Trạng thái	Đã xong
Bắt đầu vào lúc	Thứ Bảy, 10 tháng 5 2025, 6:58 AM
Kết thúc lúc	Thứ Bảy, 10 tháng 5 2025, 7:14 AM
Thời gian thực hiện	15 phút 31 giây

Câu hỏi **1**

Đúng

Đạt điểm 1,00

Hiện thực hàm sau:
int findMax(int *ptr, int n);
Tìm và trả về phần tử lớn nhất trong mảng 1 chiều được cho bởi con trỏ.
Trong đó:
ptr là con trỏ tới phần tử đầu tiên trong mảng.
n là kích thước của mảng.

Implement the following function:
int findMax(int *ptr, int n);
Find and return the maximum element of a 1-dimension array given by a pointer.
Where:
ptr is a pointer to the first element in the array.
n is the size of the array.

For example:

Test	Result
`int arr[] = {1, 2, 3, 4, 5};` `cout << findMax(arr, sizeof(arr) / sizeof(arr[0]));`	5

Answer: (penalty regime: 0, 0, 0, 0, 0, 100 %)

Reset answer

```
1  int findMax(int *ptr, int n)
2  {
3      int *temp = ptr;
4      int max = *temp;
5
6      for(int i = 0; i < n; ++i){
7          if(*temp > max)
8              max = *temp;
9          temp++;
10     }
11     return max;
12 }
```

	Test	Expected	Got	
✓	`int arr[] = {1, 2, 3, 4, 5};` `cout << findMax(arr, sizeof(arr) / sizeof(arr[0]));`	5	5	✓

Passed all tests! ✓

Câu hỏi **2**

Đúng

Đạt điểm 1,00

Hiện thực hàm sau:
void reverse(int *ptr, int n);
Đảo ngược mảng 1 chiều được cho bởi con trỏ.
Trong đó:
ptr là con trỏ tới phần tử đầu tiên trong mảng.
n là kích thước của mảng 1 chiều.

Lưu ý: Bạn cần phải dùng dereference operator (*) để lấy giá trị của các phần tử trong mảng. Không được dùng subscript operator ([]).

Implement the following function:
void findMax(int *ptr, int n);
Reverse the 1-dimension array given by a pointer.
Where:
ptr is a pointer to the first element in the array.
n is the size of the array.

Note: You need to use the dereference operator (*) to get the values of the elements in the array. The subscript operator ([]) cannot be used.

For example:

Test	Result

`int arr[] = {1, 2, 3, 4, 5};`
`reverse(arr, sizeof(arr) / sizeof(arr[0]));`
`cout << arr[0];`
`for (int i = 1; i < sizeof(arr) /`
`sizeof(arr[0]); i++)`
`{`
`  cout << ", " << arr[i];`
`}`
	5, 4, 3, 2, 1

Answer: (penalty regime: 0, 0, 0, 0, 0, 0, 0, 0, 50, 100 %)

Reset answer

```
1 void reverse(int *ptr, int n) {
2     int *temp = ptr;
3
4     for(int i = 0; i < n / 2; ++i){
5         swap(*(temp + i), *(temp + n - i - 1));
6
7     }
8 }
```

	Test	Expected	Got	
✓				✓

	int arr[] = {1, 2, 3, 4, 5};
	reverse(arr, sizeof(arr) / sizeof(arr[0]));
	cout << arr[0];
	for (int i = 1; i < sizeof(arr) / sizeof(arr[0]); i++)
	{
	    cout << ", " << arr[i];
	}
		5, 4, 3, 2, 1	5, 4, 3, 2, 1	

Passed all tests! ✓

Câu hỏi **3**

Đúng

Đạt điểm 1,00

Hiện thực hàm sau:
bool isSymmetry(int *head, int *tail);
Kiểm tra mảng 1 chiều có phải là một mảng đối xứng hay không.
Trong đó:
head, tail lần lượt là con trỏ tới phần tử đầu tiên và cuối cùng trong mảng.

Implement the following function:
bool isSymmetry(int *head, int *tail);
Checks if the 1-dimensional array is a symmetric array.
Where:
head, tail respectively are pointers to the first element and last element of the array.

Lưu ý: Sinh viên chỉ có 5 lần nộp không tính penalty, ở lần nộp thứ 6 trở đi bài làm sẽ được tính là 0 điểm.

For example:

Test	Result
`int arr[] = {1, 2, 1};` `cout << isSymmetry(arr, arr + (sizeof(arr) / sizeof(arr[0])) - 1);`	1

Answer: (penalty regime: 0, 0, 0, 0, 0, 100 %)

Reset answer

```
1  bool isSymmetry(int *head, int *tail)
2  {
3      while(*head != *tail){
4          return false;
5          head--;
6          tail++;
7      }
8      return true;
9  }
```

	Test	Expected	Got	
✓	`int arr[] = {1, 2, 1};` `cout << isSymmetry(arr, arr + (sizeof(arr) / sizeof(arr[0])) - 1);`	1	1	✓

Passed all tests! ✓