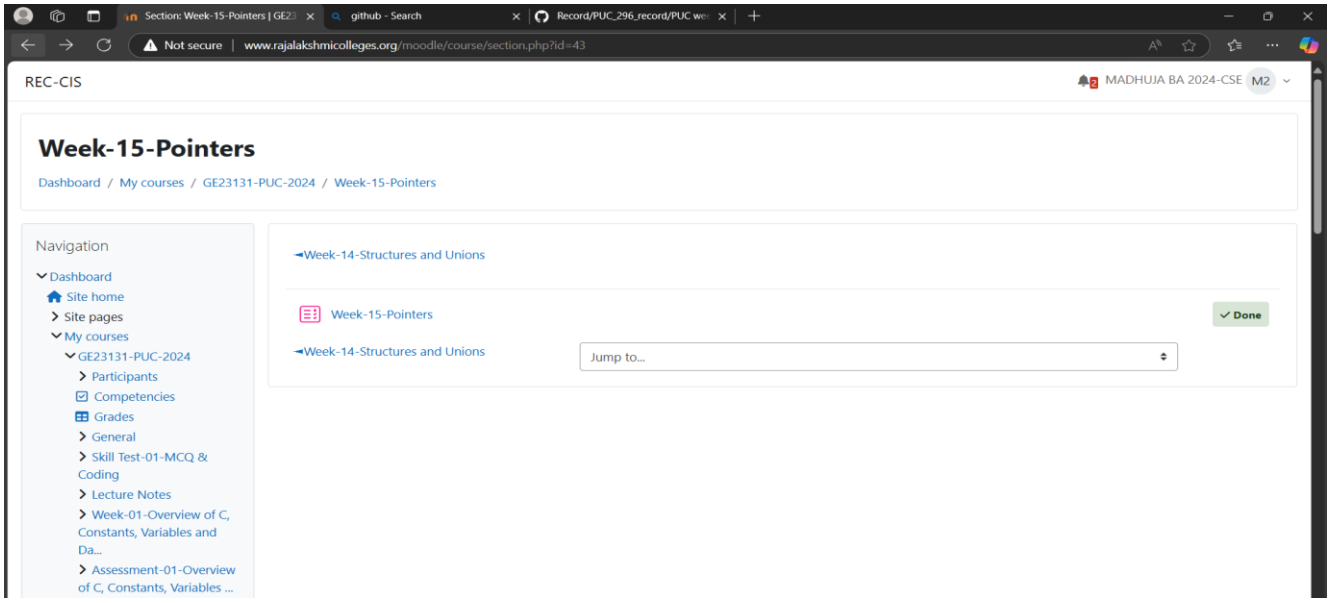
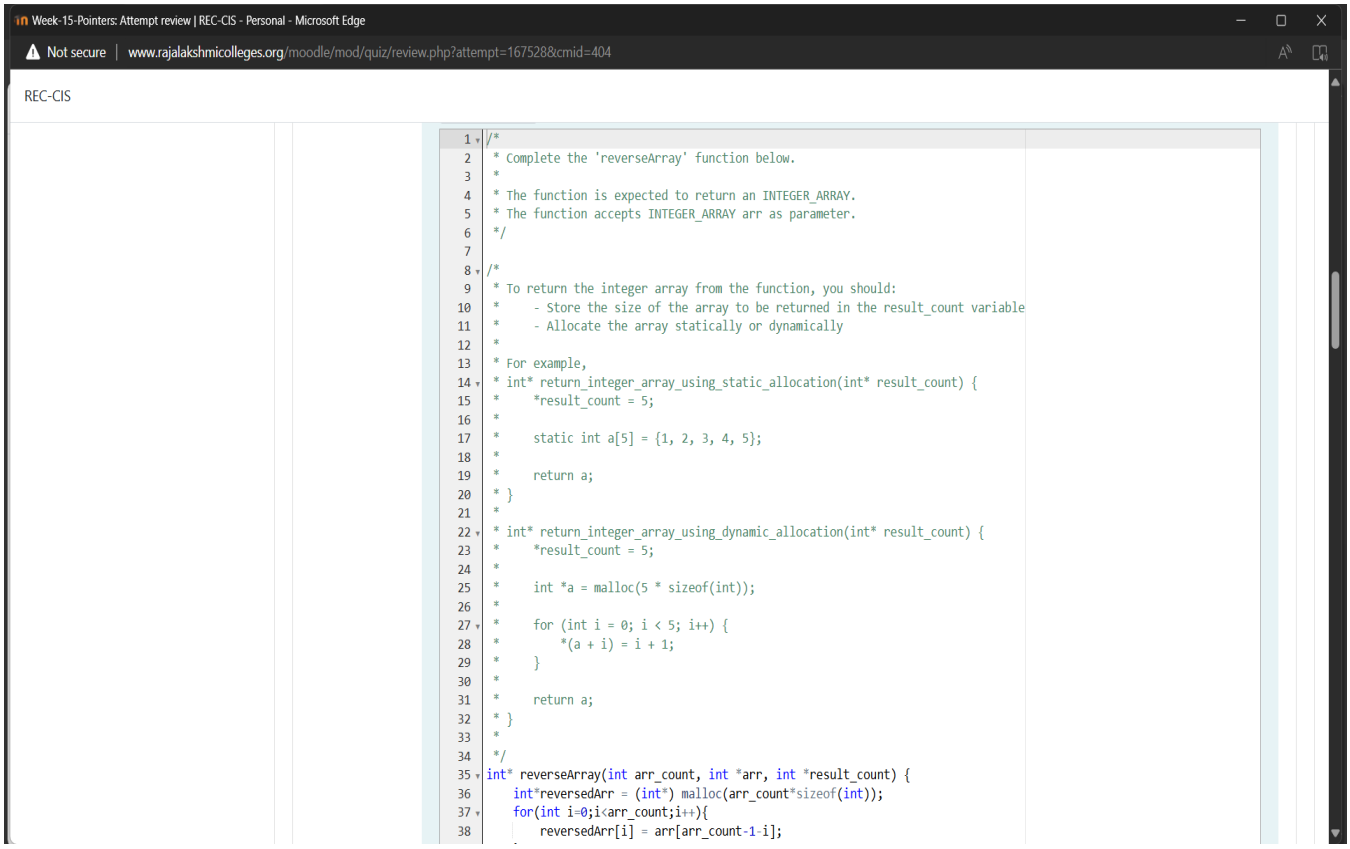
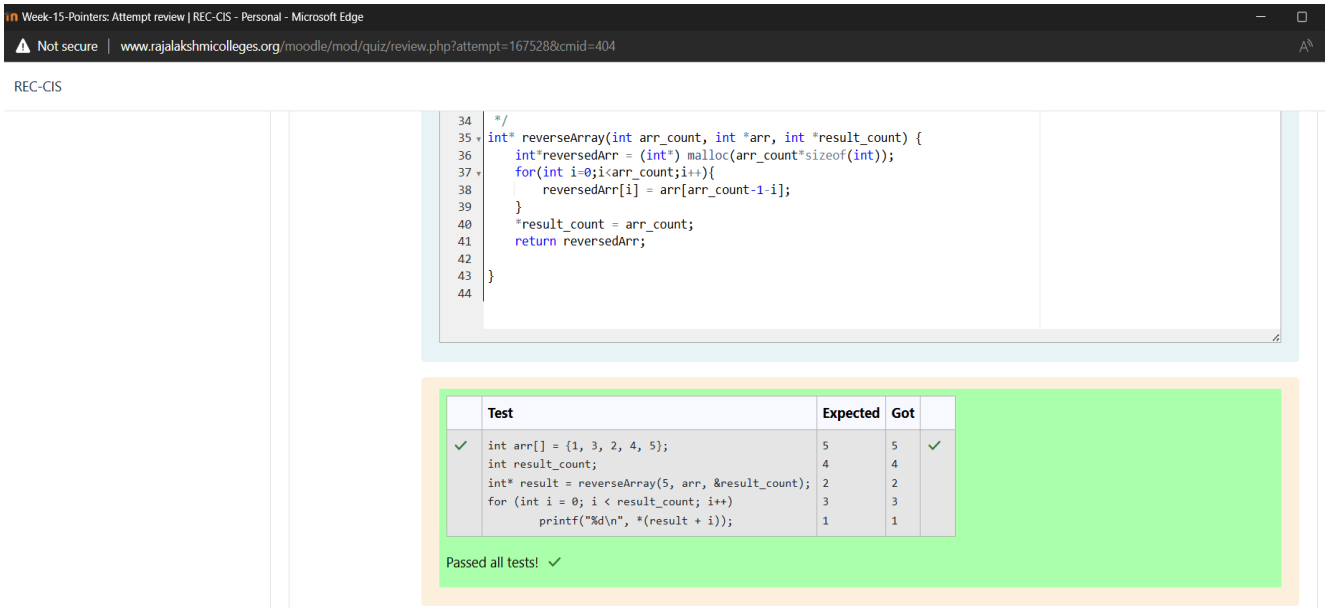
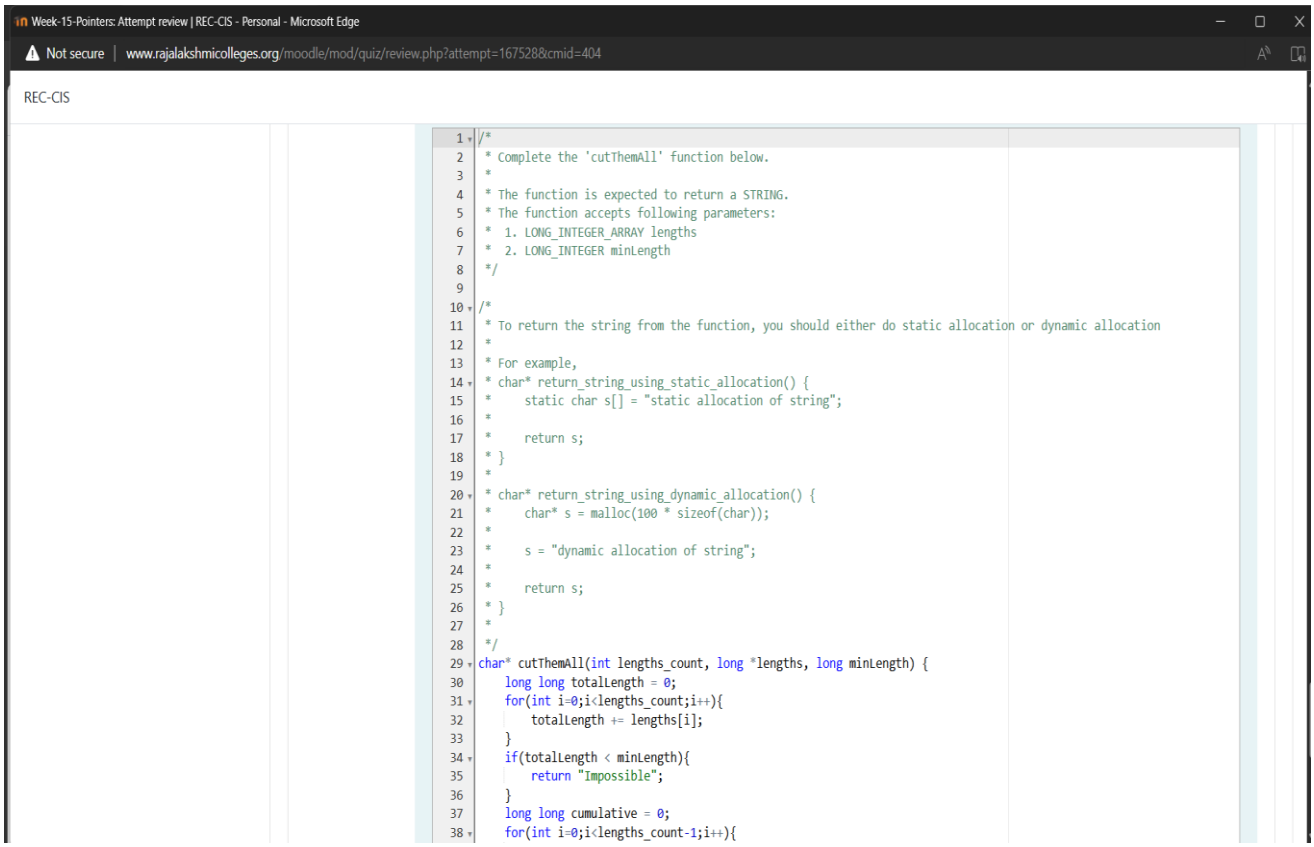

Section: Week-15-Pointers | GE23
github - Search
Record/PUC_296_record/PUC we
Not secure | www.rajalakshmicolleges.org/moodle/course/section.php?id=43
REC-CIS
MADHUJA BA 2024-CSE M2
Week-15-Pointers
Dashboard / My courses / GE23131-PUC-2024 / Week-15-Pointers
Navigation
Dashboard
Site home
Site pages
My courses
GE23131-PUC-2024
Participants
Competencies
Grades
General
Skill Test-01-MCQ & Coding
Lecture Notes
Week-01-Overview of C, Constants, Variables and Da...
Assessment-01-Overview of C, Constants, Variables ...
Week-14-Structures and Unions
Week-15-Pointers
Done
Week-14-Structures and Unions
Jump to...


Week-15-Pointers: Attempt review | REC-CIS - Personal - Microsoft Edge
Not secure | www.rajalakshmicolleges.org/moodle/mod/quiz/review.php?attempt=167528&cmid=404
REC-CIS
/*
* Complete the 'reverseArray' function below.
*
* The function is expected to return an INTEGER_ARRAY.
* The function accepts INTEGER_ARRAY arr as parameter.
*/
/*
* To return the integer array from the function, you should:
* - Store the size of the array to be returned in the result_count variable
* - Allocate the array statically or dynamically
*
* For example,
* int* return_integer_array_using_static_allocation(int* result_count) {
* *result_count = 5;
*
* static int a[5] = {1, 2, 3, 4, 5};
*
* return a;
* }
*
* int* return_integer_array_using_dynamic_allocation(int* result_count) {
* *result_count = 5;
*
* int *a = malloc(5 * sizeof(int));
*
* for (int i = 0; i < 5; i++) {
* *(a + i) = i + 1;
* }
*
* return a;
* }
*
*/
int* reverseArray(int arr_count, int *arr, int *result_count) {
int*reversedArr = (int*) malloc(arr_count*sizeof(int));
for(int i=0;i<arr_count;i++){
reversedArr[i] = arr[arr_count-1-i];


Week-15-Pointers: Attempt review | REC-CIS - Personal - Microsoft Edge
Not secure | www.rajalakshmicolleges.org/moodle/mod/quiz/review.php?attempt=167528&cmid=404
REC-CIS
*/
int* reverseArray(int arr_count, int *arr, int *result_count) {
int*reversedArr = (int*) malloc(arr_count*sizeof(int));
for(int i=0;i<arr_count;i++){
reversedArr[i] = arr[arr_count-1-i];
}
*result_count = arr_count;
return reversedArr;
}
Test
Expected
Got
int arr[] = {1, 3, 2, 4, 5};
int result_count;
int* result = reverseArray(5, arr, &result_count);
for (int i = 0; i < result_count; i++)
printf("%d\n", *(result + i));
5
4
2
3
1
5
4
2
3
1
Passed all tests!


Week-15-Pointers: Attempt review | REC-CIS - Personal - Microsoft Edge
Not secure | www.rajalakshmicolleges.org/moodle/mod/quiz/review.php?attempt=167528&cmid=404
REC-CIS
/*
* Complete the 'cutThemAll' function below.
*
* The function is expected to return a STRING.
* The function accepts following parameters:
* 1. LONG_INTEGER_ARRAY lengths
* 2. LONG_INTEGER minLength
*/
/*
* To return the string from the function, you should either do static allocation or dynamic allocation
*
* For example,
* char* return_string_using_static_allocation() {
* static char s[] = "static allocation of string";
*
* return s;
* }
*
* char* return_string_using_dynamic_allocation() {
* char* s = malloc(100 * sizeof(char));
*
* s = "dynamic allocation of string";
*
* return s;
* }
*
*/
char* cutThemAll(int lengths_count, long *lengths, long minLength) {
long long totalLength = 0;
for(int i=0;i<lengths_count;i++){
totalLength += lengths[i];
}
if(totalLength < minLength){
return "Impossible";
}
long long cumulative = 0;
for(int i=0;i<lengths_count-1;i++){

```
    for(int i=0;i<lengths_count;i++){
        totalLength += lengths[i];
    }
    if(totalLength < minLength){
        return "Impossible";
    }
    long long cumulative = 0;
    for(int i=0;i<lengths_count-1;i++){
        cumulative += lengths[i];
        if(cumulative >= minLength){
            cumulative = 0;
        }
    }
    if(cumulative > 0 && cumulative<minLength){
        return "Possible";
    }
    return "Possible";

}
```

	Test	Expected	Got	
✓	`long lengths[] = {3, 5, 4, 3};` `printf("%s", cutThemAll(4, lengths, 9))`	Possible	Possible	✓
✓	`long lengths[] = {5, 6, 2};` `printf("%s", cutThemAll(3, lengths, 12))`	Impossible	Impossible	✓

Passed all tests! ✓