Basic Array Problems (1-10)

1. Find the maximum element in an array

Input-Output Format
makefile
CopyEdit

```
Input:
5
1 4 3 9 2

Output:
9
```

Solution
c
CopyEdit

```
#include <stdio.h>

int findMax(int arr[], int n) {
    int max = arr[0];
    for (int i = 1; i < n; i++)
        if (arr[i] > max)
            max = arr[i];
    return max;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", findMax(arr, n));
    return 0;
}
```

---

2. Find the minimum element in an array

Input-Output Format

makefile

CopyEdit

```
Input:
5
5 8 1 2 7

Output:
1
```

Solution

c

CopyEdit

```
#include <stdio.h>

int findMin(int arr[], int n) {
    int min = arr[0];
    for (int i = 1; i < n; i++)
        if (arr[i] < min)
            min = arr[i];
    return min;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", findMin(arr, n));
    return 0;
}
```

---

3. Find the sum of all elements in an array

Input-Output Format

makefile

CopyEdit

```
Input:
4
1 2 3 4

Output:
10
```

Solution

c

CopyEdit

```
#include <stdio.h>

int arraySum(int arr[], int n) {
    int sum = 0;
    for (int i = 0; i < n; i++)
        sum += arr[i];
    return sum;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", arraySum(arr, n));
    return 0;
}
```

---

4. Reverse an array

Input-Output Format

makefile
CopyEdit

```
Input:
5
1 2 3 4 5

Output:
5 4 3 2 1
```

Solution

c
CopyEdit

```c
#include <stdio.h>

void reverseArray(int arr[], int n) {
    for (int i = n - 1; i >= 0; i--)
        printf("%d ", arr[i]);
    printf("\n");
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    reverseArray(arr, n);
    return 0;
}
```

---

5. Find the second largest element in an array

Input-Output Format

makefile

CopyEdit

```
Input:
6
12 35 1 10 34 1

Output:
34
```

Solution

c

CopyEdit

```
#include <stdio.h>
#include <limits.h>

int secondLargest(int arr[], int n) {
    int first = INT_MIN, second = INT_MIN;
    for (int i = 0; i < n; i++) {
        if (arr[i] > first) {
            second = first;
            first = arr[i];
        } else if (arr[i] > second && arr[i] != first) {
            second = arr[i];
        }
    }
    return second;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", secondLargest(arr, n));
    return 0;
}
```

---

6. Sort an array using bubble sort

Input-Output Format

makefile

CopyEdit

```
Input:
5
5 2 9 1 3

Output:
1 2 3 5 9
```

Solution

c

CopyEdit

```
#include <stdio.h>

void bubbleSort(int 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]) {
                int temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
            }
```

```c
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    bubbleSort(arr, n);

    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    return 0;
}
```

---

7. Check if an array is sorted

Input-Output Format

yaml

CopyEdit

```
Input:
5
1 2 3 4 5

Output:
YES
```

Solution

c

CopyEdit

```c
#include <stdio.h>
```

```
int isSorted(int arr[], int n) {
    for (int i = 1; i < n; i++)
        if (arr[i] < arr[i - 1])
            return 0;
    return 1;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%s\n", isSorted(arr, n) ? "YES" : "NO");
    return 0;
}
```

---

8. Find the frequency of elements in an array

Input-Output Format

rust
CopyEdit

```
Input:
6
1 2 3 2 1 1

Output:
1 -> 3
2 -> 2
3 -> 1
```

Solution

c
CopyEdit

```c
#include <stdio.h>

void frequencyCount(int arr[], int n) {
    int visited[100] = {0};
    for (int i = 0; i < n; i++) {
        if (visited[i]) continue;
        int count = 1;
        for (int j = i + 1; j < n; j++) {
            if (arr[i] == arr[j]) {
                count++;
                visited[j] = 1;
            }
        }
        printf("%d -> %d\n", arr[i], count);
    }
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    frequencyCount(arr, n);
    return 0;
}
```

---

9. Rotate an array by K positions

Input-Output Format
makefile
CopyEdit
`Input:`
5

```
1 2 3 4 5
2

Output:
4 5 1 2 3
```

Solution

c

CopyEdit

```
#include <stdio.h>

void rotateArray(int arr[], int n, int k) {
    int temp[n];
    for (int i = 0; i < n; i++)
        temp[(i + k) % n] = arr[i];
    for (int i = 0; i < n; i++)
        arr[i] = temp[i];
}

int main() {
    int n, k, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);
    scanf("%d", &k);

    rotateArray(arr, n, k);

    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    return 0;
}
```

10. Find if a number is present in an array

Input-Output Format

yaml

CopyEdit

```
Input:
6
5 2 9 1 3 4
3

Output:
YES
```

Solution

c

CopyEdit

```
#include <stdio.h>

int isPresent(int arr[], int n, int num) {
    for (int i = 0; i < n; i++) {
        if (arr[i] == num) {
            return 1;
        }
    }
    return 0;
}

int main() {
    int n, num, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);
    scanf("%d", &num);
    if (isPresent(arr, n, num))
        printf("YES\n");
    else
        printf("NO\n");

    return 0;
}
```

Intermediate Array Problems (11-20)

11. Merge two sorted arrays

Input-Output Format

makefile

CopyEdit

```
Input:
5
1 3 5 7 9
4
2 4 6 8

Output:
1 2 3 4 5 6 7 8 9
```

Solution

c

CopyEdit

```
#include <stdio.h>

void mergeArrays(int arr1[], int n1, int arr2[], int n2) {
    int i = 0, j = 0;
    while (i < n1 && j < n2) {
        if (arr1[i] < arr2[j]) {
            printf("%d ", arr1[i]);
            i++;
        } else {
            printf("%d ", arr2[j]);
            j++;
        }
    }
    while (i < n1) {
        printf("%d ", arr1[i]);
        i++;
    }
    while (j < n2) {
        printf("%d ", arr2[j]);
        j++;
    }
    printf("\n");
}

int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d", &n1);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    scanf("%d", &n2);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);

    mergeArrays(arr1, n1, arr2, n2);
    return 0;
}
```

---

12. Find the intersection of two arrays

Input-Output Format

makefile
CopyEdit

```
Input:
5
1 2 3 4 5
4
2 4 6 8

Output:
2 4
```

Solution

c

CopyEdit

```c
#include <stdio.h>

void intersection(int arr1[], int n1, int arr2[], int n2) {
    for (int i = 0; i < n1; i++) {
        for (int j = 0; j < n2; j++) {
            if (arr1[i] == arr2[j]) {
                printf("%d ", arr1[i]);
                break;
            }
        }
    }
    printf("\n");
}

int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d", &n1);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    scanf("%d", &n2);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);
    intersection(arr1, n1, arr2, n2);
    return 0;
}
```

---

13. Find the union of two arrays

Input-Output Format

makefile

CopyEdit

```
Input:
5
1 2 3 4 5
4
2 4 6 8

Output:
1 2 3 4 5 6 8
```

Solution

c

CopyEdit

```
#include <stdio.h>

void unionArrays(int arr1[], int n1, int arr2[], int n2) {
    int i, j;
    for (i = 0; i < n1; i++) {
        printf("%d ", arr1[i]);
    }
    for (i = 0; i < n2; i++) {
        int found = 0;
        for (j = 0; j < n1; j++) {
            if (arr2[i] == arr1[j]) {
                found = 1;
                break;
            }
        }
        if (!found) {
            printf("%d ", arr2[i]);
        }
    }
    printf("\n");
}

int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d", &n1);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    scanf("%d", &n2);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);

    unionArrays(arr1, n1, arr2, n2);
    return 0;
}
```

14. Remove duplicates from an array

Input-Output Format

makefile

CopyEdit

```
Input:
6
1 2 3 2 4 1

Output:
1 2 3 4
```

Solution

c

CopyEdit

```
#include <stdio.h>

void removeDuplicates(int arr[], int *n) {
    int temp[100];
    int k = 0;
    for (int i = 0; i < *n; i++) {
        int found = 0;
        for (int j = 0; j < k; j++) {
            if (arr[i] == temp[j]) {
                found = 1;
                break;
            }
        }
        if (!found) {
            temp[k++] = arr[i];
        }
    }
    *n = k;
    for (int i = 0; i < *n; i++) {
        arr[i] = temp[i];
    }
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    removeDuplicates(arr, &n);

    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    return 0;
}
```

---

15. Check if two arrays are equal

Input-Output Format

yaml

CopyEdit

```
Input:
4
1 2 3 4
4
1 2 3 4

Output:
YES
```

Solution

c

CopyEdit

```
#include <stdio.h>

int areArraysEqual(int arr1[], int arr2[], int n1, int n2) {
    if (n1 != n2)
        return 0;

    for (int i = 0; i < n1; i++) {
        if (arr1[i] != arr2[i])
            return 0;
    }
    return 1;
}
```

```
int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d", &n1);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    scanf("%d", &n2);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);

    printf("%s\n", areArraysEqual(arr1, arr2, n1, n2) ? "YES" :
"NO");
    return 0;
}
```

---

16. Find the longest consecutive sequence in an array

Input-Output Format

makefile

CopyEdit

```
Input:
7
100 4 200 1 3 2 1

Output:
4
```

Solution

c

CopyEdit

```
#include <stdio.h>
#include <stdlib.h>

int compare(const void *a, const void *b) {
    return (*(int *)a - *(int *)b);
}

int longestConsecutive(int arr[], int n) {
    if (n == 0) return 0;
    qsort(arr, n, sizeof(int), compare);

    int maxCount = 1, count = 1;
    for (int i = 1; i < n; i++) {
        if (arr[i] == arr[i - 1] + 1)
            count++;
        else if (arr[i] != arr[i - 1])
            count = 1;
        maxCount = count > maxCount ? count : maxCount;
    }
    return maxCount;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", longestConsecutive(arr, n));
    return 0;
}
```

---

17. Find the common elements in three arrays

Input-Output Format
makefile
CopyEdit

```
Input:
```

```
5
1 2 3 4 5
4
2 3 4 6
6
3 4 5 7 8 9

Output:
3 4 5
```

Solution

c

CopyEdit

```
#include <stdio.h>

void commonElements(int arr1[], int n1, int arr2[], int n2, int
arr3[], int n3) {
    for (int i = 0; i < n1; i++) {
        for (int j = 0; j < n2; j++) {
            if (arr1[i] == arr2[j]) {
                for (int k = 0; k < n3; k++) {
                    if (arr1[i] == arr3[k]) {
                        printf("%d ", arr1[i]);
                        break;
                    }
                }
                break;
            }
        }
    }
    printf("\n");
}

int main() {
    int n1, n2, n3, arr1[100], arr2[100], arr3[100];
    scanf("%d", &n1);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    scanf("%d", &n2);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);
    scanf("%d", &n3);
    for (int i = 0; i < n3; i++)
        scanf("%d", &arr3[i]);

    commonElements(arr1, n1, arr2, n2, arr3, n3);
    return 0;
}
```

---

18. Find the pair with the maximum sum in an array

Input-Output Format

makefile
CopyEdit

```
Input:
5
1 2 3 4 5

Output:
9
```

Solution

c
CopyEdit

```
#include <stdio.h>

int maxSumPair(int arr[], int n) {
    int maxSum = arr[0] + arr[1];
    for (int i = 0; i < n - 1; i++) {
        for (int j = i + 1; j < n; j++) {
            int sum = arr[i] + arr[j];
            if (sum > maxSum)
                maxSum = sum;
        }
    }
    return maxSum;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", maxSumPair(arr, n));
    return 0;
}
```

---

19. Find if an array is a palindrome

Input-Output Format

yaml

CopyEdit

```
Input:
5
1 2 3 2 1

Output:
YES
```

Solution

c

CopyEdit

```
#include <stdio.h>

int isPalindrome(int arr[], int n) {
    for (int i = 0; i < n / 2; i++) {
        if (arr[i] != arr[n - i - 1])
            return 0;
    }
    return 1;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%s\n", isPalindrome(arr, n) ? "YES" : "NO");
    return 0;
}
```

---

20. Find the longest increasing subsequence in an array

Input-Output Format

makefile

CopyEdit

```
Input:
6
10 22 9 33 21 50

Output:
4
```

Solution

c

CopyEdit

```
#include <stdio.h>

int longestIncreasingSubsequence(int arr[], int n) {
    int lis[100];
    for (int i = 0; i < n; i++)
        lis[i] = 1;

    for (int i = 1; i < n; i++) {
        for (int j = 0; j < i; j++) {
            if (arr[i] > arr[j] && lis[i] < lis[j] + 1)
                lis[i] = lis[j] + 1;
        }
    }
    int maxLis = lis[0];
    for (int i = 1; i < n; i++) {
        if (lis[i] > maxLis)
            maxLis = lis[i];
    }
    return maxLis;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", longestIncreasingSubsequence(arr, n));
    return 0;
}
```

21. Rotate a matrix by 90 degrees (clockwise)

Input-Output Format

makefile
CopyEdit

```
Input:
3
1 2 3
4 5 6
7 8 9

Output:
7 4 1
8 5 2
9 6 3
```

Solution
c
CopyEdit

```c
#include <stdio.h>

void rotateMatrix(int matrix[100][100], int n) {
    for (int i = 0; i < n / 2; i++) {
        for (int j = i; j < n - i - 1; j++) {
            int temp = matrix[i][j];
            matrix[i][j] = matrix[n - j - 1][i];
            matrix[n - j - 1][i] = matrix[n - i - 1][n - j - 1];
            matrix[n - i - 1][n - j - 1] = matrix[j][n - i - 1];
            matrix[j][n - i - 1] = temp;
        }
    }
}

int main() {
    int n, matrix[100][100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < n; j++)
```

```
            scanf("%d", &matrix[i][j]);

    rotateMatrix(matrix, n);

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++)
            printf("%d ", matrix[i][j]);
        printf("\n");
    }

    return 0;
}
```

---

22. Transpose of a matrix

Input-Output Format

makefile

CopyEdit

```
Input:
3
1 2 3
4 5 6
7 8 9

Output:
1 4 7
2 5 8
3 6 9
```

Solution

c

CopyEdit

```
#include <stdio.h>
```

```
void transposeMatrix(int matrix[100][100], int n, int m) {
    for (int i = 0; i < m; i++) {
        for (int j = 0; j < n; j++) {
            printf("%d ", matrix[j][i]);
        }
        printf("\n");
    }
}

int main() {
    int n, m, matrix[100][100];
    scanf("%d %d", &n, &m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            scanf("%d", &matrix[i][j]);

    transposeMatrix(matrix, n, m);
    return 0;
}
```

---

23. Find the maximum sum subarray (Kadane’s Algorithm)

Input-Output Format
makefile
CopyEdit

```
Input:
6
-2 1 -3 4 -1 2 1 -5 4

Output:
6
```

Solution
c

CopyEdit

```
#include <stdio.h>

int maxSubArraySum(int arr[], int n) {
    int maxSum = arr[0], currentSum = arr[0];
    for (int i = 1; i < n; i++) {
        currentSum = (currentSum + arr[i] > arr[i]) ? currentSum
+ arr[i] : arr[i];
        maxSum = (maxSum > currentSum) ? maxSum : currentSum;
    }
    return maxSum;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", maxSubArraySum(arr, n));
    return 0;
}
```

---

24. Count the number of islands in a 2D matrix

Input-Output Format
makefile
CopyEdit

```
Input:
4 5
1 1 0 0 0
1 1 0 0 1
0 0 0 1 1
0 0 0 0 0
```

```
Output:
3
```

Solution

c

CopyEdit

```
#include <stdio.h>

void dfs(int matrix[100][100], int visited[100][100], int x, int
y, int n, int m) {
    if (x < 0 || x >= n || y < 0 || y >= m || visited[x][y] ||
matrix[x][y] == 0)
        return;

    visited[x][y] = 1;

    dfs(matrix, visited, x + 1, y, n, m);
    dfs(matrix, visited, x - 1, y, n, m);
    dfs(matrix, visited, x, y + 1, n, m);
    dfs(matrix, visited, x, y - 1, n, m);
}

int countIslands(int matrix[100][100], int n, int m) {
    int visited[100][100] = {0};
    int count = 0;
    for (int i = 0; i < n; i++) {
        for (int j = 0; j < m; j++) {
            if (matrix[i][j] == 1 && !visited[i][j]) {
                dfs(matrix, visited, i, j, n, m);
                count++;
            }
        }
    }
    return count;
}
```

```
int main() {
    int n, m, matrix[100][100];
    scanf("%d %d", &n, &m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            scanf("%d", &matrix[i][j]);

    printf("%d\n", countIslands(matrix, n, m));
    return 0;
}
```

---

25. Multiply two matrices

Input-Output Format

makefile

CopyEdit

```
Input:
2 3
1 2 3
4 5 6
3 2
7 8
9 10
11 12

Output:
58 64
139 154
```

Solution

c

CopyEdit

```
#include <stdio.h>
```

```
void multiplyMatrices(int mat1[100][100], int mat2[100][100],
int res[100][100], int n1, int m1, int n2, int m2) {
    for (int i = 0; i < n1; i++) {
        for (int j = 0; j < m2; j++) {
            res[i][j] = 0;
            for (int k = 0; k < m1; k++) {
                res[i][j] += mat1[i][k] * mat2[k][j];
            }
        }
    }
}

int main() {
    int n1, m1, mat1[100][100], n2, m2, mat2[100][100],
res[100][100];
    scanf("%d %d", &n1, &m1);
    for (int i = 0; i < n1; i++)
        for (int j = 0; j < m1; j++)
            scanf("%d", &mat1[i][j]);
    scanf("%d %d", &n2, &m2);
    for (int i = 0; i < n2; i++)
        for (int j = 0; j < m2; j++)
            scanf("%d", &mat2[i][j]);

    multiplyMatrices(mat1, mat2, res, n1, m1, n2, m2);

    for (int i = 0; i < n1; i++) {
        for (int j = 0; j < m2; j++)
            printf("%d ", res[i][j]);
        printf("\n");
    }

    return 0;
}
```

---

26. Find the row with the maximum sum in a matrix

Input-Output Format

makefile

CopyEdit

```
Input:
3 3
1 2 3
4 5 6
7 8 9

Output:
7 8 9
```

Solution

c

CopyEdit

```c
#include <stdio.h>

void findRowWithMaxSum(int matrix[100][100], int n, int m) {
    int maxSum = 0, rowIndex = 0;
    for (int i = 0; i < n; i++) {
        int rowSum = 0;
        for (int j = 0; j < m; j++) {
            rowSum += matrix[i][j];
        }
        if (rowSum > maxSum) {
            maxSum = rowSum;
            rowIndex = i;
        }
    }
    for (int j = 0; j < m; j++) {
        printf("%d ", matrix[rowIndex][j]);
    }
    printf("\n");
}

int main() {
    int n, m, matrix[100][100];
    scanf("%d %d", &n, &m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            scanf("%d", &matrix[i][j]);

    findRowWithMaxSum(matrix, n, m);
    return 0;
}
```

---

27. Find the maximum element in each row of a matrix

Input-Output Format

makefile

CopyEdit

```
Input:
3 3
1 2 3
4 5 6
7 8 9

Output:
3 6 9
```

Solution

c

CopyEdit

```
#include <stdio.h>
```

```
void findMaxInEachRow(int matrix[100][100], int n, int m) {
    for (int i = 0; i < n; i++) {
        int maxElement = matrix[i][0];
        for (int j = 1; j < m; j++) {
            if (matrix[i][j] > maxElement) {
                maxElement = matrix[i][j];
            }
        }
        printf("%d ", maxElement);
    }
    printf("\n");
}

int main() {
    int n, m, matrix[100][100];
    scanf("%d %d", &n, &m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            scanf("%d", &matrix[i][j]);

    findMaxInEachRow(matrix, n, m);
    return 0;
}
```

28. Count occurrences of each element in an array

Input-Output Format
makefile
CopyEdit

```
Input:
6
1 2 2 3 3 3

Output:
1: 1
```

```
2: 2
3: 3
```

Solution

c

CopyEdit

```c
#include <stdio.h>

void countOccurrences(int arr[], int n) {
    int counted[n];
    for (int i = 0; i < n; i++) {
        counted[i] = 0;
    }

    for (int i = 0; i < n; i++) {
        if (counted[i] == 0) {
            int count = 1;
            for (int j = i + 1; j < n; j++) {
                if (arr[i] == arr[j]) {
                    counted[j] = 1;
                    count++;
                }
            }
            printf("%d: %d\n", arr[i], count);
        }
    }
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    countOccurrences(arr, n);
    return 0;
}
```

---

29. Move all zeroes to the end of an array

Input-Output Format

makefile

CopyEdit

```
Input:
5
0 1 2 0 3

Output:
1 2 3 0 0
```

Solution

c

CopyEdit

```c
#include <stdio.h>

void moveZeroesToEnd(int arr[], int n) {
    int index = 0;

    for (int i = 0; i < n; i++) {
        if (arr[i] != 0) {
            arr[index++] = arr[i];
        }
    }

    while (index < n) {
        arr[index++] = 0;
    }
}
```

```
int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    moveZeroesToEnd(arr, n);

    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    return 0;
}
```

---

30. Find the intersection of two arrays

Input-Output Format

makefile

CopyEdit

```
Input:
5 4
1 2 2 1
2 2

Output:
2 2
```

Solution

c

CopyEdit

```
#include <stdio.h>

void findIntersection(int arr1[], int arr2[], int n1, int n2) {
    int i = 0, j = 0;
    while (i < n1 && j < n2) {
        if (arr1[i] < arr2[j]) {
            i++;
        } else if (arr1[i] > arr2[j]) {
            j++;
        } else {
            printf("%d ", arr1[i]);
            i++;
            j++;
        }
    }
    printf("\n");
}

int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d %d", &n1, &n2);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);

    findIntersection(arr1, arr2, n1, n2);
    return 0;
}
```

---

31. Spiral traversal of a matrix

Input-Output Format

makefile

CopyEdit

```
Input:
3 3
```

```
1 2 3
4 5 6
7 8 9

Output:
1 2 3 6 9 8 7 4 5
```

Solution

c

CopyEdit

```c
#include <stdio.h>

void spiralTraversal(int matrix[100][100], int n, int m) {
    int top = 0, bottom = n - 1, left = 0, right = m - 1;

    while (top <= bottom && left <= right) {
        for (int i = left; i <= right; i++) {
            printf("%d ", matrix[top][i]);
        }
        top++;

        for (int i = top; i <= bottom; i++) {
            printf("%d ", matrix[i][right]);
        }
        right--;

        if (top <= bottom) {
            for (int i = right; i >= left; i--) {
                printf("%d ", matrix[bottom][i]);
            }
            bottom--;
        }

        if (left <= right) {
            for (int i = bottom; i >= top; i--) {
                printf("%d ", matrix[i][left]);
            }
            left++;
        }
    }
    printf("\n");
}

int main() {
    int n, m, matrix[100][100];
    scanf("%d %d", &n, &m);
    for (int i = 0; i < n; i++)
        for (int j = 0; j < m; j++)
            scanf("%d", &matrix[i][j]);

    spiralTraversal(matrix, n, m);
    return 0;
}
```

---

32. Find the longest increasing subsequence

Input-Output Format

makefile
CopyEdit

```
Input:
6
10 22 9 33 21 50

Output:
4
```

Solution

c
CopyEdit

```c
#include <stdio.h>

int longestIncreasingSubsequence(int arr[], int n) {
    int lis[n];
    lis[0] = 1;

    for (int i = 1; i < n; i++) {
        lis[i] = 1;
        for (int j = 0; j < i; j++) {
            if (arr[i] > arr[j] && lis[i] < lis[j] + 1) {
                lis[i] = lis[j] + 1;
            }
        }
    }

    int max = 1;
    for (int i = 0; i < n; i++) {
        if (lis[i] > max)
            max = lis[i];
    }
    return max;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", longestIncreasingSubsequence(arr, n));
    return 0;
}
```

---

33. Rotate a matrix 90 degrees in-place (without extra space)

Input-Output Format

makefile

CopyEdit

```
Input:
3
1 2 3
4 5 6
7 8 9

Output:
7 4 1
8 5 2
9 6 3
```

Solution

c

CopyEdit

```
#include <stdio.h>

void rotateMatrixInPlace(int matrix[100][100], int n) {
    for (int i = 0; i < n / 2; i++) {
        for (int j = i; j < n - i - 1; j++) {
            int temp = matrix[i][j];
            matrix[i][j] = matrix[n - j - 1][i];
            matrix[n - j - 1][i] = matrix[n - i - 1][n - j - 1];
            matrix[n - i - 1][n - j - 1] = matrix[j][n - i - 1];
            matrix[j][n - i - 1] = temp;
        }
    }
}

int main() {
    int n, matrix[100][100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
```

```
        for (int j = 0; j < n; j++)
            scanf("%d", &matrix[i][j]);

    rotateMatrixInPlace(matrix, n);

    for (int i = 0; i < n; i++) {
        for (int j = 0; j < n; j++)
            printf("%d ", matrix[i][j]);
        printf("\n");
    }

    return 0;
}
```

---

34. Count the number of peaks in an array

Input-Output Format

makefile

CopyEdit

```
Input:
5
1 3 2 1 4

Output:
2
```

Solution

c

CopyEdit

```
#include <stdio.h>

int countPeaks(int arr[], int n) {
    int count = 0;
    if (n > 1) {
        if (arr[0] > arr[1]) count++;  // First element
        if (arr[n - 1] > arr[n - 2]) count++;  // Last element
        for (int i = 1; i < n - 1; i++) {
            if (arr[i] > arr[i - 1] && arr[i] > arr[i + 1])
                count++;
        }
    }
    return count;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", countPeaks(arr, n));
    return 0;
}
```

---

35. Check if an array is a subset of another array

Input-Output Format

yaml

CopyEdit

```
Input:
5 6
1 2 3 4 5
4 3 2 1 6 7

Output:
No
```

Solution

c
CopyEdit

```
#include <stdio.h>

int isSubset(int arr1[], int arr2[], int n1, int n2) {
    for (int i = 0; i < n1; i++) {
        int found = 0;
        for (int j = 0; j < n2; j++) {
            if (arr1[i] == arr2[j]) {
                found = 1;
                break;
            }
        }
        if (!found) return 0;
    }
    return 1;
}

int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d %d", &n1, &n2);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);

    if (isSubset(arr1, arr2, n1, n2))
        printf("Yes\n");
    else
        printf("No\n");

    return 0;
}
```

36. Rearrange the array such that even-indexed elements are even and odd-indexed elements are odd

Input-Output Format

makefile

CopyEdit

```
Input:
6
1 2 3 4 5 6

Output:
2 1 4 3 6 5
```

Solution

c

CopyEdit

```
#include <stdio.h>

void rearrangeEvenOdd(int arr[], int n) {
    int evenIndex = 0, oddIndex = n - 1;
    while (evenIndex < oddIndex) {
        while (arr[evenIndex] % 2 == 0 && evenIndex < oddIndex)
{
            evenIndex++;
        }
        while (arr[oddIndex] % 2 != 0 && evenIndex < oddIndex) {
            oddIndex--;
        }
        if (evenIndex < oddIndex) {
            int temp = arr[evenIndex];
            arr[evenIndex] = arr[oddIndex];
            arr[oddIndex] = temp;
        }
    }
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    rearrangeEvenOdd(arr, n);

    for (int i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    return 0;
}
```

---

37. Find the smallest subarray with sum greater than a given value

Input-Output Format

makefile

CopyEdit

```
Input:
6 8
1 4 45 6 10 19

Output:
3
```

Solution

c

CopyEdit

```
#include <stdio.h>

int smallestSubarrayWithSum(int arr[], int n, int x) {
    int minLength = n + 1;
    int start = 0, end = 0, sum = 0;
    while (end < n) {
        sum += arr[end];
        while (sum > x) {
            minLength = (end - start + 1) < minLength ? (end -
start + 1) : minLength;
            sum -= arr[start];
            start++;
        }
        end++;
    }

    return minLength;
}

int main() {
    int n, x, arr[100];
    scanf("%d %d", &n, &x);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    int result = smallestSubarrayWithSum(arr, n, x);
    if (result == n + 1)
        printf("No such subarray\n");
    else
        printf("%d\n", result);

    return 0;
}
```

---

38. Find the maximum product subarray

Input-Output Format
makefile
CopyEdit

```
Input:
6
1 -2 -3 0 4 -1

Output:
96
```

Solution

c

CopyEdit

```c
#include <stdio.h>

int maxProductSubarray(int arr[], int n) {
    int maxProd = arr[0], minProd = arr[0], result = arr[0];

    for (int i = 1; i < n; i++) {
        if (arr[i] < 0) {
            int temp = maxProd;
            maxProd = minProd;
            minProd = temp;
        }

        maxProd = (arr[i] > maxProd * arr[i]) ? arr[i] : maxProd
* arr[i];
        minProd = (arr[i] < minProd * arr[i]) ? arr[i] : minProd
* arr[i];

        result = (maxProd > result) ? maxProd : result;
    }

    return result;
}

int main() {
    int n, arr[100];
    scanf("%d", &n);
    for (int i = 0; i < n; i++)
        scanf("%d", &arr[i]);

    printf("%d\n", maxProductSubarray(arr, n));
    return 0;
}
```

---

39. Find the longest common prefix among an array of strings

Input-Output Format

makefile

CopyEdit

```
Input:
4
flower flow flight

Output:
fl
```

Solution

c

CopyEdit

```
#include <stdio.h>
#include <string.h>

void longestCommonPrefix(char arr[][100], int n) {
    if (n == 0) {
        printf("\n");
        return;
    }

    int minLen = strlen(arr[0]);
    for (int i = 1; i < n; i++) {
        minLen = (minLen < strlen(arr[i])) ? minLen :
strlen(arr[i]);
    }

    int i = 0;
    while (i < minLen) {
        char ch = arr[0][i];
        for (int j = 1; j < n; j++) {
            if (arr[j][i] != ch) {
                printf("%.*s\n", i, arr[0]);
                return;
            }
        }
        i++;
    }

    printf("%.*s\n", i, arr[0]);
}

int main() {
    int n;
    scanf("%d", &n);
    char arr[n][100];
    for (int i = 0; i < n; i++)
        scanf("%s", arr[i]);

    longestCommonPrefix(arr, n);
    return 0;
}
```

---

40. Merge two sorted arrays

Input-Output Format
makefile

CopyEdit

```
Input:
3 3
1 3 5
2 4 6

Output:
1 2 3 4 5 6
```

Solution

c

CopyEdit

```
#include <stdio.h>

void mergeSortedArrays(int arr1[], int arr2[], int n1, int n2) {
    int i = 0, j = 0;
    while (i < n1 && j < n2) {
        if (arr1[i] < arr2[j]) {
            printf("%d ", arr1[i]);
            i++;
        } else {
            printf("%d ", arr2[j]);
            j++;
        }
    }

    while (i < n1) {
        printf("%d ", arr1[i]);
        i++;
    }

    while (j < n2) {
        printf("%d ", arr2[j]);
        j++;
    }
    printf("\n");
}

int main() {
    int n1, n2, arr1[100], arr2[100];
    scanf("%d %d", &n1, &n2);
    for (int i = 0; i < n1; i++)
        scanf("%d", &arr1[i]);
    for (int i = 0; i < n2; i++)
        scanf("%d", &arr2[i]);

    mergeSortedArrays(arr1, arr2, n1, n2);
    return 0;
}
```