Name and Title: Array

Important Instructions:

1. Use Save or Save All buttons to save additions/changes made in code on a periodic basis
2. DO NOT DELETE provided code in the src folder
3. DO NOT EDIT OR DELETE provided test cases' code in the test folder.
4. Validate the created code with a provided test case(s) rigorously before submission
5. Once the test is submitted or total test time is elapsed, no changes can be done to the code
6. Be aware of syntax errors and run-time errors before submitting the code. In case of any such errors, the test cases will not be executed.
    - **Syntax Errors:**
        - To check any syntax errors, click on the Build button. This will highlight any syntax errors in the Console Output window. In case if there is no Syntax Error(s) in the code written, the "Build Succeeded" message will be visible in the Console Output window.
    - **Run-time errors:**
        - To check run-time errors in the code,(it is mandatory that there are no syntax errors in your code, and the "Build Succeeded" message appears) you can click on the Run As Console and Run Unit Tests buttons
        - When you click Run As Console, the program execution starts from the main function as the entry point, and when you click Run Unit Tests, the test cases created will be executed against the code you have written.
        - Run-time errors can appear anytime when you click on the Run As Console or Run Unit Tests buttons. In this case, the test case(s) are not able to validate your code and your scores become void/zero.
        - Following are some sample messages that can appear as run-time errors on Console Output Window:
            - Runtime Error. Exit Code:139
            - Result: 139
            - segmentation fault or RunTime Error. Exit Code:136

*Here 139 and 136 are a few of the error codes displayed, for the run time error that has occurred. The code number can vary depending on the type of error detected during run-time execution.

Test case(s) Execution:

- After clicking on the “Run Unit Tests” button, in case no Syntax and run time errors are present, the test case(s) are successfully executed. Following are sample messages which will appear on the Output Console window:
    - [echo] Running Test cases from the test directory
    - [exec] Running cxxtest tests
- Navigate through the Output Console window to check if some or all test cases have passed.
    - **Failed Test cases:** In case one or more test cases have failed, intimation will be shown on the Output Console window. There will also be details on errors encountered while test execution. A sample message is shown below (where X is a number of failed test cases):
        - ❖ [exec] Failed X
    - Passed Test Cases: In case all test cases have passed intimation will be shown on the Output Console window. A sample message is shown below:
        - ❖ [exec] OK!

Name and Title: Array

Introduction:
An **array** is a group of similar data type which are stored under a single name. The main advantage of array are code optimization, easy to transverse data, easy to sort data, random access.

This problem statement in C Language focuses on the Array's. The analysis and computations that can be done on the data.

Problem Description:

In this problem statement, you are expected to complete the following function.

Function Name	Arguments	Return value	Description
arr_length	const char*	int	The function returns length of given string..
rep_chara_check	const char*	Bool	The function returns bool (either true(1) or false(0)). In this case, the function will check repeated character in a given string, if it's found, then return true or else returns false,
check_any_index	const char*, int	Bool	The function returns bool (either true(1) or false(0)). In this case, the function will check any index value is a character or a Number. It has two arguments one is string and other one is an integer value. The integer value helps to indicate the index in the string, if that index value has numbered it returns 1, else it return 0.
check_mid_chara	const char*	Character	The function returns middle character in given string. **Case 1: Array elements are odd** If the array elements are odd, it's easy to identify the middle element in given string. **Case2: Array elements are even.** If the array elements are even then we need compare and the greater one has to print. Ex: Array elements "abcd" there is no middle character so we need compare 'b' element and 'c' which is greater that has to print. In above example c is greater, then output will be 'c'

Test cases:
****PLEASE DO NOT EDIT THE TEST CASES**
The above-mentioned requirements have been considered for creating test cases.
The below table gives details of test cases to be used for validating written code.

Test Case no.	**Test case description**	**Input values**	**Expected Output**	**Explanation**
1	To validate arr_length function	arr_length(“buffer”)	6	As per the input value, the length of the given string calculated.
2	To validate rep_chara_check	rep_chara_check(“aabb”)	1	The function will check repeated characters, if yes it will returns true or 0. **Condition: Either pass lower case or upper case characters.**
3	To validate check_any_index	check_any_index(“anyindex”,4)	0	The function will check, number or character, since the integer value is 4, in 4 th position the string character is ‘i’, hence expected output is false.
4	To validate check_mid_chara	check_mid_chara(“games”),’m’)	m	The function will print middle character in given string.

Also, please note that there are a separate set of test cases (NOT VISIBLE to TEST TAKER) and a separate set of data records that are used for the validation of submitted code and calculating the final score.

ALL THE BEST!!