

Laboratory Record Notebook

Name: GOWTHAM BR

Register No: 230701524

Branch: B.E COMPUTER SCIENCE

AND ENGINEERING

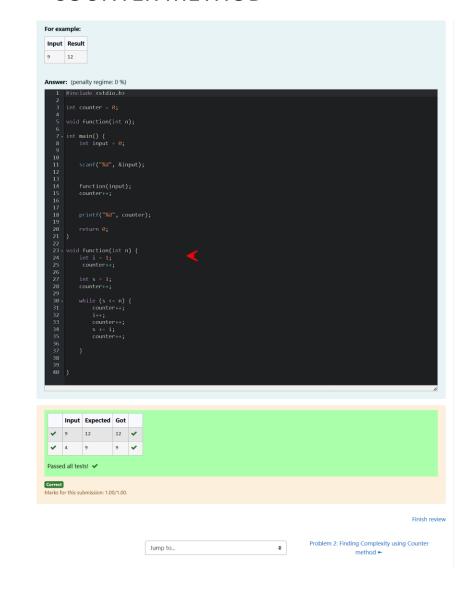
Year: II

Section: C

Semester: III

WEEK 1

FINDING TIME COMPLEXITY OF ALGORITHMS USING COUNTER METHOD





	Input	Expected	Got	
~	12	31	31	~
~	25	54	54	~
~	4	12	12	~
Passed	d all tes	ts! 🗸		
Correct Marks for this submission: 1.00/1.00.				

Finish review

→ Problem 2: Finding Complexity using Counter method

Jump to... 💠

Problem 4: Finding Complexity using Counter Method ►

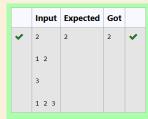


	Input	Expected	Got	
~	4	30	30	~
~	10	212	212	~

Correct

	Input	Expected	Got	
~	12	11	11	~
~	1234	19	19	~

WEEK-2 GREEDY ALGORITHMS



Correct

	Test	Input	Expected	Got	
~	Test Case 1	3 1 3 2	18	18	~
~	Test Case 2	4 7 4 9 6	389	389	~
*	Test Case 3	3 5 10 7	76	76	~

Correct

	Input	Expected	Got	
~	5	40	40	~
	2			
	5			
	3			
	4			
	0			
~	10	191	191	~
	2			
	2			
	2			
	4			
	4			
	3			
	3			
	5			
	5			
	5			
~	2	45	45	~
	45			
	3			

Correct

	Input	Expected	Got	
~	3	28	28	~
	1			
	2			
	3			
	4			
	5			
	6			
~	4	22	22	~
	7			
	5			
	1			
	2			
	1			
	3			
	4			
	1			
~	5	590	590	~
	20			
	10			
	30			
	10			
	40			
	8			
	9			
	4			
	3			
	10			

Correct

WEEK-3 DIVIDE AND CONQURE

```
Answer: (penalty regime: 0 %)
```

```
#include <stdio.h>

int countZeros(int arr[], int low, int high) {
    if (low > high)
        return 0;

    if (arr[low] == 0 && arr[high] == 0)
        return high - low + 1;

if (arr[low] == 1 && arr[high] == 1)
        return 0;

int mid = (low + high) / 2;

return countZeros(arr, low, mid) + countZer

int main() {
    int m;

    scanf("%d", &m);

    int arr[m];

for (int i = 0; i < m; i++) {
        scanf("%d", &arr[i]);
    }

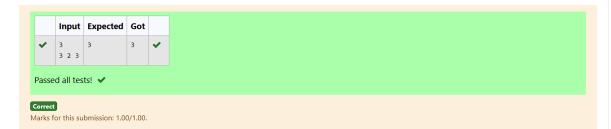
int zeroCount = countZeros(arr, 0, m - 1);
    printf("%d", zeroCount);

return 0;

return 0;

}</pre>
                                                      int zeroCount = countZeros(arr, 0, m - 1);
printf( "%d", zeroCount);
```

	Input	Expected	Got	
~	5 1 1 1 0	2	2	~
~	10 1 1 1 1 1 1 1 1 1 1 1	Θ	Ø	~
~	8 0 0 0 0 0 0 0	8	8	•
~	17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	2	2	*



Finish review

	Input	Expected	Got	
~	4	4	4	~
	2	10	10	
	4			
	8			
	10			
	14			
~	5	No	No	~
	2			
	4			
	6			
	8			
	10			
	100			

Correct

	Input	Expected	Got	
~	5 67 34 12 98 78	12 34 67 78 98	12 34 67 78 98	~
~	10 1 56 78 90 32 56 11 10 90 114	1 10 11 32 56 56 78 90 90 114	1 10 11 32 56 56 78 90 90 114	~
~	12 9 8 7 6 5 4 3 2 1 10 11 90	1 2 3 4 5 6 7 8 9 10 11 90	1 2 3 4 5 6 7 8 9 10 11 90	~

Correct

WEEK-4 DYNAMIC PROGRAMING

```
Answer: (penalty regime: 0 %)
```

	Input	Expected	Got	
~	3	19	19	~
	1 2 4			
	2 3 4			
	8 7 1			
~	3	12	12	~
	1 3 1			
	1 5 1			
	4 2 1			
~	4	28	28	~
	1 1 3 4			
	1 5 7 8			
	2 3 4 6			
	1 6 9 0			

Correct

	Input	Expected	Got	
~	aab azb	2	2	~
~	ABCD ABCD	4	4	~

Correct

	Input	Expected	Got	
~	9 -1 3 4 5 2 2 2 2 3	6	6	~
~	7 1 2 2 4 5 7 6	6	6	~

Correct

WEEK-5 COMPITATIVE PROGRAMMING

```
Answer: (penalty regime: 0 %)
           1 #include <stdio.h>
                            int slow = arr[0];
int fast = arr[arr[0]];
                            while (slow != fast) {
    slow = arr[slow];
    fast = arr[arr[fast]];
                          fast = 0;
while (slow != fast) {
    slow = arr[slow];
    fast = arr[fast];
     16 v while (sl
17 slow
18 fast
19 }
20
21 return sl
22 }
23
24 v int main() {
25 int n;
26
27
28 scanf("%d
29
30 int arr[n
31
32
33 v for (int
34 scanf
35 }
36
37
38 int dupli
39 printf("%
40
41 return 0;
42 }
43
                            int duplicate = findDuplicate(arr, n);
printf("%d\n", duplicate);
```

	Input	Expected	Got	
~	11 10 9 7 6 5 1 2 3 8 4 7	7	7	~
~	5 1 2 3 4 4	4	4	~
~	5 1 1 2 3 4	1	1	~
asse	ed all tests! 🗸			
ect				

```
Answer: (penalty regime: 0 %)
```

	Input	Expected	Got	
~	11 10 9 7 6 5 1 2 3 8 4 7	7	7	~
~	5 1 2 3 4 4	4	4	~
~	5 1 1 2 3 4	1	1	~

Correct

	Input	Expected	Got	
~	1 3 10 17 57 6 2 7 10 15 57 246	10 57	10 57	*
~	1 6 1 2 3 4 5 6 2 1 6	1 6	1 6	~

Correct

```
Answer: (penalty regime: 0 %)
         #include <stdio.h>
     3 void findIntersection(int arr1[], int n1, int arr2[], int n2) {
4    int i = 0, j = 0;
5    int found = 0;
                while (i < n1 && j < n2) {
   if (arr1[i] < arr2[j]) {
  printf("%d ", arr1[i]);
found = 1;
                 if (!found) {
    printf("No Intersection");
}
                      scanf("%d", &n1);
int arr1[n1];
for (int i = 0; i < n1; i++) {
    scanf("%d", &arr1[i]);
}</pre>
                       scanf("%d", &n2);
int arr2[n2];
for (int i = 0; i < n2; i++) {
    scanf("%d", &arr2[i]);
}</pre>
```

	Input	Expected	Got	
~	1 3 10 17 57 6 2 7 10 15 57 246	10 57	10 57	~
~	1 6 1 2 3 4 5 6 2 1 6	1 6	1 6	~

Correct

	Input	Expected	Got	
~	3 1 3 5 4	1	1	~
~	10 1 4 6 8 12 14 15 20 21 25 1	1	1	~
~	10 1 2 3 5 11 14 16 24 28 29 0	0	0	~
~	10 0 2 3 7 13 14 15 20 24 25 10	1	1	~

Correct

	Input	Expected	Got	
~	3 1 3 5 4	1	1	~
~	10 1 4 6 8 12 14 15 20 21 25 1	1	1	~
~	10 1 2 3 5 11 14 16 24 28 29 0	0	0	~
~	10 0 2 3 7 13 14 15 20 24 25 10	1	1	~

Correct