



main.c



Run

Output

Clear

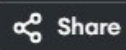
```
11 }
12 char* firstPalindromicString(char words[][20], int size) {
13     for (int i = 0; i < size; i++) {
14         if (isPalindrome(words[i])) {
15             return words[i];
16         }
17     }
18     return "";
19 }
20 int main() {
21     char words[5][20] = {"abc", "car", "ada", "racecar", "cool"};
22     int size = 5;
23
24     char* result = firstPalindromicString(words, size);
25     if (strlen(result) != 0) {
26         printf("First palindromic string: %s\n", result);
27     } else {
28         printf("No palindromic string found.\n");
29     }
30     return 0;
31 }
```

```
/tmp/TfD6AD2WE4.o
First palindromic string: ada

=== Code Execution Successful ===
```



```
main.cpp
12 for (int i = 0; i < m; i++) {
13     for (int j = 0; j < n; j++) {
14         if (nums2[i] == nums1[j]) {
15             answer2++;
16             break;
17         }
18     }
19 }
20 result[0] = answer1;
21 result[1] = answer2;
22 }
23 int main() {
24     int nums1[] = {4,2, 3, 2,1};
25     int nums2[] = {1,3,2, 2,5,6};
26     int n = sizeof(nums1) / sizeof(nums1[0]);
27     int m = sizeof(nums2) / sizeof(nums2[0]);
28     int result[2];
29     calculateAnswers(nums1, n, nums2, m, result);
30     printf("[%d, %d]\n", result[0], result[1]);
31
32     return 0;
33 }
```



Run

Output

Clear

```
/tmp/7osEex7vgW.o
[4, 4]

=== Code Execution Successful ===
```

main.cpp

Run

Share

```
1 #include <stdio.h>
2 #include <string.h>
3 #define MAX 100001
4 int sumOfSquaresOfDistinctCounts(int nums[], int n) {
5     int result = 0;
6     for (int i = 0; i < n; i++) {
7         int freq[MAX] = {0};
8         int distinct_count = 0;
9         for (int j = i; j < n; j++) {
10             if (freq[nums[j]] == 0) {
11                 distinct_count++;
12             }
13             freq[nums[j]]++;
14             result += distinct_count * distinct_count;
15         }
16     }
17 }
18
19 return result;
20 }
21 int main() {
22     int nums[] = {1, 2, 1};
```

Output

Clear

/tmp/N915ElC1CV.o

The sum of the squares of distinct counts of all subarrays is: 15

=== Code Execution Successful ===



main.c



Run

Output

Clear

```
1 #include <stdio.h>
2 int main() {
3     int nums[] = {3, 1, 2, 2, 1, 3};
4     int k = 2;
5     int n = 7;
6     int count = 0;
7
8     for (int i = 0; i < n; i++) {
9         for (int j = i + 1; j < n; j++) {
10            if (nums[i] == nums[j] && (i * j) % k == 0) {
11                count++;
12            }
13        }
14    }
15
16    printf("%d\n", count);
17    return 0;
18 }
19
20
```

/tmp/gG74gEmJM3.o

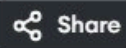
1 2 5 5 6 9

=== Code Execution Successful ===





main.c



Run

Output

Clear

```
1 #include <stdio.h>
2
3 int main() {
4     int nums[] = {-10, 2, 3, -4, 5};
5     int n = 5;
6     int max = nums[0];
7
8     for (int i = 1; i < n; i++) {
9         if (nums[i] > max) {
10             max = nums[i];
11         }
12     }
13
14     printf("%d\n", max);
15     return 0;
16 }
```

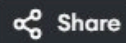
/tmp/u8EdPvgEWY.o

5

=== Code Execution Successful ===



main.c



Run

Output

Clear

```
1 #include <stdio.h>
2
3 void swap(int *a, int *b) {
4     int temp = *a;
5     *a = *b;
6     *b = temp;
7 }
8
9 int partition(int arr[], int low, int high) {
10     int pivot = arr[high];
11     int i = low - 1;
12
13     for (int j = low; j < high; j++) {
14         if (arr[j] < pivot) {
15             i++;
16             swap(&arr[i], &arr[j]);
17         }
18     }
19     swap(&arr[i + 1], &arr[high]);
20     return i + 1;
21 }
22
```

/tmp/PRQfJSEmdK.o

Sorted List: 1 1 3 4 5

Maximum element: 5

=== Code Execution Successful ===

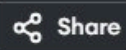


Search

ENG  
IN12:21 PM  
10/8/2024



main.c



Run

Output

Clear

```
1 #include <stdio.h>
2
3 int main() {
4     int nums[] = {1, 2, 2, 3, 4, 4, 5};
5     int n = 7;
6     int unique[100];
7     int unique_count = 0;
8     int is_unique;
9     for (int i = 0; i < n; i++) {
10         is_unique = 1;
11         for (int j = 0; j < unique_count; j++) {
12             if (nums[i] == unique[j]) {
13                 is_unique = 0;
14                 break;
15             }
16         }
17         if (is_unique) {
18             unique[unique_count] = nums[i];
19             unique_count++;
20         }
21     }
22     printf("Unique elements: ");
```

```
/tmp/gcyFoDpVRP.o
Unique elements: 1 2 3 4 5
```

```
=== Code Execution Successful ===
```



```
main.c
10     arr[j+1] = temp;
11 }
12 }
13 }
14 }
15
16 void printArray(int arr[], int size) {
17     for (int i = 0; i < size; i++)
18         printf("%d ", arr[i]);
19     printf("\n");
20 }
21
22 int main() {
23     int arr[] = {64, 34, 25, 12, 22, 11, 90};
24     int n = sizeof(arr)/sizeof(arr[0]);
25     bubbleSort(arr, n);
26     printf("Sorted array: \n");
27     printArray(arr, n);
28     return 0;
29 }
30
```



Run

Output

Clear

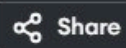
```
/tmp/R3EHCA9wdb.o
Sorted array:
11 12 22 25 34 64 90
```

```
=== Code Execution Successful ===
```





main.c



Run

Output

Clear

```
1 #include <stdio.h>
2 int binarySearch(int arr[], int size, int x) {
3     int left = 0, right = size - 1;
4     while (left <= right) {
5         int mid = left + (right - left) / 2;
6
7         if (arr[mid] == x) {
8             return mid;
9         }
10        if (arr[mid] < x) {
11            left = mid + 1;
12        } else {
13            right = mid - 1;
14        }
15    }
16    return -1;
17 }
18 int main() {
19     int arr[] = { -9, 3, 4, 6, 8, 9, 10, 30 };
20     int size = sizeof(arr) / sizeof(arr[0]);
21     int key1 = 10;
22     int key2 = 100;
```

```
/tmp/HBVlj5wiQ1.o
Element 10 is found at position 6
Element 100 is not found

=== Code Execution Successful ===
```



main.c



Share

Run

Output

Clear

```
1 #include <stdio.h>
2 void merge(int arr[], int left, int mid, int right) {
3     int i, j, k;
4     int n1 = mid - left + 1;
5     int n2 = right - mid;
6     int L[n1], R[n2];
7     for (i = 0; i < n1; i++)
8         L[i] = arr[left + i];
9     for (j = 0; j < n2; j++)
10        R[j] = arr[mid + 1 + j];
11    i = 0;
12    j = 0;
13    k = left;
14    while (i < n1 && j < n2) {
15        if (L[i] <= R[j]) {
16            arr[k] = L[i];
17            i++;
18        } else {
19            arr[k] = R[j];
20            j++;
21        }
22        k++;

```

/tmp/gG74gEmJM3.o

1 2 5 5 6 9

=== Code Execution Successful ===

