



**RAJALAKSHMI**  
**ENGINEERING COLLEGE**  
An AUTONOMOUS Institution  
Affiliated to ANNA UNIVERSITY, Chennai

## **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

For example:

Input	Result
9	12

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int counter = 0;
4
5 void function(int n);
6
7 int main() {
8     int input = 0;
9
10
11     scanf("%d", &input);
12
13     function(input);
14     counter++;
15
16
17     printf("%d", counter);
18
19     return 0;
20 }
21
22 void function(int n) {
23     int i = 1;
24     counter++;
25
26     int s = 1;
27     counter++;
28
29     while (s <= n) {
30         counter++;
31         i++;
32         counter++;
33         s += 1;
34         counter++;
35     }
36 }
37
38
39
40 }
```

	Input	Expected	Got	
✓	9	12	12	✓
✓	4	9	9	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

[Finish review](#)

Jump to...

[Problem 2: Finding Complexity using Counter method ▶](#)

Answer:

```
1 #include <stdio.h>
2
3 int counter = 0;
4
5 void Factor(int num){
6     for(int i = 1; i <= num; i++){
7         counter++;
8         if(num % i == 0){
9             counter++;
10        }
11        counter++;
12    }
13 }
14
15 int main(){
16     int n = 0;
17
18     scanf("%d", &n);
19
20     Factor(n);
21     counter++;
22
23     printf("%d", counter);
24     return 0;
25 }
26 }
```

	Input	Expected	Got	
✓	12	31	31	✓
✓	25	54	54	✓
✓	4	12	12	✓

Passed all tests! ✓

**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 ▶](#)



Answer:

```
1 #include<stdio.h>
2
3 int counter = 0;
4 void function(int n)
5 {
6     int c= 0;
7
8     for(int i=n/2; i<n; i++){
9         counter++;
10
11         for(int j=1; j<n; j = 2 * j){
12             counter++;
13
14             for(int k=1; k<n; k = k * 2){
15                 counter++;
16                 c++;
17                 counter++;
18             }
19         }
20     }
21     counter++;
22 }
23 counter++;
24 }
25 counter++;
26 }
27
28 int main(){
29     int n = 0;
30
31     scanf("%d",&n);
32
33     function(n);
34     counter++;
35
36
37     printf("%d",counter);
38
39     return 0;
40 }
```

	Input	Expected	Got	
✓	4	30	30	✓
✓	10	212	212	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer:

```
1  #include <stdio.h>
2  int count = 0;
3  void reverse(int n)
4  {
5      int rev = 0, remainder;
6      count++;
7
8
9      while (n != 0)
10     {
11         count++;
12         remainder = n % 10;
13         count++;
14
15
16         rev = rev * 10 + remainder;
17         count++;
18
19         n /= 10;
20         count++;
21
22
23     }
24     count++;
25
26
27 }
28
29
30 int main(){
31     int n ;
32
33
34     scanf("%d",&n);
35     count++;
36     reverse(n);
37     printf("%d",count);
38
39 }
```

	Input	Expected	Got	
✓	12	11	11	✓
✓	1234	19	19	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

# WEEK-2

## GREEDY ALGORITHMS

Answer: (penalty regime: 0 %)

```
1
2 #include <stdio.h>
3 #include <limits.h>
4
5 int min(int a, int b) {
6     return (a < b) ? a : b;
7 }
8
9 int minChange(int V) {
10     int denominations[] = {1, 2, 5, 10, 20, 50, 100, 500, 1000};
11     int n = sizeof(denominations) / sizeof(denominations[0]);
12     int dp[V + 1];
13     dp[0] = 0;
14     for (int i = 1; i <= V; i++) {
15         dp[i] = INT_MAX;
16         for (int j = 0; j < n; j++) {
17             if (denominations[j] <= i) {
18                 dp[i] = min(dp[i], dp[i - denominations[j]] + 1);
19             }
20         }
21     }
22     return dp[V];
23 }
24
25 int main() {
26     int V;
27     scanf("%d", &V);
28     printf("%d\n", minChange(V));
29     return 0;
30 }
31 }
```

	Input	Expected	Got	
✓	49	5	5	✓

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int compare(const void* a, const void* b) {
5     return (*(int*)a - *(int*)b);
6 }
7
8 int findContentChildren(int* g, int gSize, int* s, int sSize) {
9
10     qsort(g, gSize, sizeof(int), compare);
11     qsort(s, sSize, sizeof(int), compare);
12
13     int i = 0, j = 0;
14     int satisfied = 1;
15
16
17     while (i < gSize && j < sSize) {
18         if (s[j] >= g[i])
19             j++;
20     }
21
22     satisfied++;
23     i++;
24 }
25
26 return satisfied;
27 }
28
29 int main() {
30     int g[] = {1, 2, 3};
31     int s[] = {1, 2};
32
33     int gSize = sizeof(g) / sizeof(g[0]);
34     int sSize = sizeof(s) / sizeof(s[0]);
35
36     int result = findContentChildren(g, gSize, s, sSize);
37     printf("%d\n", result);
38 }
```

	Input	Expected	Got	
✓	2	2	2	✓
	1 2			
	3			
	1 2 3			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 |
2 | #include<stdio.h>
3 | #include<math.h>
4 | int minDistance(int calories[],int n){
5 |     for(int i=0;i<n-1;i++){
6 |         for(int j=0;j<n-i-1;j++){
7 |             if(calories[j]<calories[j+1]){
8 |                 int temp=calories[j];
9 |                 calories[j]=calories[j+1];
10 |                calories[j+1]=temp;
11 |            }
12 |        }
13 |    }
14 |    int distance=0;
15 |    for(int i=0;i<n;i++){
16 |        distance+=pow(n,i)*calories[i];
17 |    }
18 |    return distance;
19 | }
20 | int main(){
21 |     int n;
22 |     scanf("%d",&n);
23 |     int calories[n];
24 |     for(int i=0;i<n;i++){
25 |         scanf("%d",&calories[i]);
26 |     }
27 |     int min=minDistance(calories,n);
28 |     printf("%d",min);
29 |     return 0;
30 | }
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 void swap(int* a, int* b) {
4     int temp = *a;
5     *a = *b;
6     *b = temp;
7 }
8
9 int partition(int arr[], int low, int high) {
10     int pivot = arr[high];
11     int i = (low - 1);
12
13     for (int j = low; j <= high - 1; j++) {
14         if (arr[j] < pivot) {
15             i++;
16             swap(&arr[i], &arr[j]);
17         }
18     }
19
20     swap(&arr[i + 1], &arr[high]);
21     return (i + 1);
22 }
23
24
25 void quickSort(int arr[], int low, int high) {
26     if (low < high) {
27         int pivot = partition(arr, low, high);
28
29         quickSort(arr, low, pivot - 1);
30         quickSort(arr, pivot + 1, high);
31     }
32 }
33
34 // Function to print the array
35 /*void printArray(int arr[], int size) {
36     for (int i = 0; i < size; i++) {
37         printf("%d ", arr[i]);
38     }
39     printf("\n");
40 }
41 */
42 int maximizeSum(int arr[], int n) {
43     quickSort(arr, 0, n-1);
44
45     int sum = 0;
46     for (int i = 0; i < n; i++) {
47         sum += arr[i] * i;
48     }
49
50     return sum;
51 }
52 int main() {
```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 void sortAscending(int arr[], int n) {
4     for (int i = 0; i < n - 1; i++) {
5         for (int j = i + 1; j < n; j++) {
6             if (arr[i] > arr[j]) {
7                 int temp = arr[i];
8                 arr[i] = arr[j];
9                 arr[j] = temp;
10            }
11        }
12    }
13 }
14
15 void sortDescending(int arr[], int n) {
16     for (int i = 0; i < n - 1; i++) {
17         for (int j = i + 1; j < n; j++) {
18             if (arr[i] < arr[j]) {
19                 int temp = arr[i];
20                 arr[i] = arr[j];
21                 arr[j] = temp;
22            }
23        }
24    }
25 }
26
27 int minProductSum(int array_One[], int array_Two[], int n) {
28     sortAscending(array_One, n);
29     sortDescending(array_Two, n);
30
31     int sum = 0;
32     for (int i = 0; i < n; i++) {
33         sum += array_One[i] * array_Two[i];
34     }
35     return sum;
36 }
37
38 int main() {
39     int n;
40
41     scanf("%d", &n);
42
43     int array_One[n], array_Two[n];
44
45     for (int i = 0; i < n; i++) {
46         scanf("%d", &array_One[i]);
47     }
48
49     for (int i = 0; i < n; i++) {
50         scanf("%d", &array_Two[i]);
51     }
52 }
```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

WEEK-3

DIVIDE AND CONQUIRE

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3
4 int countZeros(int arr[], int low, int high) {
5
6     if (low > high)
7         return 0;
8
9     if (arr[low] == 0 && arr[high] == 0)
10         return high - low + 1;
11
12     if (arr[low] == 1 && arr[high] == 1)
13         return 0;
14
15     int mid = (low + high) / 2;
16
17     return countZeros(arr, low, mid) + countZeros(arr, mid + 1, high);
18 }
19
20 int main() {
21     int m;
22
23
24     scanf("%d", &m);
25
26     int arr[m];
27
28
29     for (int i = 0; i < m; i++) {
30         scanf("%d", &arr[i]);
31     }
32
33     int zeroCount = countZeros(arr, 0, m - 1);
34     printf(" %d", zeroCount);
35
36     return 0;
37 }
38
```

	Input	Expected	Got	
✓	5 1 1 1 0 0	2	2	✓
✓	10 1 1 1 1 1 1 1 1 1 1 1	0	0	✓
✓	8 0 0 0 0 0 0 0 0	8	8	✓
✓	17 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0	2	2	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int nums[n];
8
9     for (int i = 0; i < n; i++) {
10         scanf("%d", &nums[i]);
11     }
12
13     int count = 0;
14     int candidate = 0;
15
16     for (int i = 0; i < n; i++) {
17         if (count == 0) {
18             candidate = nums[i];
19         }
20         if (nums[i] == candidate) {
21             count++;
22         } else {
23             count--;
24         }
25     }
26
27     printf("%d\n", candidate);
28
29     return 0;
30 }
31
```

	Input	Expected	Got	
✓	3	3	3	✓
	3 2 3			

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n, x;
5     scanf("%d", &n);
6     int arr[n];
7
8     for (int i = 0; i < n; i++) {
9         scanf("%d", &arr[i]);
10    }
11
12    scanf("%d", &x);
13
14    int left = 0, right = n - 1;
15    int floor = -1;
16
17    while (left <= right) {
18        int mid = left + (right - left) / 2;
19
20        if (arr[mid] == x) {
21            floor = arr[mid];
22            break;
23        }
24
25        if (arr[mid] < x) {
26            floor = arr[mid];
27            left = mid + 1;
28        } else {
29            right = mid - 1;
30        }
31    }
32
33    if (floor != -1) {
34        printf("%d\n", floor);
35    } else {
36        printf("No floor value found for %d in the array.\n", x);
37    }
38
39    return 0;
40 }
41
```

	Input	Expected	Got	
✓	6 1 2 8 10 12 19 5	2	2	✓
✓	5 10 22 85 108 129 100	85	85	✓
✓	7 3 5 7 9 11 13 15 10	9	9	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Finish review

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int main() {
4     int n, x;
5     scanf("%d", &n);
6     int arr[n];
7     for (int i = 0; i < n; i++) {
8         scanf("%d", &arr[i]);
9     }
10
11     scanf("%d", &x);
12
13     int left = 0, right = n - 1;
14     int found = 0;
15
16     while (left < right) {
17         int sum = arr[left] + arr[right];
18
19         if (sum == x) {
20             printf("%d\n", arr[left]);
21             printf("%d\n", arr[right]);
22             found = 1;
23             break;
24         }
25
26         if (sum < x) {
27             left++;
28         } else {
29             right--;
30         }
31     }
32
33     if (!found) {
34         printf("No");
35     }
36
37     return 0;
38 }
39
40
```

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

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer:

```
1 #include <stdio.h>
2
3 int main() {
4     int n;
5     scanf("%d", &n);
6
7     int a[n];
8
9
10    for (int i = 0; i < n; i++) {
11        scanf("%d", &a[i]);
12    }
13
14
15    for (int i = 0; i < n; i++) {
16        for (int j = i + 1; j < n; j++) {
17            if (a[j] < a[i]) {
18                // Swap a[i] and a[j]
19                int temp = a[i];
20                a[i] = a[j];
21                a[j] = temp;
22            }
23        }
24    }
25
26
27    for (int i = 0; i < n; i++) {
28        printf("%d ", a[i]);
29    }
30    printf("\n");
31
32    return 0;
33 }
34
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



# WEEK-4

## DYNAMIC PROGRAMING

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

```
1 #include <stdio.h>
2
3 long long count_ways(int n) {
4     long long dp[n + 1];
5
6     dp[0] = 1;
7     dp[1] = 1;
8     dp[2] = 1;
9     dp[3] = 2;
10
11
12
13     for (int i = 4; i <= n; i++) {
14         dp[i] = dp[i - 1] + dp[i - 3];
15     }
16
17     return dp[n];
18 }
19
20 int main() {
21     int n;
22     scanf("%d", &n);
23     printf("%lld\n", count_ways(n));
24     return 0;
25 }
```

	Input	Expected	Got	
✓	6	6	6	✓
✓	25	8641	8641	✓
✓	100	24382819596721629	24382819596721629	✓

Passed all tests! ✓

**Correct**

Marks for this submission: 10.00/10.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2 #define MAX 100
3
4
5 int max(int a, int b) {
6     return (a > b) ? a : b;
7 }
8
9
10 int maxMonetaryPath(int chessboard[MAX][MAX], int n) {
11     int dp[MAX][MAX];
12
13     for (int i = 0; i < n; i++) {
14         for (int j = 0; j < n; j++) {
15             dp[i][j] = 0;
16         }
17     }
18
19     dp[0][0] = chessboard[0][0];
20
21     for (int j = 1; j < n; j++) {
22         dp[0][j] = dp[0][j - 1] + chessboard[0][j];
23     }
24
25     for (int i = 1; i < n; i++) {
26         dp[i][0] = dp[i - 1][0] + chessboard[i][0];
27     }
28
29     for (int i = 1; i < n; i++) {
30         for (int j = 1; j < n; j++) {
31             dp[i][j] = chessboard[i][j] + max(dp[i - 1][j], dp[i][j - 1]);
32         }
33     }
34
35     return dp[n - 1][n - 1];
36 }
37
38
39 int main() {
40     int n;
41     int chessboard[MAX][MAX];
42
43     scanf("%d", &n);
44
45     for (int i = 0; i < n; i++) {
46         for (int j = 0; j < n; j++) {
```

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

Passed all tests! ✓

Correct

Marks for this submission: 10.00/10.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2 #include <string.h>
3
4
5 int longest_common_subsequence(char s1[], char s2[]) {
6     int m = strlen(s1);
7     int n = strlen(s2);
8     int dp[m + 1][n + 1];
9
10
11     for (int i = 0; i <= m; i++) {
12         for (int j = 0; j <= n; j++) {
13             if (i == 0 || j == 0) {
14                 dp[i][j] = 0;
15             } else if (s1[i - 1] == s2[j - 1]) {
16                 dp[i][j] = dp[i - 1][j - 1] + 1;
17             } else {
18                 dp[i][j] = (dp[i - 1][j] > dp[i][j - 1]) ? dp[i - 1][j] : dp[i][j - 1];
19             }
20         }
21     }
22
23     return dp[m][n];
24 }
25
26 int main() {
27     char s1[100], s2[100];
28
29     scanf("%s", s1);
30     scanf("%s", s2);
31
32
33     int result = longest_common_subsequence(s1, s2);
34
35
36     printf("%d", result);
37
38     return 0;
39 }
40
41
```

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

Passed all tests! ✓

**Correct**

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int longest_non_decreasing_subsequence(int arr[], int n) {
4     int dp[n];
5     int max_length = 1;
6
7     for (int i = 0; i < n; i++) {
8         dp[i] = 1;
9     }
10
11     for (int i = 1; i < n; i++) {
12         for (int j = 0; j < i; j++) {
13             if (arr[j] <= arr[i]) {
14                 dp[i] = (dp[i] > dp[j] + 1) ? dp[i] : dp[j] + 1;
15             }
16         }
17         if (dp[i] > max_length) {
18             max_length = dp[i];
19         }
20     }
21     return max_length;
22 }
23
24 int main() {
25     int n;
26
27     scanf("%d", &n);
28
29     int arr[n];
30
31     for (int i = 0; i < n; i++) {
32         scanf("%d", &arr[i]);
33     }
34
35     int result = longest_non_decreasing_subsequence(arr, n);
36     printf("%d", result);
37
38     return 0;
39 }
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

# WEEK-5

## COMPITATIVE PROGRAMMING

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3
4 int findDuplicate(int arr[], int n) {
5     int slow = arr[0];
6     int fast = arr[arr[0]];
7
8
9     while (slow != fast) {
10         slow = arr[slow];
11         fast = arr[arr[fast]];
12     }
13
14
15     fast = 0;
16     while (slow != fast) {
17         slow = arr[slow];
18         fast = arr[fast];
19     }
20
21     return slow;
22 }
23
24 int main() {
25     int n;
26
27     scanf("%d", &n);
28
29     int arr[n];
30
31
32
33     for (int i = 0; i < n; i++) {
34         scanf("%d", &arr[i]);
35     }
36
37
38     int duplicate = findDuplicate(arr, n);
39     printf("%d\n", duplicate);
40
41     return 0;
42 }
43
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int findDuplicate(int arr[], int n) {
4     int slow = arr[0];
5     int fast = arr[arr[0]];
6
7     // Phase 1: Detect cycle
8     while (slow != fast) {
9         slow = arr[slow];
10        fast = arr[arr[fast]];
11    }
12
13    // Phase 2: Find entry point of cycle (duplicate)
14    fast = 0;
15    while (slow != fast) {
16        slow = arr[slow];
17        fast = arr[fast];
18    }
19
20    return slow;
21 }
22
23 int main() {
24     int n;
25     scanf("%d", &n);
26     int arr[n];
27
28     for (int i = 0; i < n; i++) {
29         scanf("%d", &arr[i]);
30     }
31
32     int duplicate = findDuplicate(arr, n);
33     printf("%d", duplicate);
34
35     return 0;
36 }
37
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 void findIntersection(int arr1[], int n1, int arr2[], int n2) {
4     int i = 0, j = 0;
5     int found = 0;
6
7     while (i < n1 && j < n2) {
8         if (arr1[i] < arr2[j]) {
9             i++;
10        } else if (arr2[j] < arr1[i]) {
11            j++;
12        } else {
13            // Intersection found
14            printf("%d ", arr1[i]);
15            found = 1;
16            i++;
17            j++;
18        }
19    }
20
21    if (!found) {
22        printf("No Intersection");
23    }
24    printf("\n");
25 }
26
27 int main() {
28     int T;
29     scanf("%d", &T);
30
31     while (T--) {
32         int n1, n2;
33
34         scanf("%d", &n1);
35         int arr1[n1];
36         for (int i = 0; i < n1; i++) {
37             scanf("%d", &arr1[i]);
38         }
39
40         scanf("%d", &n2);
41         int arr2[n2];
42         for (int i = 0; i < n2; i++) {
43             scanf("%d", &arr2[i]);
44         }
45
46         findIntersection(arr1, n1, arr2, n2);
47     }
48
49     return 0;
50 }
51
52
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 void findIntersection(int arr1[], int n1, int arr2[], int n2) {
4     int i = 0, j = 0;
5     int found = 0;
6
7     while (i < n1 && j < n2) {
8         if (arr1[i] < arr2[j]) {
9             i++;
10        } else if (arr2[j] < arr1[i]) {
11            j++;
12        } else {
13
14            printf("%d ", arr1[i]);
15            found = 1;
16            i++;
17            j++;
18        }
19    }
20
21    if (!found) {
22        printf("No Intersection");
23    }
24    printf("\n");
25 }
26
27 int main() {
28     int T;
29     scanf("%d", &T);
30
31     while (T--) {
32         int n1, n2;
33
34
35         scanf("%d", &n1);
36         int arr1[n1];
37         for (int i = 0; i < n1; i++) {
38             scanf("%d", &arr1[i]);
39         }
40
41
42         scanf("%d", &n2);
43         int arr2[n2];
44         for (int i = 0; i < n2; i++) {
45             scanf("%d", &arr2[i]);
46         }
47
48
49         findIntersection(arr1, n1, arr2, n2);
50     }
51
52     return 0;
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.



Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int findPairWithDifference(int arr[], int n, int k) {
4     int i = 0, j = 1;
5
6     while (i < n && j < n) {
7         int diff = arr[j] - arr[i];
8
9         if (i != j && diff == k) {
10             return 1;
11         }
12         else if (diff < k) {
13             j++;
14         }
15         else {
16             i++;
17         }
18     }
19
20     return 0;
21 }
22
23
24 int main() {
25     int n, k;
26     scanf("%d", &n);
27     int arr[n];
28
29     for (int i = 0; i < n; i++) {
30         scanf("%d", &arr[i]);
31     }
32
33     scanf("%d", &k);
34     int result = findPairWithDifference(arr, n, k);
35     printf("%d\n", result);
36
37     return 0;
38 }
39
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.

Answer: (penalty regime: 0 %)

```
1 #include <stdio.h>
2
3 int findPairWithDifference(int arr[], int n, int k) {
4     int i = 0, j = 1;
5
6     while (j < n) {
7         int diff = arr[j] - arr[i];
8
9         if (i != j && diff == k) {
10             return 1;
11         }
12         else if (diff < k) {
13             j++;
14         }
15         else {
16             i++;
17             if (i == j) {
18                 j++;
19             }
20         }
21     }
22
23     return 0;
24 }
25
26
27
28
29
30 int main() {
31     int n, k;
32     scanf("%d", &n);
33     int arr[n];
34
35     for (int i = 0; i < n; i++) {
36         scanf("%d", &arr[i]);
37     }
38
39     scanf("%d", &k);
40     int result = findPairWithDifference(arr, n, k);
41     printf("%d", result);
42
43     return 0;
44 }
45
```

	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	✓

Passed all tests! ✓

Correct

Marks for this submission: 1.00/1.00.