Câu hỏi 1

Đúng

Đạt điểm 10,00 trên 10,00

Hai mảng giống nhau là hai mảng có cùng số phần tử và các phần tử ở vị trí giống nhau là như nhau.

Yêu cầu

Viết chương trình nhận vào từ bàn phím hai mảng và trả lời hai mảng có giống nhau không.

Input

- Dòng đầu tiên chứa số nguyên n
- Hai dòng tiếp theo lần lượt là n số nguyên thuộc hai mảng a và mảng b

Ouput

- Nếu hai mảng giống nhau không, in ra "YES", ngược lại in ra "NO".

Gợi ý

1. Cần duyệt mảng từng vị trí.
2. Nếu có một vị trí khác nhau, in ra "NO" và dừng chương trình
3. Nếu duyệt hết mảng mà không tìm ra vị trí nào khác nhau, in ra "YES" và dừng chương trình

Lưu ý

1. Có thể dùng `<vector>` để khai báo mảng
2. Có thể duyệt mảng từ trái qua phải hoặc từ phải qua trái
3. Có thể dùng hàm với tham số là hai mảng cần so sánh và đầu ra có kiểu `bool (true / false)`

For example:

Input	Result
2 4 5 4 5	YES
2 4 4 4 5	NO

Answer: (penalty regime: 0 %)

```cpp
#include <iostream>
#include <vector>
using namespace std;

signed main() {
  int n;
  cin >> n;
  vector<int> a(n), b(n);
  for (int &x : a) cin >> x;
  for (int &x : b) cin >> x;

  for (int i = 0; i < n; ++i) {
    if (a[i] != b[i]) {
      cout << "NO";
      return 0;
    }
  }
  cout << "YES";
}
```

	Input	Expected	Got	
✓	2 4 5 4 5	YES	YES	✓
✓	2 4 4 4 5	NO	NO	✓
✓	4 1 3 5 7 1 3 2 3	NO	NO	✓
✓	5 1 2 3 4 5 1 2 3 4 5	YES	YES	✓
✓	1 1 1	YES	YES	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 2

Đúng

Đạt điểm 10,00 trên 10,00

Điểm tổng kết Lập trình nâng cao của n sinh viên được lưu trong một mảng 1 chiều.

Yêu cầu

Viết chương trình sắp xếp lại bảng điểm trên theo thứ tự giảm dần và in ra màn hình.

Input

- Dòng đầu chứa số nguyên n
- Dòng thứ hai chứa n số thực là điểm của n sinh viên

Ouput

- Một dòng chứa n điểm số đã được sắp xếp theo thứ tự giảm dần, hai số liền nhau cách nhau bằng một dấu cách
- Các con số được làm tròn đến chữ số thập phân thứ hai

Gợi ý

1. Để sắp xếp ta sử dụng hai vòng lặp lồng nhau để duyệt mảng
2. Vòng lặp thứ nhất chỉ số i từ 0 đến $n-1$
3. Vòng lặp thứ hai chỉ số j từ $i+1$ đến $n-1$, tức là $j > i$
4. Tráo đổi hai phần tử thứ i và thứ j nếu chúng chưa được sắp xếp giảm dần

Lưu ý

1. Cách sắp xếp ở phần gợi ý chưa phải cách sắp xếp nhanh nhất có thể
2. Đọc thêm: thư viện `<algorithm>`. Mặc dù mục đích của bài này là viết chương trình để hiểu một cách sắp xếp (như gợi ý là đạt yêu cầu) nhưng sinh viên có thể tìm hiểu thêm cách sử dụng thư viện `<algorithm>` để sắp xếp cũng như nhiều thuật toán khác.

For example:

Input	Result
15 2.91364701067 4.92844392133 0.784151561401 7.45297643936 0.361399762262 4.95631157295 9.83352340969 0.422081509266 7.86714702984 3.89690308818 4.99809841067 4.26767489958 1.34403895566 2.68317658315 8.80729068767	9.83 8.81 7.87 7.45 5.00 4.96 4.93 4.27 3.90 2.91 2.68 1.34 0.78 0.42 0.36

Answer: (penalty regime: 0 %)

```
#include <iostream>
#include <vector>
#include <iomanip>
using namespace std;

signed main() {
  int n;
  cin >> n;
  vector<double> a(n);
  for (double &x : a) cin >> x;

  for (int i = 0; i < n-1; ++i) {
    for (int j = i+1; j < n; ++j) {
      if (a[j] > a[i]) {
        swap(a[i], a[j]);
      }
    }
  }
  cout << fixed << setprecision(2);
  for (double x : a) cout << x << ' ';
}
```

	Input	Expected	Got	
✓	15 2.91364701067 4.92844392133 0.784151561401 7.45297643936 0.361399762262 4.95631157295 9.83352340969 0.422081509266 7.86714702984 3.89690308818 4.99809841067 4.26767489958 1.34403895566 2.68317658315 8.80729068767	9.83 8.81 7.87 7.45 5.00 4.96 4.93 4.27 3.90 2.91 2.68 1.34 0.78 0.42 0.36	9.83 8.81 7.87 7.45 5.00 4.96 4.93 4.27 3.90 2.91 2.68 1.34 0.78 0.42 0.36	✓
✓	15 7.21907452705 0.878577101963 6.03602923324 2.0344845004 7.82063737578 0.0973769359693 9.85169780453 4.99579173751 2.12118676212 6.5749396641 5.4901426112 5.7066330807 1.94822014317 0.486663835645 2.10218967251	9.85 7.82 7.22 6.57 6.04 5.71 5.49 5.00 2.12 2.10 2.03 1.95 0.88 0.49 0.10	9.85 7.82 7.22 6.57 6.04 5.71 5.49 5.00 2.12 2.10 2.03 1.95 0.88 0.49 0.10	✓
✓	64 3.95099465183 6.06838977332 0.0164275631446 1.86036619917 8.87952069096 5.20612836225 3.42043894504 1.1822061393 1.28909831655 3.0122058109 8.4557139917 7.59094093014 6.08204202224 4.61262238477 5.82718301466 0.447329310916 2.1478509568 4.01877005193 8.73892996038 6.76894172826 4.91166571455 3.89899577511 6.51128399693 1.10590894738 3.97633618145 1.02364355 9.1348243227 4.81412062382 0.903557551251 0.219100557646 2.44014397834 7.08629565333 5.4821901029 4.01407442789 9.41956675047 9.15692070719 8.54966376937 7.85165964829 3.86802883902 7.97138659653 5.16006208026 8.65787973474 9.91118920464 9.85744322835 5.88756616857 3.08065988506 5.48382637378 7.4467593248 0.650610983252 3.2452573593 8.48145998543 6.8816624943 4.22617759977 1.86792043726 8.22716898617 2.53837841855 7.55337075504 9.70518094912 1.0896204347 8.80427499457 5.8058084723 4.89467389872 4.66547006462 0.73433507502	9.91 9.86 9.71 9.42 9.16 9.13 8.88 8.80 8.74 8.66 8.55 8.48 8.46 8.23 7.97 7.85 7.59 7.55 7.45 7.09 6.88 6.77 6.51 6.08 6.07 5.89 5.83 5.81 5.48 5.48 5.21 5.16 4.91 4.89 4.81 4.67 4.61 4.23 4.02 4.01 3.98 3.95 3.90 3.87 3.42 3.25 3.08 3.01 2.54 2.44 2.15 1.87 1.86 1.29 1.18 1.11 1.09 1.02 0.90 0.73 0.65 0.45 0.22 0.02	9.91 9.86 9.71 9.42 9.16 9.13 8.88 8.80 8.74 8.66 8.55 8.48 8.46 8.23 7.97 7.85 7.59 7.55 7.45 7.09 6.88 6.77 6.51 6.08 6.07 5.89 5.83 5.81 5.48 5.48 5.21 5.16 4.91 4.89 4.81 4.67 4.61 4.23 4.02 4.01 3.98 3.95 3.90 3.87 3.42 3.25 3.08 3.01 2.54 2.44 2.15 1.87 1.86 1.29 1.18 1.11 1.09 1.02 0.90 0.73 0.65 0.45 0.22 0.02	✓
✓	50 0.0741714086618 5.43687901272 4.07503385353 2.4824629193 8.81970969244 7.75428463969 5.97683102192 8.00579224903 2.93905789007 9.48129815574 0.281702907428 9.74289323701 3.88958106316 3.34491007016 3.56483318362 4.86139136711 4.92545326481 7.24268013018 0.100441367638 2.28883514867 5.74507682197 8.03665952276 1.55448989326 3.717053409 5.80301526722 4.25425101717 0.0906161821488 2.45414969522 7.72282864665 5.92278384375 0.381565547688 9.400300637 9.93231718438 3.47965901516 1.76299262558 1.04882972569 9.55736675903 4.64661229694 0.170985237801 9.28945704467 9.39046872324 9.41032878728 9.01317566388 2.32060740125 5.3059668099 8.03099019768 3.05599863386 6.20095258786 1.56075483891 3.5773748678	9.93 9.74 9.56 9.48 9.41 9.40 9.39 9.29 9.01 8.82 8.04 8.03 8.01 7.75 7.72 7.24 6.20 5.98 5.92 5.80 5.75 5.44 5.31 4.93 4.86 4.65 4.25 4.08 3.89 3.72 3.58 3.56 3.48 3.34 3.06 2.94 2.48 2.45 2.32 2.29 1.76 1.56 1.55 1.05 0.38 0.28 0.17 0.10 0.09 0.07	9.93 9.74 9.56 9.48 9.41 9.40 9.39 9.29 9.01 8.82 8.04 8.03 8.01 7.75 7.72 7.24 6.20 5.98 5.92 5.80 5.75 5.44 5.31 4.93 4.86 4.65 4.25 4.08 3.89 3.72 3.58 3.56 3.48 3.34 3.06 2.94 2.48 2.45 2.32 2.29 1.76 1.56 1.55 1.05 0.38 0.28 0.17 0.10 0.09 0.07	✓

	Input	Expected	Got	
✓	100 6.02363457817 1.03751478679 3.95424895573 4.64900568732 6.15685232504 4.69477931288 4.33045289687 9.92522336289 6.37452834406 4.37406622044 4.54879061203 5.3607432809 8.58396699028 3.17022021645 3.45607804851 7.36153049517 8.32142061282 3.37115848485 0.103793377186 9.84777856167 8.77701821236 3.96936345037 2.68479477243 9.48345075194 1.95830625875 6.26443961343 0.949558579474 4.04772624778 7.35577040304 9.54990329081 9.11916102569 4.85494494196 2.35840339262 2.22828914282 2.08092852132 6.52345163876 8.63687991577 9.12791740576 5.08087893368 7.64565085932 4.42890184999 6.78153014939 1.73978560159 3.77106567011 6.43985504336 3.18426649128 4.5667865619 1.62650108724 7.7146180378 7.88459882156 4.53775202402 1.01008238859 3.45300534204 2.00786312967 7.35295492788 3.31062097221 7.76281552081 8.90183087789 7.84318960999 7.65656776588 8.79804462553 7.71913461401 9.67299678433 2.16311521986 8.56978594934 5.67198280582 1.82764831079 4.81067511068 3.77373580496 0.852715091963 0.427489331616 0.224019853832 1.56351271089 1.70027245321 2.39758975492 0.825901690989 3.42203402741 7.75712495366 9.7978288058 6.54866661693 4.55355101011 4.0751057731 0.346668819026 4.07818823455 6.4086252309 2.91937249902 0.863522553346 6.86891048101 4.34035464693 7.70519903902 4.42834639067 8.01887159755 4.63653370644 5.83722675209 1.00548717073 1.27120910259 0.520381954476 3.8778262136 3.69788997338 2.1005319756	9.93 9.85 9.80 9.67 9.55 9.48 9.13 9.12 8.90 8.80 8.78 8.64 8.58 8.57 8.32 8.02 7.88 7.84 7.76 7.76 7.72 7.71 7.71 7.66 7.65 7.36 7.36 7.35 6.87 6.78 6.55 6.52 6.44 6.41 6.37 6.26 6.16 6.02 5.84 5.67 5.36 5.08 4.85 4.81 4.69 4.65 4.64 4.57 4.55 4.55 4.54 4.43 4.43 4.37 4.34 4.33 4.08 4.08 4.05 3.97 3.95 3.88 3.77 3.77 3.70 3.46 3.45 3.42 3.37 3.31 3.18 3.17 2.92 2.68 2.40 2.36 2.23 2.16 2.10 2.08 2.01 1.96 1.83 1.74 1.70 1.63 1.56 1.27 1.04 1.01 1.01 0.95 0.86 0.85 0.83 0.52 0.43 0.35 0.22 0.10	9.93 9.85 9.80 9.67 9.55 9.48 9.13 9.12 8.90 8.80 8.78 8.64 8.58 8.57 8.32 8.02 7.88 7.84 7.76 7.76 7.72 7.71 7.71 7.66 7.65 7.36 7.36 7.35 6.87 6.78 6.55 6.52 6.44 6.41 6.37 6.26 6.16 6.02 5.84 5.67 5.36 5.08 4.85 4.81 4.69 4.65 4.64 4.57 4.55 4.55 4.54 4.43 4.43 4.37 4.34 4.33 4.08 4.08 4.05 3.97 3.95 3.88 3.77 3.77 3.70 3.46 3.45 3.42 3.37 3.31 3.18 3.17 2.92 2.68 2.40 2.36 2.23 2.16 2.10 2.08 2.01 1.96 1.83 1.74 1.70 1.63 1.56 1.27 1.04 1.01 1.01 0.95 0.86 0.85 0.83 0.52 0.43 0.35 0.22 0.10	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 3

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu:

Viết chương trình nhận đầu vào là số nguyên n và dãy gồm n số nguyên đã được sắp xếp theo thứ tự tăng dần, in ra màn hình dãy với các số riêng biệt sắp xếp theo thứ tự tăng dần (tức là loại bỏ các số lặp lại nhiều lần trong dãy ban đầu).

Input:

- Dòng đầu tiên là độ dài $n, 0 < n \leq 1000$.
- Dòng thứ 2 chứa n số đã được sắp xếp theo thứ tự tăng dần cách nhau bởi dấu cách

Output:

- In ra màn hình dãy với các số không trùng nhau theo thứ tự tăng dần

Gợi ý:

1. Khởi tạo biến chứa giá trị n và nhập giá trị n.
2. khởi tạo mảng, dùng vòng for để nhập mảng.
3. Dùng vòng for để duyệt giá trị của mảng. In ra số tại ví trí i nếu i==0 hoặc số đó khác số ở vị trí trước đó.

Lưu ý:

1. Chú ý trường hợp i==0 nếu so sánh với số tại vị trí i-1 nó sẽ không tồn tại

For example:

Input	Result
3 2 2 3	2 3

Answer: (penalty regime: 0 %)

```
#include <iostream>
#include <vector>
using namespace std;

signed main() {
  int n;
  cin >> n;
  vector<int> a(n);
  for (int &x : a) cin >> x;

  for (int i = 0; i < n-1; ++i) {
    for (int j = i+1; j < n; ++j) {
      if (a[j] < a[i]) {
        swap(a[i], a[j]);
      }
    }
  }
  cout << a[0] << ' ';
  for (int i = 1; i < n; ++i) {
    if (a[i] != a[i-1]) {
      cout << a[i] << ' ';
    }
  }
}
```

	Input	Expected	Got	
✓	3 2 2 3	2 3	2 3	✓
✓	4 1 2 3 4	1 2 3 4	1 2 3 4	✓
✓	10 1 2 2 2 2 2 2 2 2 2	1 2	1 2	✓
✓	6 1 1 1 1 1 1	1	1	✓
✓	10 1 2 2 3 4 4 4 5 6 6	1 2 3 4 5 6	1 2 3 4 5 6	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi **4**

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu:

Cho một mảng gồm n số thực đã được sắp xếp theo thứ tự tăng dần.

Viết chương trình chèn một số thực k vào mảng trên sao cho thứ tự sắp xếp không đổi. In dãy mới ra màn hình.

Kết quả làm tròn đến số thập phân thứ hai.

Input:

- Dòng đầu tiên là độ dài $n, 0 < n \leq 1000$.
- Dòng thứ 2 chứa n số thực cách nhau bởi dấu cách.
- Dòng thứ 3 chứa số thực x cần chèn vào.

Output:

- Dãy $n + 1$ số thực đã được chèn thêm số mới với thứ tự sắp xếp tăng dần và được làm tròn đến số thập phân thứ 2

Gợi ý:

1. Khai báo biến số nguyên để lưu trữ số lượng phần tử và nhập dữ liệu.
2. Khởi tạo mảng, dùng vòng lặp để nhập vào từng phần tử của mảng.
3. Khai báo số thực cần chèn vào và nhập dữ liệu.
4. Khai báo biến chạy i và j
5. Xác định vị trí chèn số thực, dùng vòng for và biến chạy i để duyệt mảng từ đầu đến vị trí đầu tiên lớn hơn số thực x. Dung lệnh `break;` để dừng vòng for.
6. Các số từ vị trí i đến cuối mảng cần được đẩy lùi về sau 1 vị trí. Dùng vòng for với biến chạy j từ cuối mảng đến vị tí i:
   `array[j+1] = array[j];`
7. Cuối cùng gán vị trí phần tử i bằng x

Chú ý:

1. Không gán x vào vị trí i trước khi đẩy lùi các giá trị từ vị trí i về sau 1 đơn vị. Nếu không sẽ mất số ở vị trí i ban đầu.
2. Vòng for dùng để đẩy lùi phải chạy từ cuối đến vị trí i để tránh mất giá trị của số ở vị trí i+1.
3. Các hàm làm tròn là `fixed` và `setprecision` nằm trong thư viện `<iomanip>`

For example:

Input	Result
5 -1.61996539202 0.968616409208 2.12587576231 7.56980530885 11.305029863 -2.50635869654	-2.51 -1.62 0.97 2.13 7.57 11.31

Answer: (penalty regime: 0 %)

```
#include <iostream>
#include <vector>
#include <iomanip>
using namespace std;

signed main() {
  int n;
  cin >> n;
  vector<double> a(n);
  for (double &x : a) cin >> x;
  double k;
  cin >> k;

  if (k > a[n-1]) {
    a.emplace_back(k);
```

```cpp
  } else {
    for (int i = 0; i < n; ++i) {
      if (k < a[i]) {
        a.insert(a.begin() + i, k);
        break;
      }
    }
  }
  cout << fixed << setprecision(2);
  for (double x : a) cout << x << ' ';
}
```

	Input	Expected	Got	
✓	5 -1.61996539202 0.968616409208 2.12587576231 7.56980530885 11.305029863 -2.50635869654	-2.51 -1.62 0.97 2.13 7.57 11.31	-2.51 -1.62 0.97 2.13 7.57 11.31	✓
✓	5 -8.08076929787 -6.47268489558 -5.14371884832 5.37159276481 6.77924146934 4.92793607475	-8.08 -6.47 -5.14 4.93 5.37 6.78	-8.08 -6.47 -5.14 4.93 5.37 6.78	✓

	Input	Expected	Got	
✓	100 -29.3235478438 -20.9220576434 -19.5707065441 -17.6587017695 -16.6010112667 -15.5650729808 -15.1277832776 -14.688239386 -14.5710199344 -14.3617535799 -13.9842876915 -13.7722289965 -13.6004178886 -12.9784142201 -12.5985063419 -11.9309728003 -11.7830388343 -10.9392858707 -10.5730993344 -10.3319478242 -9.7529324221 -9.70585956233 -9.31904905615 -9.29853023349 -8.90478964495 -8.45819998488 -8.04599004429 -8.0346153743 -7.94926280505 -7.88434110905 -7.823574344 -7.60233278572 -7.47951644975 -7.34978811616 -7.29530537726 -5.5483291453 -5.27645382851 -4.97720790435 -4.79224390662 -4.51593634541 -4.08172642756 -3.05511040257 -3.04160723482 -3.03611935862 -3.00474136759 -2.73359106019 -2.4033742979 -2.24439627251 -2.06949064853 -2.05070763335 -1.962416675 -1.84007969027 -0.950603836211 -0.93295132839 -0.703545865189 -0.0813846512361 0.294717489155 0.300739300058 0.651433519179 0.680020167494 0.985637478402 1.10872370933 1.18927209797 1.33507232145 1.56466328236 1.61662772067 1.62655254121 1.79536821921 2.81047748431 2.90318009804 2.97287247349 3.95592581071 4.03032324789 4.08946194936 4.25715883418 4.54529057333 4.90180655733 5.10558105843 5.42592332157 5.91833635651 6.28899263404 6.79682572042 7.00646995056 8.94220748137 9.26186175761 9.35445909702 9.8367929271 10.0933909647 10.1878738423 10.4279078876 10.9471713072 12.2891551075 13.3870984366 13.4821701755 13.5872477108 15.0413983707 17.8201710401 19.059716493 20.8703813875 36.4983253209 12.384094591	-29.32 -20.92 -19.57 -17.66 -16.60 -15.57 -15.13 -14.69 -14.57 -14.36 -13.98 -13.77 -13.60 -12.98 -12.60 -11.93 -11.78 -10.94 -10.57 -10.33 -9.75 -9.71 -9.32 -9.30 -8.90 -8.46 -8.05 -8.03 -7.95 -7.88 -7.82 -7.60 -7.48 -7.35 -7.30 -5.55 -5.28 -4.98 -4.79 -4.52 -4.08 -3.06 -3.04 -3.04 -3.00 -2.73 -2.40 -2.24 -2.07 -2.05 -1.96 -1.84 -0.95 -0.93 -0.70 -0.08 0.29 0.30 0.65 0.68 0.99 1.11 1.19 1.34 1.56 1.62 1.63 1.80 2.81 2.90 2.97 3.96 4.03 4.09 4.26 4.55 4.90 5.11 5.43 5.92 6.29 6.80 7.01 8.94 9.26 9.35 9.84 10.09 10.19 10.43 10.95 12.29 12.38 13.39 13.48 13.59 15.04 17.82 19.06 20.87 36.50	-29.32 -20.92 -19.57 -17.66 -16.60 -15.57 -15.13 -14.69 -14.57 -14.36 -13.98 -13.77 -13.60 -12.98 -12.60 -11.93 -11.78 -10.94 -10.57 -10.33 -9.75 -9.71 -9.32 -9.30 -8.90 -8.46 -8.05 -8.03 -7.95 -7.88 -7.82 -7.60 -7.48 -7.35 -7.30 -5.55 -5.28 -4.98 -4.79 -4.52 -4.08 -3.06 -3.04 -3.04 -3.00 -2.73 -2.40 -2.24 -2.07 -2.05 -1.96 -1.84 -0.95 -0.93 -0.70 -0.08 0.29 0.30 0.65 0.68 0.99 1.11 1.19 1.34 1.56 1.62 1.63 1.80 2.81 2.90 2.97 3.96 4.03 4.09 4.26 4.55 4.90 5.11 5.43 5.92 6.29 6.80 7.01 8.94 9.26 9.35 9.84 10.09 10.19 10.43 10.95 12.29 12.38 13.39 13.48 13.59 15.04 17.82 19.06 20.87 36.50	✓
✓	49 -23.5431900808 -12.0588613104 -11.5916064405 -11.5701991792 -11.0906962311 -9.68567663184 -8.95459247853 -8.87377773467 -8.47757954995 -8.44870030534 -7.91582050361 -7.52596210494 -6.95999263338 -6.54794815863 -6.53130176386 -6.14362476219 -5.46374607615 -5.19706371846 -5.05200211281 -3.73251974818 -3.59699418411 -2.6409843316 -2.63934102685 -2.5966587278 -1.71257031964 -0.813287597101 -0.228237328881 -0.200463094387 0.524665045565 1.15903999384 2.09015526815 3.96450956993 4.06726655975 4.67173176886 4.73155142239 4.99203697466 5.32213332981 6.13138630965 6.45036220859 8.24976236845 8.61593274178 8.99573358031 9.13352011411 9.84991759956 10.0608796271 11.3194883709 12.0711570309 12.6698684794 23.7731949094 -10.0723885197	-23.54 -12.06 -11.59 -11.57 -11.09 -10.07 -9.69 -8.95 -8.87 -8.48 -8.45 -7.92 -7.53 -6.96 -6.55 -6.53 -6.14 -5.46 -5.20 -5.05 -3.73 -3.60 -2.64 -2.64 -2.60 -1.71 -0.81 -0.23 -0.20 0.52 1.16 2.09 3.96 4.07 4.67 4.73 4.99 5.32 6.13 6.45 8.25 8.62 9.00 9.13 9.85 10.06 11.32 12.07 12.67 23.77	-23.54 -12.06 -11.59 -11.57 -11.09 -10.07 -9.69 -8.95 -8.87 -8.48 -8.45 -7.92 -7.53 -6.96 -6.55 -6.53 -6.14 -5.46 -5.20 -5.05 -3.73 -3.60 -2.64 -2.64 -2.60 -1.71 -0.81 -0.23 -0.20 0.52 1.16 2.09 3.96 4.07 4.67 4.73 4.99 5.32 6.13 6.45 8.25 8.62 9.00 9.13 9.85 10.06 11.32 12.07 12.67 23.77	✓

	Input	Expected	Got	
✓	65 -324.729422285 -205.117186089 -185.029923741 -140.310672543 -137.576516757 -119.601870379 -114.162538139 -97.4425888244 -93.4140328426 -71.5632454406 -67.3385346946 -61.9520278881 -55.8514149689 -53.6208457955 -52.6385388995 -50.2654503194 -22.9849096872 -13.1087506386 -12.9808928182 -10.7498135708 -7.27846699963 -6.56697532694 -5.65005854091 -3.92387155238 1.68774719468 1.72730718857 11.2579918837 15.1387028491 18.5261412224 20.0538437757 21.5180770085 22.9549659059 23.4504225558 27.5104345706 30.2306225848 37.9454501842 39.5241527785 41.2805600914 41.4619914647 53.4344233531 56.9268111754 60.301960241 62.4219845083 63.7712491599 64.532227855 66.8217887735 71.1307906739 72.3081025118 74.3199925801 74.3708878858 74.6250885183 76.7729262904 81.258650039 81.5581468751 87.0510685529 88.0174466823 108.572418609 114.132215067 128.580077286 134.901479331 141.708696355 148.798666614 187.454440084 229.622880554 257.659436981 -100.237283266	-324.73 -205.12 -185.03 -140.31 -137.58 -119.60 -114.16 -100.24 -97.44 -93.41 -71.56 -67.34 -61.95 -55.85 -53.62 -52.64 -50.27 -22.98 -13.11 -12.98 -10.75 -7.28 -6.57 -5.65 -3.92 1.69 1.73 11.26 15.14 18.53 20.05 21.52 22.95 23.45 27.51 30.23 37.95 39.52 41.28 41.46 53.43 56.93 60.30 62.42 63.77 64.53 66.82 71.13 72.31 74.32 74.37 74.63 76.77 81.26 81.56 87.05 88.02 108.57 114.13 128.58 134.90 141.71 148.80 187.45 229.62 257.66	-324.73 -205.12 -185.03 -140.31 -137.58 -119.60 -114.16 -100.24 -97.44 -93.41 -71.56 -67.34 -61.95 -55.85 -53.62 -52.64 -50.27 -22.98 -13.11 -12.98 -10.75 -7.28 -6.57 -5.65 -3.92 1.69 1.73 11.26 15.14 18.53 20.05 21.52 22.95 23.45 27.51 30.23 37.95 39.52 41.28 41.46 53.43 56.93 60.30 62.42 63.77 64.53 66.82 71.13 72.31 74.32 74.37 74.63 76.77 81.26 81.56 87.05 88.02 108.57 114.13 128.58 134.90 141.71 148.80 187.45 229.62 257.66	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi **5**

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu

Một mảng số nguyên có độ dài n được cho là đối xứng nếu phần tử thứ k trong mảng $(0 \leq k \leq n-1)$ có giá trị bằng phần tử thứ $n-1-k$.

Viết chương trình nhập vào mảng số nguyên có n phần tử, kiểm tra xem mảng đó có đối xứng hay không.

Nếu có, in ra màn hình "Symmetric array.", ngược lại, in ra màn hình "Asymmetric array."

Input

- Hai dòng văn bản, dòng 1 chứa số $n > 0$ là kích thước của mảng số nguyên.
- Dòng 2 chứa n số nguyên phân tách bởi dấu cách là n phần tử của mang.

Output

- Nếu mảng đối xứng in ra "Symmetric array.", ngược lại, in ra màn hình "Asymmetric array."

Gợi ý

1. Đọc dữ liệu vào biến n.
2. Khai báo mảng số nguyên có n phần tử, dòng vòng for để đọc dữ liệu vào mảng.
3. Cách 1: for 1 chiều => dùng công thức để so sánh số đối diện.
4. Cách 2: Dùng 2 biến chạy đầu cuối

Lưu ý

1. Với cách 1 bạn chỉ cần duyệt 1 nửa mảng.
2. Bạn có thể tham khảo cách 1 như code dưới đây, và hãy thử lại bằng cách 2.

```
for (int k = 0; k < n/2; k++) {
    if ( array[k] == array[n-1-k] ){
        ...
    } else {
        ...
    }
}
```

For example:

Input	Result
4 2 3 3 2	Symmetric array.
5 1 2 3 1 3	Asymmetric array.

Answer: (penalty regime: 0 %)

```
#include <iostream>
using namespace std;

signed main() {
  int n;
  cin >> n;
  int a[n];
  for (int &x : a) cin >> x;

  for (int i = 0; i < n/2; ++i) {
    if (a[i] != a[n-i-1]) {
```

```
12          cout << "Asymmetric array.";
13          return 0;
14        }
15      }
16      cout << "Symmetric array.";
17  }
```

	Input	Expected	Got	
✓	4 2 3 3 2	Symmetric array.	Symmetric array.	✓
✓	5 1 2 3 1 3	Asymmetric array.	Asymmetric array.	✓
✓	20 25 65 76 33 68 39 86 83 22 28 38 20 4 4 62 69 99 82 35 10	Asymmetric array.	Asymmetric array.	✓
✓	21 44 80 29 67 40 65 40 28 39 64 15 64 39 28 40 65 40 67 29 80 44	Symmetric array.	Symmetric array.	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 6

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu

Viết một chương trình đếm số từ trong một câu.

Input

- Một câu được nằm trên một dòng, mỗi từ cách nhau một dấu cách.

Output

- Số từ trong câu đó.

Gợi ý

1. Để lưu trữ 1 xâu kí tự có thể dùng 2 kiểu xâu string và char[]
2. Chỉ số các phần tử trong xâu bắt đầu từ 0.
3. Để truy cập tới phần tử có chỉ số i trong xâu s, dùng s[i].
4. Để đọc xâu có dấu cách hãy dùng lệnh getline, ví dụ: getline(cin, s)
5. Kí tự kết thúc xâu là '\0'.

Lưu ý

1. Để đếm các từ không nên đếm số lượng dấu cách trong xâu, mà đếm có bao nhiêu kí tự là xuất phát của 1 từ.
2. Dấu hiệu nhận biết kí tự thứ i là xuất phát của một từ: s[i] != ' ' && s[i-1] == ' '
3. Dấu hiệu nhận biết bên trên có 1 trường hợp ngoại lệ với từ đầu tiên, bạn cần xử lý trường hợp đấy. Có 2 cách để xử lý trường hợp đặc biệt này.
4. Cách 1: Đối với kí tự đầu tiên, chỉ cần kiểm tra khác dấu cách là được, không cần kiểm tra kí tự đằng trước nó.
5. Cách 2: Có thể thêm 1 dấu cách vào đầu xâu kí tự. Như vậy tất cả các trường hợp đều đưa về 1 dạng. Nhược điểm của cách này là phải tạo thêm bộ nhớ mới. Tham khảo code theo cách 2 dưới đây.

```
// s là kiểu string
s = " " + s; // Thêm dấu cách vào đầu xâu kí tự.
for (int i = 1; i < s.length(); i++) { // s.length() trả về độ dài của xâu kí tự.
    if (s[i] != ' ' && s[i-1] == ' ') {
        num_word++;
    }
}
```

For example:

Input	Result
Lorem ipsum dolor sit amet, consectetur adipiscing elit.	8

Answer: (penalty regime: 0 %)

```
#include <iostream>
#include <algorithm>
using namespace std;

signed main() {
  string s;
  getline(cin, s);
  int cnt = count(begin(s), end(s), ' ') + 1;
  cout << cnt;
}
```

	Input	Expected	Got	
✓	Lorem ipsum dolor sit amet, consectetur adipiscing elit.	8	8	✓
✓	Ut sem justo, sagittis non augue et, pretium cursus est.	10	10	✓
✓	Lorem.	1	1	✓
✓	Etiam egestas ligula feugiat arcu rhoncus ullamcorper.	7	7	✓
✓	Maecenas finibus, lacus et faucibus fermentum, sapien leo pulvinar arcu, fermentum tincidunt neque nisi et risus.	16	16	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 7

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu:

Viết chương trình đếm số lượng chữ cái, chữ số, và kí tự đặc biệt trong một xâu ký tự.

Input:

- Một xâu kí tự nằm trên một dòng (xâu ký tự có thể chứa dấu cách).

Output:

- In trên một dòng số lượng chữ cái, chữ số và kí tự đặc biệt trong xâu, cách nhau bởi dấu cách.

Gợi ý:

1. Dùng `getline` để đọc vào xâu `s` có chứa dấu cách.
2. Tạo 3 biến đếm số lượng tương ứng.
3. Dùng vòng lặp để duyệt từng ký tự trong xâu:
4. Với mỗi ký tự `s[i]`, dùng lệnh rẽ nhánh `if {} else {}` để xét các trường hợp tương ứng, tham khảo:

```
if (s[i] is alphabetical) { // this line is pseudo code
    countAlpha++;
} else if (s[i] is number) { // this line is pseudo code
    countNumber++;
} else {
    countOther++;
}
```

Lưu ý:

1. Để kiểm tra là chữ cái dùng hàm isalpha hoặc dùng trực tiếp điều kiện `'a' <= c && c <= 'z' || 'A' <= c && c <= 'Z'`(*).
2. Để kiểm tra là chữ số dùng hàm isdigit hoặc dùng trực tiếp điều kiện `'0' <= c && c <= '9'`.
3. CÂU HỎI nhắc lại: Trong điều kiện (*) ở trên không có ngoặc, như vậy đã chính xác chưa? Vì sao?
4. Nếu không trả lời được câu hỏi trên cần ôn lại thứ tự thực hiện phép toán và cách thức thực hiện các toán tử `||` và `&&`.

For example:

Input	Result
Code the future!	13 0 3

Answer: (penalty regime: 0 %)

```
#include <cctype>
#include <iostream>
using namespace std;

signed main() {
  string s;
  getline(cin, s);
  int alpha = 0, digit = 0, other = 0;
  for (char c : s) {
    if (isalpha(c)) {
      ++alpha;
    } else if (isdigit(c)) {
      ++digit;
    } else {
      ++other;
    }
  }
  cout << alpha << ' ' << digit << ' ' << other;
}
```

	Input	Expected	Got	
✓	Code the future!	13 0 3	13 0 3	✓
✓	2017 (MMXVII) is the current year, and is a common year starting on Sunday of the Gregorian calendar, the 2017th year of the Common Era (CE) and Anno Domini (AD) designations, the 17th year of the 3rd millennium, the 17th year of the 21st century, and the 8th year of the 2010s decade.	201 20 64	201 20 64	✓
✓	What we're left with is a solid but relatively conventional horror movie, above average but overlong-especially given the decision to limit its scope to half of King's novel.	142 0 32	142 0 32	✓
✓	Call it a symphony of orchestral meta-horror, an elaborate waking nightmare in which you, as the dreamer, are constantly reminded of what the film is trying to do, and yet are powerless to stop it.	157 0 40	157 0 40	✓
✓	Batman is a fictional superhero appearing in American comic books published by DC Comics. The character was created by artist Bob Kane and writer Bill Finger, and first appeared in Detective Comics #27.	164 2 36	164 2 36	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 8

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu:

Khi làm bài tập đặt câu với các vị thần Hy Lạp, Hercules đã vô tình viết sai tên của cha mình là thần Zeus thành "Zues". Hãy viết chương trình giúp Hercules sửa lại những chỗ viết sai này.

Input:

- Một xâu kí tự gồm các từ cách nhau bởi dấu cách được viết trên một dòng.

Output:

- In xâu kí tự viết đúng tên thần Zeus.

Gợi ý:

1. Đọc xâu kí tự sử dụng hàm `getline` và lưu vào biến `string text;`
2. Sử dụng vòng lặp với $i = 0$ đến `text.size()-1`, in ra từng kí tự `text[i]` nếu `i > text.size()-4` hoặc `text.substr(i,4) != "Zues"`, ngược lại, in ra "Zeus" và nhảy cách đến vị trí $i + 4$.

Lưu ý:

1. Kiểm soát giá trị của biến chạy i để không bị truy cập ra ngoài mảng.

For example:

Input	Result
The wrath of Zues.	The wrath of Zeus.

Answer: (penalty regime: 0 %)

```
#include <iostream>
using namespace std;

signed main() {
  string s;
  getline(cin, s);
  int n = s.length();
  for (int i = 0; i < n; ) {
    if (s.substr(i, 4) == "Zues") {
      cout << "Zeus";
      i += 4;
    } else {
      cout << s[i++];
    }
  }
}
```

	Input	Expected	Got	
✓	The wrath of Zues.	The wrath of Zeus.	The wrath of Zeus.	✓
✓	the Z of Zues and zoology.	the Z of Zeus and zoology.	the Z of Zeus and zoology.	✓
✓	On Tuesday, Zues conspired with Ariana.	On Tuesday, Zeus conspired with Ariana.	On Tuesday, Zeus conspired with Ariana.	✓
✓	Zues: Is this a trick?.	Zeus: Is this a trick?.	Zeus: Is this a trick?.	✓
✓	Zeus thought about this.	Zeus thought about this.	Zeus thought about this.	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 9

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu:

Trong soạn thảo văn bản, giữa hai từ chỉ được có một dấu cách. Tuy nhiên trong lúc mệt mỏi, Nhật đã gõ thừa một vài dấu cách giữa các từ trong các câu của văn bản.
Hãy viết một một chương trình giúp Nhật sửa lại các sai sót này (Trong trường hợp có nhiều dấu cách ở đầu dòng thì đó là chủ ý của Nhật, đừng sửa lại).

Input:

- Một chuỗi kí tự là câu bị gõ sai trên một dòng.

Output:

- In ra câu đã sửa đúng quy tắc soạn thảo.

Gợi ý:

1. Đọc xâu kí tự sử dụng hàm `getline` và lưu vào biến `string text;`
2. Đếm số dấu cách ở đầu xâu kí tự sử dụng vòng lặp, điều kiện dừng là đến cuối xâu hoặc khi gặp kí tự khác dấu cách, tạo biến `int countStartSpaces;` để lưu giá trị này
3. In ra màn hình `countStartSpaces` dấu cách.
4. Sử dụng vòng lặp với `i=countStartSpaces` đến `text.size()-1`, in ra từng kí tự `text[i]` nếu `text[i] != " "`, ngược lại, in ra " " và nhảy cách đến vị trí tiếp theo không phải kí tự " " (dùng vòng lặp ở trong).

For example:

Input	Result
`You only  live once, but  if  you  do it right, once is enough.`	`You only live once, but if you do it right, once is enough.`

Answer: (penalty regime: 0 %)

```
#include <iostream>
#include <sstream>
using namespace std;

signed main() {
  string s;
  getline(cin, s);
  for (int i = 0; s[i] == ' '; ++i) {
    cout << ' ';
  }
  stringstream ss(s);
  string t;
  while (ss >> t) {
    cout << t << ' ';
  }
}
```

	Input	Expected	Got	
✓	You only live once, but if you do it right, once is enough.	You only live once, but if you do it right, once is enough.	You only live once, but if you do it right, once is enough.	✓
✓	A room without books is like a body without a soul.	A room without books is like a body without a soul.	A room without books is like a body without a soul.	✓
✓	Be the change that you wish to see in the world.	Be the change that you wish to see in the world.	Be the change that you wish to see in the world.	✓
✓	No one can make you feel inferior without your consent.	No one can make you feel inferior without your consent.	No one can make you feel inferior without your consent.	✓
✓	Without music, life would be a mistake.	Without music, life would be a mistake.	Without music, life would be a mistake.	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Câu hỏi 10

Đúng

Đạt điểm 10,00 trên 10,00

Yêu cầu:

Garena là một công ty dịch vụ kỹ thuật số tham gia chơi game, thể thao điện tử khá nổi tiếng tại Việt Nam.

Để tham gia vào chơi game, đầu tiên, bạn phải có một tài khoản hợp lệ.

Tên đăng nhập Garena hợp lệ phải:

- Sử dụng ký tự trong bảng chữ cái và số, không phân biệt viết hoa hay viết thường.
- Không chứa ký tự đặc biệt.
- Độ dài từ 6 đến 15 ký tự.
- Không bắt đầu bằng số.

Viết chương trình kiểm tra xem một tên đăng nhập có phải tên đăng nhập Garena hợp lệ hay không.

Input:

- Một dòng duy nhất chứa một tên đăng nhập.

Output:

- Nếu tên đăng nhập hợp lệ, in ra màn hình "Valid username.", ngược lại, in ra màn hình "Invalid username."

Gợi ý:

1. Khởi tạo một xâu ký tự và nhập vào tên đăng nhập từ bàn phím.
2. Kiểm tra tất cả ký tự có nằm trong bảng chữ cái hay số không. Nếu có, tiếp tục bước 3. Nếu không, in ra "Invalid username." và dừng lại.
3. Kiểm tra có ký tự nào là ký tự đặc biệt không. Nếu không, tiếp tục bước 4. Nếu có, in ra "Invalid username." và dừng lại.
4. Kiểm tra độ dài có hợp lệ không. Nếu có tiếp tục bước 5. Nếu có, in ra "Invalid username." và dừng lại.
5. Kiểm tra xâu có bắt đầu bằng số không. Nếu không, in ra "Valid username.". Nếu có in ra "Invalid username.".

Lưu ý:

1. Cách kiểm tra chữ cái và chữ số đã học trong bài trước. Ngoài ra, có thể dùng hàm isalnum là hàm kết hợp cả chữ cái và chữ số.
2. Nếu đã có bước 3 thì bước 4 có thực sự cần thiết? Khi làm các bài tập với điều kiện phức tạp cần chia các trường hợp cẩn thận để không kiểm tra thừa.
3. Thứ tự thực hiện các bước như trên có hợp lý không? Điều gì xảy ra nếu chúng ta kiểm tra một xâu có 7749 ký tự hợp lệ rồi đến bước 4 mới phát hiện ra là xâu quá dài? Như vậy, thứ tự thực hiện, kiểm tra các điều kiện cũng quan trọng.
4. Một chương trình mà viết tới 4 khối lệnh để kiểm tra nhiều điều kiện khác nhau như thế này có dễ nhìn không? Hẹn các bạn trong bài học sau về Hàm!

For example:

Input	Result
jby3pL8wRb	Valid username.

Answer: (penalty regime: 0 %)

```
#include <cctype>
#include <iostream>
using namespace std;

bool check(string &s) {
  int n = s.length();
  if (n < 6 || n > 15) {
    return false;
  }
  if (isdigit(s[0])) {
    return false;
  }
```

```
  for (char c : s) {
    if (!isalnum(c)) {
      return false;
    }
  }
  return true;
}

signed main() {
  string s;
  getline(cin, s);
  if (check(s)) {
    cout << "Valid username.";
  } else {
    cout << "Invalid username.";
  }
}
```

	Input	Expected	Got	
✓	jby3pL8wRb	Valid username.	Valid username.	✓
✓	#0E_d9RuS6BD	Invalid username.	Invalid username.	✓
✓	ObeFeKaDaL18boOp0gtzUi5gJ	Invalid username.	Invalid username.	✓
✓	0sOz0	Invalid username.	Invalid username.	✓
✓	qxom6	Invalid username.	Invalid username.	✓
✓	V10MMvD_hE	Invalid username.	Invalid username.	✓
✓	aiKeJ6xVp9uSrou	Valid username.	Valid username.	✓
✓	2yih6vSLGM7M	Invalid username.	Invalid username.	✓

Passed all tests! ✓

Đúng

Marks for this submission: 10,00/10,00.

Trở lại Khoá học