Array Reversal

RATE THIS CHALLENGE

Given an array, of size n, reverse it.

Example: If array, $arr = [1, 2, 3, 4, 5]$, after reversing it, the array should be, $arr = [5, 4, 3, 2, 1]$.

Input Format

The first line contains an integer, n, denoting the size of the array. The next line contains n space-separated integers denoting the elements of the array.

Constraints

$1 \leq n \leq 1000$

$1 \leq arr_i \leq 1000$, where arr_i is the i^{th} element of the array.

Output Format

The output is handled by the code given in the editor, which would print the array.

Sample Input 0

```
6
16 13 7 2 1 12
```

Sample Output 0

```
12 1 2 7 13 16
```

Explanation 0

Given array, arr = $[16, 13, 7, 2, 1, 12]$. After reversing the array, arr = $[12, 1, 2, 7, 13, 16]$

Sample Input 1

```
7
1 13 15 20 12 13 2
```

Sample Output 1

```
2 13 12 20 15 13 1
```

Sample Input 2

```
8
15 5 16 15 17 11 5 11
```

Sample Output 2

```
11 5 11 17 15 16 5 15
```

Change Theme Language: C

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

int main() {
    int n;

    // Read the number of elements
    scanf("%d", &n);

    // Dynamically allocate memory for the array
    int *arr = (int*)malloc(n * sizeof(int));

    // Check if memory allocation was successful
    if (arr == NULL) {
        printf("Memory allocation failed\n");
        return 1;
    }

    // Read elements into the array
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Reverse the array
    for (int i = 0; i < n / 2; i++) {
        int temp = arr[i];
        arr[i] = arr[n - i - 1];
        arr[n - i - 1] = temp;
    }

    // Print the reversed array
```

Line: 42 Col: 1

Upload Code as File Test against custom input Run Code Submit Code