5  3 5 6 | 1 7 3  -1 4 5  -1 -1 6 | | 2 6  2 2 2 2 2 2  6 5 4 3 2 1 | 2 -1 -1 -1 -1 -1  6 5 4 3 -1 1 | | \*\*\* |

|  | WAP that will take (m x n) integer inputs into a matrix of dimension m x n. Now just simply add all the integers in that matrix and show the result.   | **Sample input** | **Sample output** | | --- | --- | | 3 3  1 7 3  7 4 5  3 5 6 | 41 | | 2 6  2 2 2 2 2 2  6 5 4 3 2 1 | 33 | | \* |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |