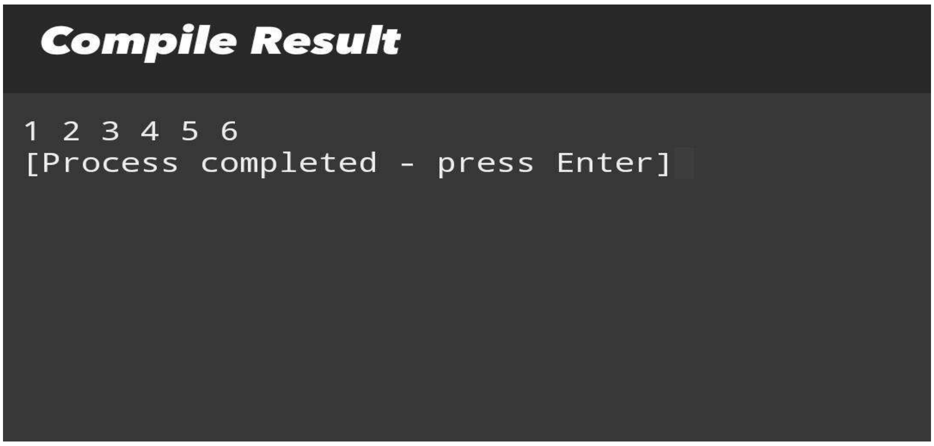
1. Insert an element at the end of an array

```
#include <stdio.h>
void insert(int arr[], int *size, int element) {
    arr[*size] = element;
    (*size)++;
}
int main() {
    int arr[100] = {1, 2, 3, 4, 5};
    int size = 5;
    int element = 6;
    insert(arr, &size, element);
    for (int i = 0; i < size; i++) {
        printf("%d ", arr[i]);
    }
    return 0;
}
```

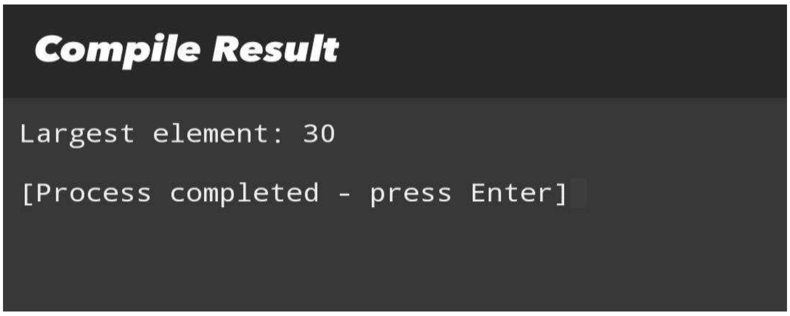


```
1 2 3 4 5 6
[Process completed - press Enter]
```

2. Find largest element in array

```
#include <stdio.h>
int Largest(int arr[], int size) {
   int max = arr[0];
   for (int i = 1; i < size; i++) {
       if (arr[i] > max) {
           max = arr[i];
       }
   }
   return max;
}

int main() {
   int arr[] = {10, 20, 5, 30, 15};
   int size = sizeof(arr) / sizeof(arr[0]);
   printf(”Largest element: %d\n”, Largest(arr, size));
   return 0;
}
```



Compile Result

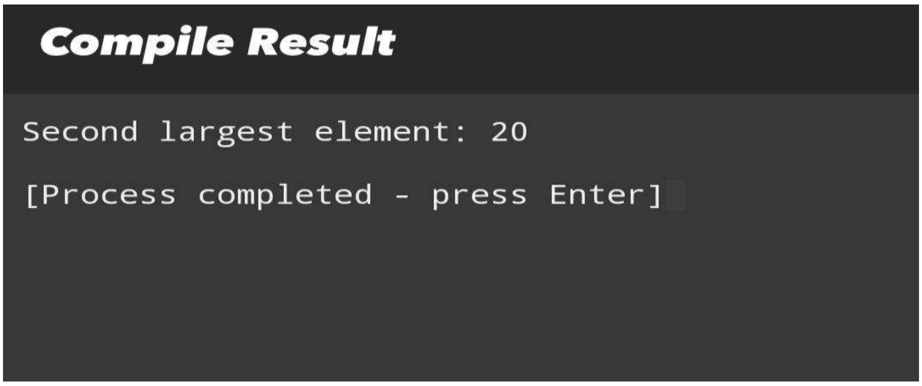
```
Largest element: 30

[Process completed - press Enter]
```

3. Find second largest element

```
#include <stdio.h>
int SecondLargest(int arr[], int size) {
   int first = -1, second = -1;
   for (int i = 0; i < size; 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 arr[] = {10, 20, 5, 30, 15};
   int size = sizeof(arr) / sizeof(arr[0]);
   printf(”Second largest element: %d\n”, SecondLargest(arr, size));
   return 0;
}
```

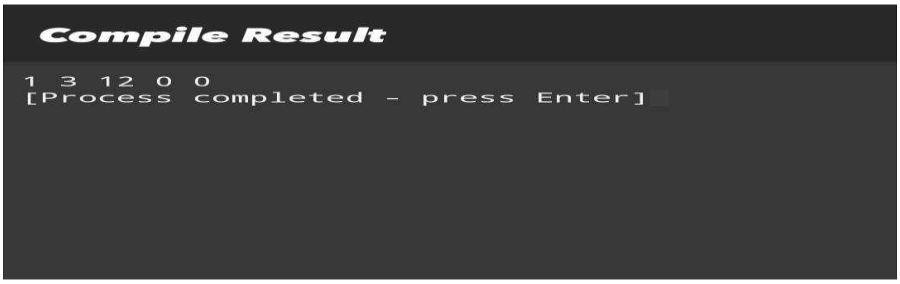


```
Second largest element: 20
[Process completed - press Enter]
```

4. Move all zeros to end

```
#include <stdio.h>
void moveZeros(int arr[], int size) {
   int count = 0;
   for (int i = 0; i < size; i++) {
       if (arr[i] != 0) {
           arr[count++] = arr[i];
       }
   }
   while (count < size) {
       arr[count++] = 0;
   }
}

int main() {
   int arr[] = {0, 1, 0, 3, 12};
   int size = sizeof(arr) / sizeof(arr[0]);
   moveZeros(arr, size);
   for (int i = 0; i < size; i++) {
       printf(”%d “, arr[i]);
   }
   return 0;
}
```

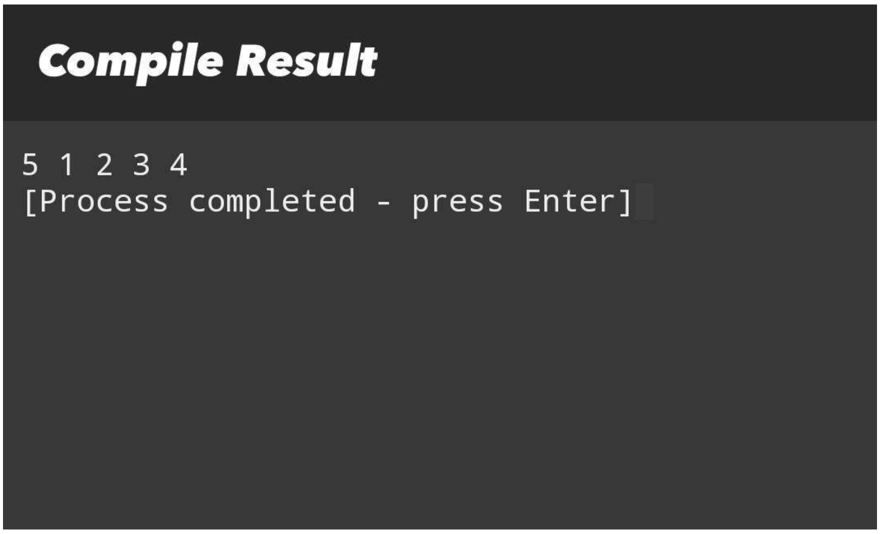


```
1 3 12 0 0
[Process completed - press Enter]
```

5. Rotate array by one

```
#include <stdio.h>
void rotateByOne(int arr[], int size) {
    int temp = arr[size - 1];
    for (int i = size - 1; i > 0; i--) {
        arr[i] = arr[i - 1];
    }
    arr[0] = temp;
}

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int size = sizeof(arr) / sizeof(arr[0]);
    rotateByOne(arr, size);
    for (int i = 0; i < size; i++) {
        printf(”%d “, arr[i]);
    }
    return 0;
}
```



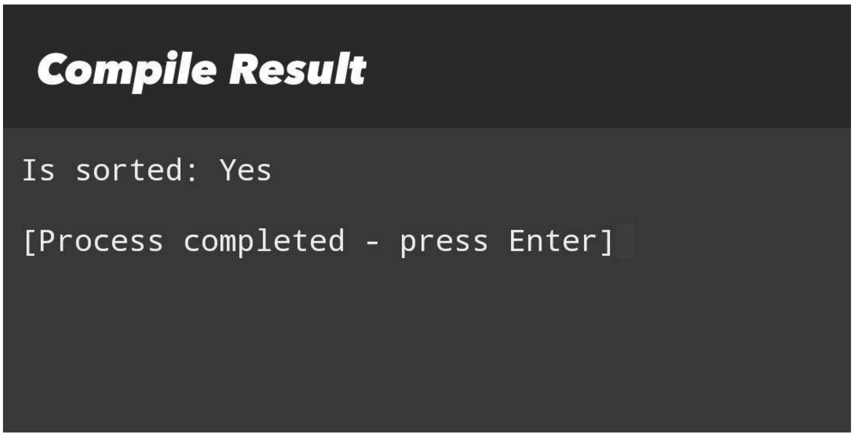
Compile Result

```
5 1 2 3 4
[Process completed - press Enter]
```

6. Check if array is sorted

```
#include <stdio.h>
#include <stdbool.h>
bool Sorted(int arr[], int size) {
    for (int i = 0; i < size - 1; i++) {
        if (arr[i] > arr[i + 1]) {
            return false;
        }
    }
    return true;
}

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int size = sizeof(arr) / sizeof(arr[0]);
    printf(”Is sorted: %s\n”, Sorted(arr, size) ? “Yes” : “No”);
    return 0;
}
```



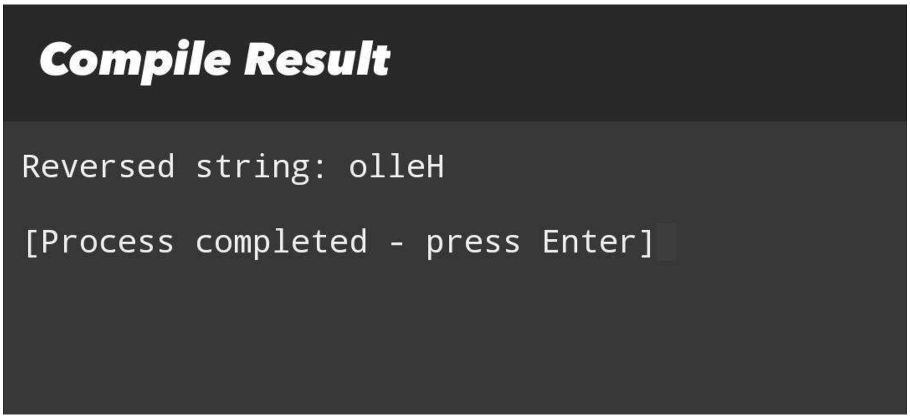
```
Is sorted: Yes

[Process completed - press Enter]
```

7. Reverse a string

```
#include <stdio.h>
#include <string.h>
void reverseString(char str[]) {
   int n = strlen(str);
   for (int i = 0; i < n / 2; i++) {
       char temp = str[i];
       str[i] = str[n - 1 - i];
       str[n - 1 - i] = temp;
   }
}

int main() {
   char str[] = “Hello”;
   reverseString(str);
   printf(”Reversed string: %s\n”, str);
   return 0;
}
```



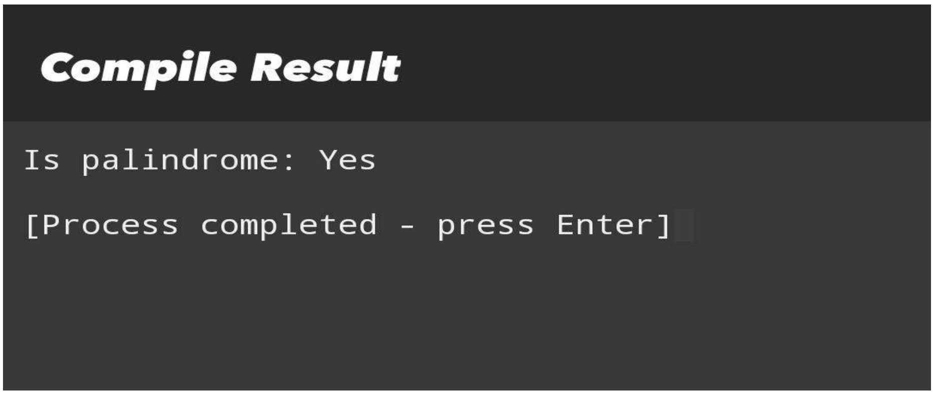
```
Reversed string: olleH

[Process completed - press Enter]
```

8. Check if string is palindrome

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
bool Palindrome(char str[]) {
    int left = 0;
    int right = strlen(str) - 1;
    while (left < right) {
        if (str[left] != str[right]) {
            return false;
        }
        left++;
        right--;
    }
    return true;
}

int main() {
    char str[] = “madam”;
    printf(”Is palindrome: %s\n”, Palindrome(str) ? “Yes” : “No”);
    return 0;
}
```



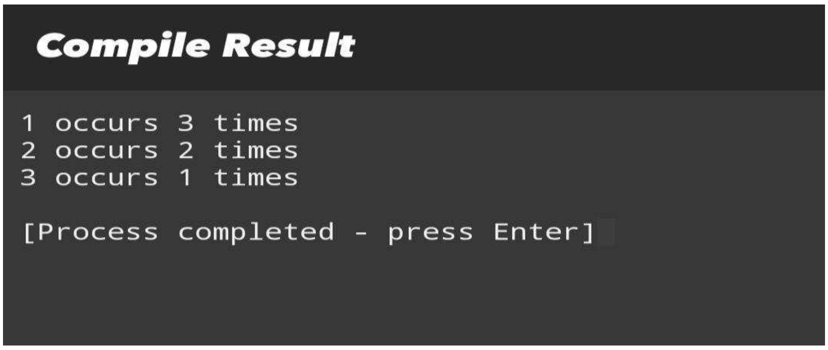
```
Is palindrome: Yes

[Process completed - press Enter]
```

9. Count frequency of array elements

```
#include <stdio.h>
void countFrequency(int arr[], int size) {
    for (int i = 0; i < size; i++) {
        int count = 1;
        if (arr[i] == -1) continue;
        for (int j = i + 1; j < size; j++) {
            if (arr[i] == arr[j]) {
                count++;
                arr[j] = -1;
            }
        }
        printf("%d occurs %d times\n", arr[i], count);
    }
}

int main() {
    int arr[] = {1, 2, 3, 1, 2, 1};
    int size = sizeof(arr) / sizeof(arr[0]);
    countFrequency(arr, size);
    return 0;
}
```



Compile Result

```
1 occurs 3 times
2 occurs 2 times
3 occurs 1 times

[Process completed - press Enter]
```

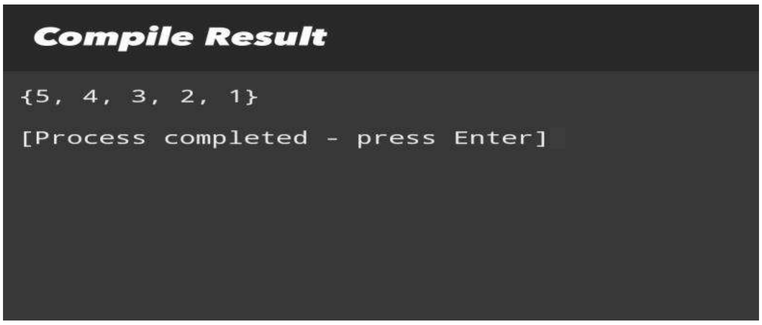
10. Reverse an array

```
#include <stdio.h>

void reverseArray(int arr[], int size) {
    for (int i = 0; i < size / 2; i++) {
        int temp = arr[i];
        arr[i] = arr[size - 1 - i];
        arr[size - 1 - i] = temp;
    }
}

int main() {
    int arr[] = {1, 2, 3, 4, 5};
    int size = sizeof(arr) / sizeof(arr[0]);
    reverseArray(arr, size);

    printf(”{”);
    for (int i = 0; i < size; i++) {
        printf("%d%s", arr[i], (i < size - 1) ? "," : "");    }
    printf(”}\n”);
    return 0;
}
```



Compile Result

```
{5, 4, 3, 2, 1}
[Process completed - press Enter]
```