C++ Programming

1D Arrays Practice 1

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: Find first and 2nd Maximum values

- Read an Integer N (< 200), then read N (distinct) integers. Find the maximum and 2nd maximum values
- Input: 5 10 20 3 30 7 ⇒ Output 30 20
  - 30 is the maximum in the array
  - If we removed it, the next maximum is 20
- Stop the video and code it

Practice: Find first and 2nd Maximum values

```cpp
int main() {
    int n, numbers[200];    // max expected size

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

    int maximum_idx = 0;
    for (int i = 1; i < n; ++i)
        if (numbers[maximum_idx] < numbers[i])
            maximum_idx = i;

    int max1 = numbers[maximum_idx];
    numbers[maximum_idx] = -1000000;    // assume good value

    maximum_idx = 0;    // same code again
    for (int i = 1; i < n; ++i)
        if (numbers[maximum_idx] < numbers[i])
            maximum_idx = i;

    int max2 = numbers[maximum_idx];
    cout << max1 << " " << max2;
    return 0;
}
```

Problems | Console | Tasks | Properties | Call Graph | Search

```
<terminated> ztemp [C/C++ Application] /home/moustafa/workspaces/eclipse_c
5  10 20 3 30 7
30 20
```

- Easy idea
- Find first maximum (idx)
  - Mark with very small value
- Find again first maximum
  - This is now the 2nd maximum
- Disadvantages
  - Need to loop twice
  - Need good value for -ve
    - Workaround: ignore previous position

Practice: Find first and 2nd Maximum values

```
4  int main() {
5      int n, numbers[200];    // max expected size
6
7      cin >> n;
8      for (int i = 0; i < n; ++i)
9          cin >> numbers[i];
10
11     int max1, max2;
12     if (numbers[0] >= numbers[1])
13         max1 = numbers[0], max2 = numbers[1];
14     else
15         max1 = numbers[1], max2 = numbers[0];
16
17     for (int i = 2; i < n; ++i)
18         if (max1 <= numbers[i])
19             max2 = max1, max1 = numbers[i];
20         else if (max2 < numbers[i])
21             max2 = numbers[i];
22
23     cout << max1 << " " << max2;
24     return 0;
25 }
26
```

- Maintain 2 variables for the 2 maximums
- Iterate on the array and update together
- Say we have so far 20 10
  - And we found value 30
  - Now we should be 30 20
- Say we have so far 20 10
  - And we found value 15
  - Now we should be 20 15

Practice: Find pair values of maximum sum

- Read an Integer N, then read N <= 200 (**distinct**) integers. Find a pair of numbers (e.g. 2 different indices) whose sum is maximum
- Input: 5 2 10 3 50 15 ⇒ 65 (from 50 + 15)
- Stop the video and code it

Practice: Find pair of max sum - Buggy

```
4  int main() {
5      int n, numbers[200];
6
7      cin >> n;
8      for (int i = 0; i < n; ++i)
9          cin >> numbers[i];
10
11     int idx1 = -1, idx2 = -1;
12
13     for (int i = 0; i < n; ++i) {
14         for (int j = 0; j < n; ++j) {
15             if (idx1 == -1)
16                 idx1 = i, idx2 = j;
17             else if (numbers[idx1] + numbers[idx2] <
18                      numbers[i] + numbers[j])
19                 idx1 = i, idx2 = j;
20         }
21     }
22     cout<<numbers[idx1]<<" "<<numbers[idx2];
23
24     return 0;
25 }
26
```

- Let's just do 2 nested loops to find the pair
  - There is a bug
  - Also half of operations is useless!

Practice: Find pair of max sum - Fixed

```
4  int main() {
5      int n, numbers[200];
6
7      cin >> n;
8      for (int i = 0; i < n; ++i)
9          cin >> numbers[i];
10
11     int idx1 = -1, idx2 = -1;
12
13     for (int i = 0; i < n; ++i) {
14         for (int j = i+1; j < n; ++j) {
15             if (idx1 == -1)
16                 idx1 = i, idx2 = j;
17             else if (numbers[idx1] + numbers[idx2] <
18                     numbers[i] + numbers[j])
19                 idx1 = i, idx2 = j;
20         }
21     }
22     cout<<numbers[idx1]<<" "<<numbers[idx2];
23
24     return 0;
25 }
```

Problems | Console | Tasks | Properties | Call Graph

erminated> ztemp [C/C++ Application] /home/moustafa/workspaces/

2 10 3 50 15

9 15

- Trick: Start j from i+1
  - Avoid duplicate bug
  - Avoid duplicate processing
    - We test positions (2, 4) and then test (4, 2) which is same locations!
- This is very inefficient code!
  - Can you do it using a single loop!
    - Hint: Simple observation

Practice: Find pair of max sum - FASTER

- Simply, the pair of maximum sum must come from **the maximum value and the 2nd maximum value**
- Use the code we explained, get them and sum them
- Think more ⇒ Code efficient

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."