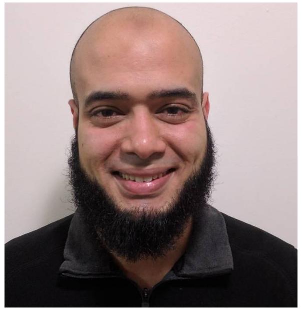
C++ Programming

1D Arrays Practice 2

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

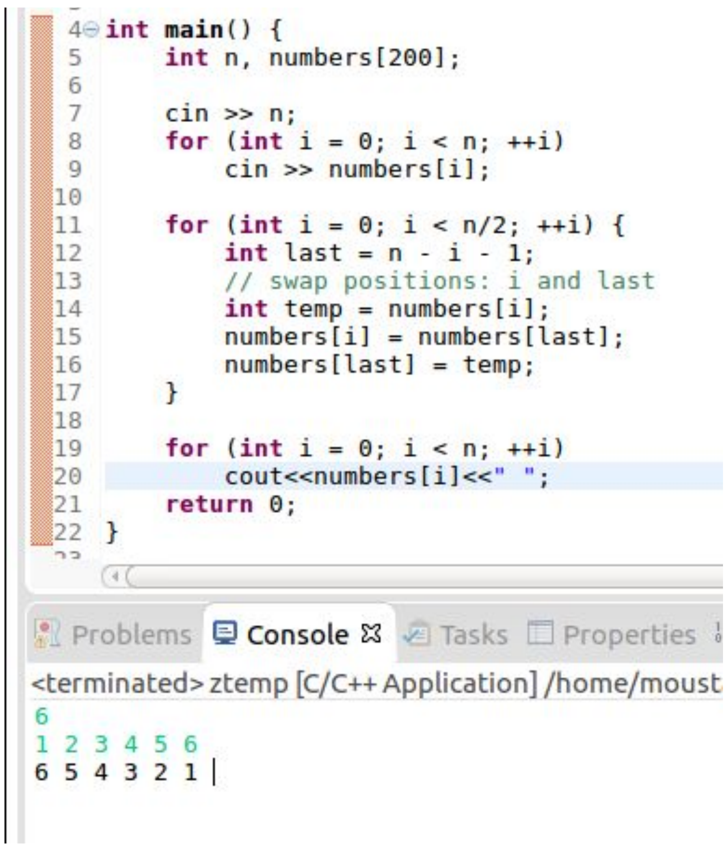
Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)



Practice: Reverse in place

- Read an Integer N, then read N <= 200 integers.
    - In-place: Change the current array, don’t use 2 arrays
- Simple idea: Iterate from the begin and end in same time
    - Swap the 2 positions
    - Do this tell the middle only
- Let say array is 1 2 3 4 5 6 7 8
    - Step 1: swap (1, 8) ⇒ 8 2 3 4 5 6 7 1
    - Step 2: swap (2, 7) ⇒ 8 7 3 4 5 6 2 1
    - Step 3: swap (3, 6) ⇒ 8 7 6 4 5 3 2 1
    - Step 4: swap (4, 6) ⇒ 8 7 6 5 4 3 2 1
        - Stop after n/2 steps

Practice: Reverse in place



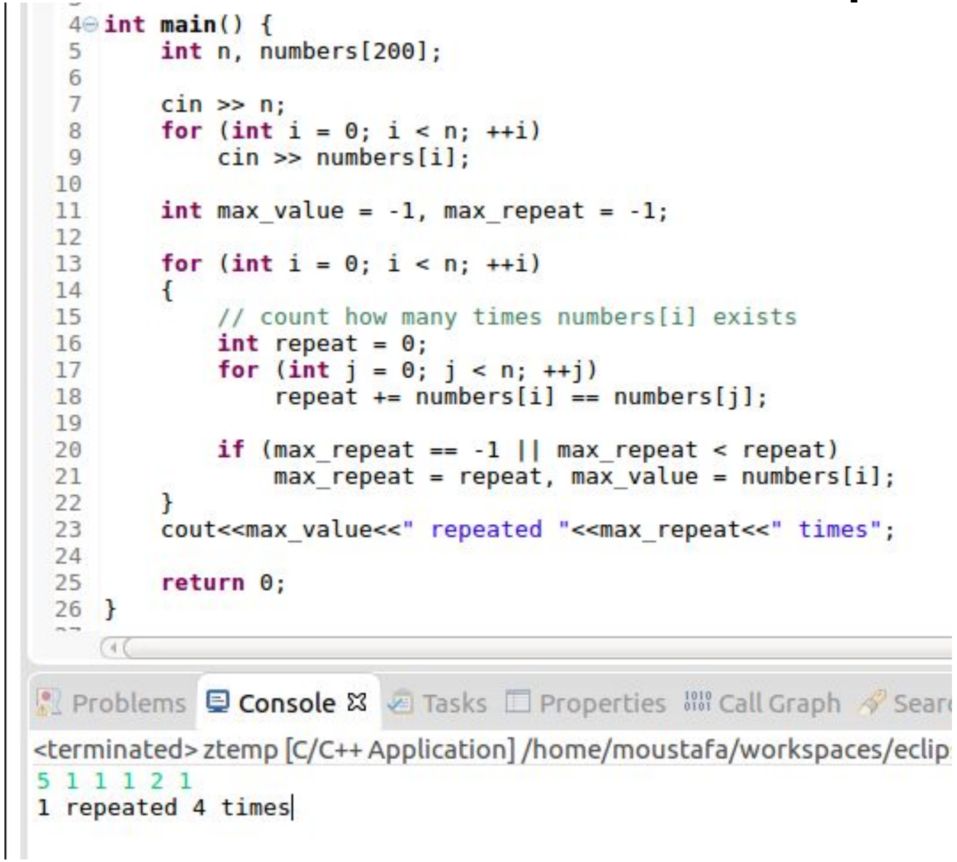
```
int main() {
    int n, numbers[200];

    cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> numbers[i];

    for (int i = 0; i < n/2; ++i) {
        int last = n - i - 1;
        // swap positions: i and last
        int temp = numbers[i];
        numbers[i] = numbers[last];
        numbers[last] = temp;
    }

    for (int i = 0; i < n; ++i)
        cout<<numbers[i]<<" ";
    return 0;
}
```

```
<terminated> ztemp [C/C++ Application] /home/moust
6
1 2 3 4 5 6
6 5 4 3 2 1
```

Practice: Find most frequent number

- Read an Integer N, then read N <= 200 integers. Find the value that repeated the most number of times.
    - Each integer is 0 <= integer <= 150
- Example for array: 1 2 1 3 1 5 5
    - 1 repeated 3 times: the largest
    - 2 repeated 1 time
    - 5 repeated 2 times
- Stop video and think

Practice: Find most frequent number

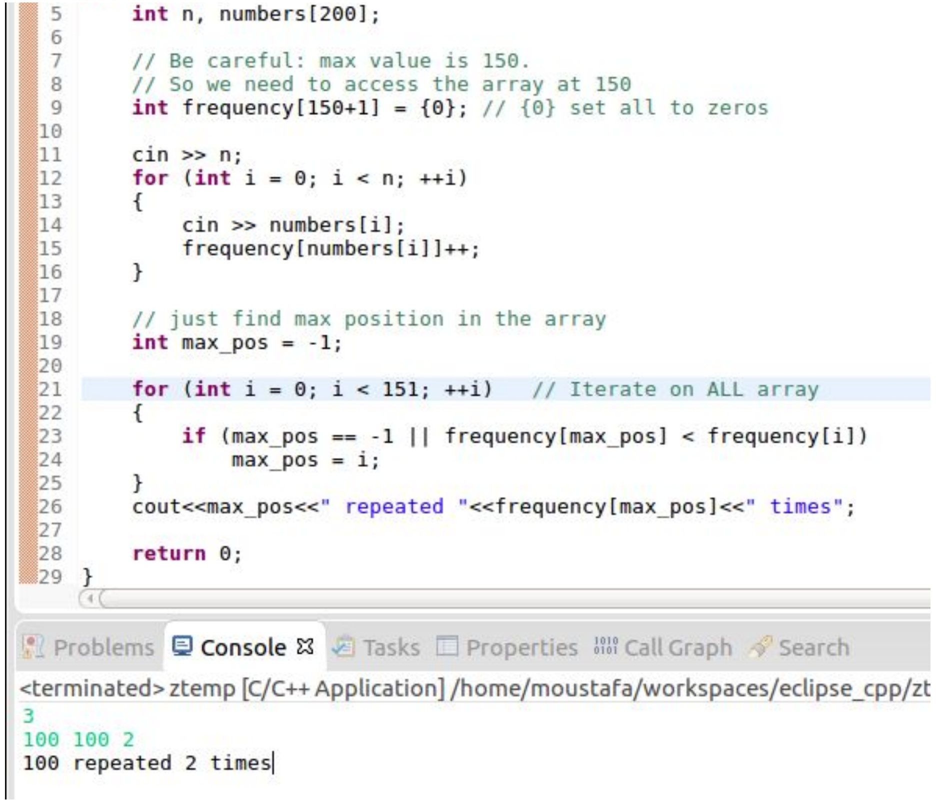


```
int main() {
    int n, numbers[200];

    cin >> n;
    for (int i = 0; i < n; ++i)
        cin >> numbers[i];

    int max_value = -1, max_repeat = -1;

    for (int i = 0; i < n; ++i)
    {
        // count how many times numbers[i] exists
        int repeat = 0;
        for (int j = 0; j < n; ++j)
            repeat += numbers[i] == numbers[j];

        if (max_repeat == -1 || max_repeat < repeat)
            max_repeat = repeat, max_value = numbers[i];
    }
    cout<<max_value<<" repeated "<<max_repeat<<" times";

    return 0;
}
```

```
<terminated> ztemp [C/C++ Application] /home/moustafa/workspaces/eclip
5 1 1 1 2 1
1 repeated 4 times
```

- One easy idea
- For each number, count how many times it in the array. Find maximum of them
- Disadvantage: nested loops (much processing)
- Can you do in 1 loop only?
    - Hint: use another array

Practice: Find most frequent number - FASTER



```
    int n, numbers[200];

    // Be careful: max value is 150.
    // So we need to access the array at 150
    int frequency[150+1] = {0}; // {0} set all to zeros

    cin >> n;
    for (int i = 0; i < n; ++i)
    {
        cin >> numbers[i];
        frequency[numbers[i]]++;
    }

    // just find max position in the array
    int max_pos = -1;

    for (int i = 0; i < 151; ++i)   // Iterate on ALL array
    {
        if (max_pos == -1 || frequency[max_pos] < frequency[i])
            max_pos = i;
    }
    cout<<max_pos<<" repeated "<<frequency[max_pos]<<" times";

    return 0;
}
```

```
<terminated> ztemp [C/C++ Application] /home/moustafa/workspaces/eclipse_cpp/zt
3
100 100 2
100 repeated 2 times
```

- Let’s use another array
- We will use a trick called **frequency array**
  - We think of the index as value
  - If we have M values, create array of M+1 values
- Iterate on the array and increment each time you meet a number
- Find max in the array

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."