

Question 1

Not answered

Mark 0.00 out of 10.00

[Template Statistics]

Viết [mẫu hàm](#) `void printStatistics(T a[], int n)` để thực hiện một [thống kê](#) cơ bản về số lớn nhất, số nhỏ nhất của dãy số a . Tham số đầu vào của hàm là mảng `a` chứa các số có kiểu `T` và `n` là số lượng phần tử của mảng đó. Hàm `printStatistics()` có nhiệm vụ in ra số lớn nhất và nhỏ nhất của mảng `a`.

Đầu vào

Gồm hai dòng được nhập vào từ bàn phím:

- Dòng thứ nhất là số `n` là số lượng phần tử của dãy a .
- Dòng thứ hai chứa `n` phần tử của dãy a , các phần tử cách nhau bởi một dấu cách.

Đầu ra

In ra màn hình số lớn nhất và nhỏ nhất của dãy a trên một dòng. Hai số cách nhau bởi một khoảng trống.

Lưu ý: Bạn chỉ cần hoàn thành hàm `printStatistics` theo yêu cầu. Phần nhập dữ liệu đầu vào đã được lập trình sẵn.

For example:

Input	Result
5	5 1
1 2 3 4 5	3.3 1.2
4	
1.2 3.3 2.2 2.8	

Answer: (penalty regime: 0 %)

1

Question 2

Not answered

Mark 0.00 out of 10.00

[Template Max Of Two]

Viết [mẫu hàm](#) `T getMax(T a, T b)` nhận tham số là hai biến số `a`, `b` thuộc kiểu `T` và trả về giá trị lớn hơn trong hai giá trị `a`, `b`.

Đầu vào

Một dòng duy nhất từ bàn phím chứa hai số `a` và `b` cách nhau bởi một khoảng trống.

Đầu ra

In ra màn hình một dòng duy nhất chứa [số lớn hơn](#) trong hai số `a` và `b`.

Lưu ý: Bạn chỉ cần hoàn thành hàm `getMax` trả về giá trị lớn hơn của hai biến. Phần nhập dữ liệu và in kết quả ra màn hình đã được lập trình sẵn.

For example:

Input	Result
3 4	4
6.2 5.2	6.2

Answer: (penalty regime: 0 %)

1

Question 3

Not answered

Mark 0.00 out of 10.00

[Get Sum Of Two]

Viết [mẫu hàm](#) `T getSum(T a, T b)` nhận tham số đầu vào là hai biến `a, b` thuộc kiểu số `T` và trả về tổng của `a` và `b`.

Đầu vào

Một dòng duy nhất từ bàn phím chứa hai số `a` và `b` cách nhau bởi một khoảng trống.

Đầu ra

Một dòng duy nhất chứa tổng của hai số `a` và `b`.

Lưu ý: Bạn chỉ cần hoàn thành hàm `getSum` để tính tổng của hai số. Phần nhập dữ liệu đầu vào và in kết quả ra màn hình đã được lập trình sẵn.

For example:

Test	Input	Result
float a, b;	3 4	7
while (cin >> a >> b) { cout << getSum(a, b) << endl; }	6.2 5.2	11.4

Answer: (penalty regime: 0 %)

1

► SHOW/HIDE QUESTION AUTHOR'S SOLUTION (CPP)

Question 4

Not answered

Mark 0.00 out of 10.00

[Get Sum Of An Array]

Viết [mẫu hàm](#) `T getSum(T a[], int n)` để tính tổng của dãy số a . Tham số đầu vào của hàm là mảng `a` chứa các phần tử kiểu số `T` và biến `n` là số phần tử của mảng `a`. Nhiệm vụ của hàm `getSum` là trả về tổng các phần tử của mảng `a`.

Đầu vào

Gồm hai dòng nhập vào từ bàn phím:

- Dòng thứ nhất chứa số n là số lượng phần tử của dãy.
- Dòng thứ hai chứa n phần tử của dãy cách nhau bởi một khoảng trống.

Đầu ra

In ra màn hình một dòng duy nhất chứa tổng các phần tử của dãy.

Lưu ý: Bạn chỉ cần hoàn thành hàm `getSum` trả về tổng giá trị của các phần tử trong dãy. Phần nhập dữ liệu và in kết quả ra màn hình đã được lập trình sẵn.

Answer: (penalty regime: 0 %)

1	
---	--

► SHOW/HIDE QUESTION AUTHOR'S SOLUTION (CPP)

Question 5

Not answered

Mark 0.00 out of 10.00

[Output Function]

Cho nguyên [mẫu hàm](#) sau để hiển thị số nguyên ra màn hình: `void output(const int &);`

Các bạn hãy viết định nghĩa hàm trên (chỉ cần định nghĩa hàm này) mà không cần viết hàm `main()`, thư viện cũng đã có sẵn cho bạn.

For example:

Input	Result
1	1

Answer: (penalty regime: 0 %)

1	
---	--

Question 6

Not answered

Mark 0.00 out of 10.00

[Check Prime Number]

Cho nguyên [mẫu hàm](#) sau để kiểm tra một số có phải là [số nguyên tố](#) không: `bool isPrime(int);`

Hàm `isPrime` 🐞 sẽ trả về giá trị **true** nếu n là [số nguyên tố](#) và giá trị **false** trong trường hợp ngược lại. Các bạn hãy viết định nghĩa hàm trên (chỉ cần định nghĩa hàm này) mà không cần viết `main()`, thư viện cũng có sẵn cho bạn rồi.

Answer: (penalty regime: 0 %)

1	
---	--

Question 7

Not answered

Mark 0.00 out of 10.00

[HappyNumber]

Số may mắn là số được định nghĩa theo quá trình sau: bắt đầu với số nguyên dương x và tính tổng bình phương y các chữ số của x , sau đó tiếp tục tính tổng bình phương các chữ số của y . Quá trình này lặp đi lặp lại cho đến khi thu được kết quả là 1 thì dừng (tổng bình phương các chữ số của số 1 chính là 1) hoặc quá trình sẽ kéo dài vô tận. Số mà quá trình tính này kết thúc bằng 1 gọi là số may mắn. Số có quá trình tính kéo dài vô tận là **số không may mắn** hay còn gọi là **số đen đủi**.

Ví dụ, 7 là số may mắn:

$$0^2 + 7^2 = 49$$

$$4^2 + 9^2 = 97$$

$$9^2 + 7^2 = 130$$

$$1^2 + 3^2 + 0^2 = 10$$

$$1^2 + 0^2 = 1$$

Viết hàm `bool isHappyNumber(int n)` trả về `true` nếu số n là số may mắn ngược lại trả về `false`.

For example:

Test	Input	Result
<pre>int n; cin >> n; cout << isHappyNumber(n);</pre>	19	1

Answer: (penalty regime: 0 %)

1	
---	--

► SHOW/HIDE QUESTION AUTHOR'S SOLUTION (CPP)

Question 8

Not answered

Mark 0.00 out of 10.00

[LongestHarmoniousSubsequence]

Dãy hài hòa là một dãy thỏa mãn điều kiện giá trị lớn nhất và giá trị nhỏ nhất của dãy khác nhau đúng bằng 1.

Viết hàm `int findLHS(int arr[], int n)` trả về độ dài của dãy hài hòa con dài nhất có thể có trong mảng giá trị nguyên `arr` có `n` phần tử

Ví dụ:

Input: 1 3 2 2 5 2 3 7

Output: 5 - Dãy hài hòa con dài nhất là [3,2,2,2,3]

For example:

Test	Input	Result
findLHS(arr, n)	4 1 2 3 4	2

Answer: (penalty regime: 0 %)

1

Question 9

Not answered

Mark 0.00 out of 10.00

[Template Swap]

Viết [mẫu hàm](#) `void swapNumber(T & x, T & y)` thực hiện việc hoán đổi giá trị của hai biến x, y là hai số có cùng kiểu T .

For example:

Input	Result
3 4	4 3
6.2 5.2	5.2 6.2

Answer: (penalty regime: 0 %)

1	
---	--

[Back to Course](#)