[CODING ROUND QUESTION & ANSWERS](https://t.me/campusdrive)

## [Find the distinct elements in a given array. (Assume size of an array n<=20) Sample Input:](https://t.me/campusdrive)

* [9 = size of an array](https://t.me/campusdrive)
* [2 3 4 5 6 1 2 3 4 = array elements](https://t.me/campusdrive)

## [Sample Output:](https://t.me/campusdrive)

* [2 3 4 5 6 1](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[// C program to print all distinct elements in a given array #include](https://t.me/campusdrive)

[void distict\_elements(int a[], int n); int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int size\_array, i, arr[20];](https://t.me/campusdrive)

[// Get the array size scanf(“%d”, &size\_array)](https://t.me/campusdrive)

[// Get the array elements for(i=0; i<size\_array; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[scanf(“%d”, &arr[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Function call to print the distinct elements in an array distict\_elements(arr, size\_array);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[void distict\_elements(int a[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j;](https://t.me/campusdrive)

[// Pick all elements one by one for (i=0; i<n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[// Check if the picked element is already printed for (j=0; j<i; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (a[i] == a[j])](https://t.me/campusdrive)

[break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// If not printed earlier, then print it if (i == j)](https://t.me/campusdrive)

[printf(“%d “, a[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Program to sort array in ascending & descending order.](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[5](https://t.me/campusdrive)

[8 6 9 2 7](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[2 6 7 8 9](https://t.me/campusdrive)

[9 8 7 6 2](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[// C program to sort the given array elements in ascending and descending order #include](https://t.me/campusdrive)

[int main(void)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int arr[10], i=0, j=0, size, temp;](https://t.me/campusdrive)

[// Get the size of an array scanf (“%d”, &size);](https://t.me/campusdrive)

[// Get the array elements as an input for (i = 0; i <size; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[scanf (“%d”, &arr[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Sorting elements in ascending order for (j=0 ; j<(size-1) ; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (i=0 ; i<(size-1) ; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (arr[i+1] < arr[i])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[temp = arr[i]; arr[i] = arr[i + 1]; arr[i + 1] = temp;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Print the elements from index value 0 to (size-1) –> ascending order for (i=0 ; i {](https://t.me/campusdrive)

[printf (“%d “, arr[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[order](https://t.me/campusdrive)

[// Print the elements from the index value (size-1) to 0 –> descending](https://t.me/campusdrive)

[for (i=size-1; i>=0 ; i–)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf (“%d “, arr[i]);](https://t.me/campusdrive)

## [Sort first half in ascending and second half in descending order.](https://t.me/campusdrive)

[**Example 1:**](https://t.me/campusdrive)

[Input:](https://t.me/campusdrive)

[8](https://t.me/campusdrive)

[2 4 7 9 3 1 6 8](https://t.me/campusdrive)

[Output:](https://t.me/campusdrive)

[1 2 3 4 9 8 7 6](https://t.me/campusdrive)

## [Example 2:](https://t.me/campusdrive)

[Input:](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

[1 2 3 4 5 6](https://t.me/campusdrive)

[Output:](https://t.me/campusdrive)

[1 2 3 6 5 4](https://t.me/campusdrive)

## [Algorithm:](https://t.me/campusdrive)

1. [Sort the given array.](https://t.me/campusdrive)
2. [Run a loop up to half the length of the array and print the elements of the sorted array.](https://t.me/campusdrive)
3. [Run a loop from the last index of the array to the middle of the array and print the elements in reverse order.](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[void sorting\_elements(int arr[], int n); void display(int arr[], int n);](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int size, arr[20], i; scanf(“%d”, &size); for(i=0; i<size; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[scanf(“%d”, &arr[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[display(arr, size); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Sort the elements in the ascending order void sorting\_elements(int arr[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i,j,temp;](https://t.me/campusdrive)

[for (j=0 ; j<(n-1) ; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (i=0 ; i<(n-1) ; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (arr[i+1] < arr[i])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[temp = arr[i]; arr[i] = arr[i + 1]; arr[i + 1] = temp;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Display the sorted elements void display(int arr[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[sorting\_elements(arr, n); int i, j](https://t.me/campusdrive)

[// Print the first half as such (i.e. from index 0 to midlle) for (i=0; i<n/2; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “, arr[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Print the second half in the reverse order (i.e. from n-1 to midlle) for (j=n-1; j>=n/2; j–)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “, arr[j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the following pattern pattern](https://t.me/campusdrive)

[**Input: Output:**](https://t.me/campusdrive)

[3 4](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[44](https://t.me/campusdrive)

[555](https://t.me/campusdrive)

[6666](https://t.me/campusdrive)

[555](https://t.me/campusdrive)

[44](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Input : Output:](https://t.me/campusdrive)

[4 4](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

[55](https://t.me/campusdrive)

[666](https://t.me/campusdrive)

[7777](https://t.me/campusdrive)

[666](https://t.me/campusdrive)

[55](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

[**Program:** #include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i,j,s,N,count=0; scanf(“%d%d”,&s,&N); for(i=s;count<4;count++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=0;j<count+1;j++)](https://t.me/campusdrive)

[printf(“%d”,i);](https://t.me/campusdrive)

[printf(“\n”); i=i+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[for(i=s+N-2;count>0;count–)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=0;j<count-1;j++) printf(“%d”,i);](https://t.me/campusdrive)

[printf(“\n”); i=i-1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the following pattern pattern](https://t.me/campusdrive)

[**Input :**](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

[2\*2](https://t.me/campusdrive)

[3\*3\*3](https://t.me/campusdrive)

[3\*3\*3](https://t.me/campusdrive)

[2\*2](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

## [Input :](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

[2\*2](https://t.me/campusdrive)

[3\*3\*3](https://t.me/campusdrive)

[4\*4\*4\*4](https://t.me/campusdrive)

[4\*4\*4\*4](https://t.me/campusdrive)

[3\*3\*3](https://t.me/campusdrive)

[2\*2](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

[**Program:** #include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i,j,k,N,count=0; scanf(“%d”,&N); for(i=1;i<=N;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[k=1;](https://t.me/campusdrive)

[for(j=0;j<i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,i); if(k<i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\*”); k=k+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[for(i=N;i>0;i–)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[k=1;](https://t.me/campusdrive)

[for(j=0;j<i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,i); if(k<i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\*”); k=k+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the below pattern](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

[2\*3](https://t.me/campusdrive)

[4\*5\*6](https://t.me/campusdrive)

[7\*8\*9\*10](https://t.me/campusdrive)

[7\*8\*9\*10](https://t.me/campusdrive)

[4\*5\*6](https://t.me/campusdrive)

[2\*3](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include int main() {](https://t.me/campusdrive)

[int i,j,count=1,n; printf(“Enter a number\n”); scanf(“%d”,&n); for(i=1;i<=n;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=1;j<=i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(j<i) printf(“%d\*”,count++); else printf(“%d”,count++);](https://t.me/campusdrive)

[} printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[count=count-n; for(i=n;i>=1;i–)](https://t.me/campusdrive)

[{ for(j=1;j<=i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(j<i) printf(“%d\*”,count++); else printf(“%d”,count++);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[count=(count+1)-2\*i; printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the following pattern](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[3 4](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[44](https://t.me/campusdrive)

[555](https://t.me/campusdrive)

[6666](https://t.me/campusdrive)

[6666](https://t.me/campusdrive)

[555](https://t.me/campusdrive)

[44](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[**Program:** #include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i,j,s,N,count=0; scanf(“%d%d”,&s,&N); for(i=s;count<4;count++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=0;j<count+1;j++) printf(“%d”,i);](https://t.me/campusdrive)

[printf(“\n”); i=i+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[for(i=s+N-2;count>0;count–)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=0;j<count-1;j++) printf(“%d”,i);](https://t.me/campusdrive)

[printf(“\n”); i=i-1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the below pattern](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[5](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[1](https://t.me/campusdrive)

[3\*2](https://t.me/campusdrive)

[4\*5\*6](https://t.me/campusdrive)

[10\*9\*8\*7](https://t.me/campusdrive)

[11\*12\*13\*14\*15](https://t.me/campusdrive)

[**Program:** #include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i,j,k,l=1,N,d,r,count=0; scanf(“%d”,&N); for(i=1;i<=N;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[k=1;](https://t.me/campusdrive)

[d=i%2; r=l+i-1;](https://t.me/campusdrive)

[for(j=0;j<i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(d==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,r); r–;](https://t.me/campusdrive)

[if(k<i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\*”); k=k+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[l++;](https://t.me/campusdrive)

[continue;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”,l); l++;](https://t.me/campusdrive)

[if(k<i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\*”); k=k+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the below pattern](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[1\*2\*3\*4\*17\*18\*19\*20](https://t.me/campusdrive)

[– -5\*6\*7\*14\*15\*16](https://t.me/campusdrive)

[– – – -8\*9\*12\*13](https://t.me/campusdrive)

[– – – – – -10\*11](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[void pattern(int); int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n; scanf(“%d”, &n); pattern(n); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[void pattern(int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, k, s, a = 1,b = n\*n + 1; for (i = n; i >= 1; i–) {](https://t.me/campusdrive)

[for (s = 0; s < n – i; s++)](https://t.me/campusdrive)

[printf(“–“); for (j = 0; j < i; j++)](https://t.me/campusdrive)

[printf(“%d\*”, a++); for (k = 0; k < i – 1; k++)](https://t.me/campusdrive)

[printf(“%d\*”, b++);](https://t.me/campusdrive)

[printf(“%d\n”, b); // last b should without \* b -= 2\*(i – 1);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print pattern](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[3 3 3](https://t.me/campusdrive)

[3 1 3](https://t.me/campusdrive)

[3 2 3](https://t.me/campusdrive)

[3 3 3](https://t.me/campusdrive)

[**Program:** #include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, n, c=1; scanf(“%d”, &n); for(i=1; i<=n+1; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=1; j<=n; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(i!=1 && j==n-1)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “, c);](https://t.me/campusdrive)

[c++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else](https://t.me/campusdrive)

[printf(“%d “, n);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

1. [**Paranthesis checker:** Check whether the given expression is valid or not(only parenthesis symbol).](https://t.me/campusdrive)

[**Test Case: 1** Input: “(( ))” Output: Valid **Test Case: 2** Input: “()(“ Output: Invalid](https://t.me/campusdrive)

[**Program:** #include #include #include](https://t.me/campusdrive)

[int top = -1; char stack[100]; void push(char);](https://t.me/campusdrive)

[void pop(); void find\_top(); void main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i;](https://t.me/campusdrive)

[char a[100];](https://t.me/campusdrive)

[scanf(“%s”, &a);](https://t.me/campusdrive)

[for (i = 0; a[i] != ‘\0’; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (a[i] == ‘(‘)](https://t.me/campusdrive)

[push(a[i]);](https://t.me/campusdrive)

[else if (a[i] == ‘)’)](https://t.me/campusdrive)

[pop();](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[find\_top();](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// to push elements in stack void push(char a)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[top++; stack[top] = a;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// to pop elements from stack void pop()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (top == -1)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“Invalid”); exit(0);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else](https://t.me/campusdrive)

[top–;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// to find top element of stack](https://t.me/campusdrive)

[void find\_top()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (top == -1) printf(“Valid”); else](https://t.me/campusdrive)

[printf(“Invalid”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the transpose of a Matrix:](https://t.me/campusdrive)

[#include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int a[10][10], transpose[10][10], r, c, i, j; printf(“Enter rows and columns of matrix: “); scanf(“%d %d”, &r, &c);](https://t.me/campusdrive)

[// Storing elements of the matrix printf(“\nEnter elements of matrix:\n”); for(i=0; i<r; ++i)](https://t.me/campusdrive)

[for(j=0; j<c; ++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“Enter element a%d%d: “,i+1, j+1); scanf(“%d”, &a[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Displaying the matrix a[][] \*/ printf(“\nEntered Matrix: \n”); for(i=0; i<r; ++i)](https://t.me/campusdrive)

[for(j=0; j<c; ++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “, a[i][j]); if (j == c-1)](https://t.me/campusdrive)

[printf(“\n\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Finding the transpose of matrix a for(i=0; i<r; ++i)](https://t.me/campusdrive)

[for(j=0; j<c; ++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[transpose[j][i] = a[i][j];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Displaying the transpose of matrix a printf(“\nTranspose of Matrix:\n”); for(i=0; i<c; ++i)](https://t.me/campusdrive)

[for(j=0; j<r; ++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “,transpose[i][j]); if(j==r-1)](https://t.me/campusdrive)

[printf(“\n\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Matrix Addition:](https://t.me/campusdrive)

[**Program:** #include int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int r, c, a[100][100], b[100][100], sum[100][100], i, j;](https://t.me/campusdrive)

[printf(“Enter number of rows (between 1 and 100): “); scanf(“%d”, &r);](https://t.me/campusdrive)

[printf(“Enter number of columns (between 1 and 100): “); scanf(“%d”, &c);](https://t.me/campusdrive)

[printf(“\nEnter elements of 1st matrix:\n”); for(i=0; i<r; ++i)](https://t.me/campusdrive)

[for(j=0; j<c; ++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“Enter element a%d%d: “,i+1,j+1); scanf(“%d”,&a[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“Enter elements of 2nd matrix:\n”); for(i=0; i<r; ++i)](https://t.me/campusdrive)

[for(j=0; j<c; ++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“Enter element a%d%d: “,i+1, j+1); scanf(“%d”, &b[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Adding Two matrices for(i=0;i<r;++i)](https://t.me/campusdrive)

[for(j=0;j<c;++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[sum[i][j]=a[i][j]+b[i][j];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[// Displaying the result](https://t.me/campusdrive)

[printf(“\nSum of two matrix is: \n\n”); for(i=0;i<r;++i)](https://t.me/campusdrive)

[{ for(j=0;j<c;++j)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “,sum[i][j]); if(j==c-1)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\n\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [Amcat Automata Fix Questions](https://t.me/campusdrive)

1. [Find the **syntax error** in the below code without modifying the logic.](https://t.me/campusdrive)

[#include <stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[float x = 1.1; switch (x)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[case 1: printf(“Choice is 1”); break;](https://t.me/campusdrive)

[default: printf(“Invalid choice”); break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Answer:](https://t.me/campusdrive)

[#include <stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

## [int x = 1;](https://t.me/campusdrive)

[switch (x)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[case 1: printf(“Choice is 1”); break;](https://t.me/campusdrive)

[default: printf(“Invalid choice”); break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[The expression used in the switch must be an integral type (int, char, and enum). Any other type of expression is not allowed.](https://t.me/campusdrive)

1. [Find the **logical error** in the below code.](https://t.me/campusdrive)

[void main () { int i, j, n = 5;](https://t.me/campusdrive)

[for(i=1; i<n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j=i;j<n;j++);](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”, i);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[**Solution:** void main () { int i, j, n = 5;](https://t.me/campusdrive)

[for(i=1; i<n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

## [for(j=i;j<n;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”, i);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[we use a semicolon in C statement to tell the compiler where’s the end of our statement. Second for loop executes one time.](https://t.me/campusdrive)

1. [Complete the below **code by reusing the existing function**.](https://t.me/campusdrive)

[Find the index of equilibrium element in the given array. In an array equilibrium element is the one where the sum of all the elements to the left side is equal to the sum of all the elements in the right side.](https://t.me/campusdrive)

## [Return Value:](https://t.me/campusdrive)

1. [Return -1 if no equilibrium element is found](https://t.me/campusdrive)
2. [In case there is more than one equilibrium element, return the element with least index value.](https://t.me/campusdrive)

[You are required to complete the given code by reusing the existing function. You can click on Compile & run anytime to check the compilation/execution status of the program you can use printf to debug your code. The submitted code should be logically/syntactically correct and pass all the test cases.](https://t.me/campusdrive)

## [Code approach For the question:](https://t.me/campusdrive)

[You will need to correct the given implementation.](https://t.me/campusdrive)

[We do not expect you to modify the approach or incorporate any additional library methods.](https://t.me/campusdrive)

## [Test Case:](https://t.me/campusdrive)

[a[] = {1,2,3,4,3,3}. 4 is the equilibrium element since its left side sum (1+2+3) is equal to its right side sum (3+3)](https://t.me/campusdrive)

[#include <stdio.h>](https://t.me/campusdrive)

## [// Return the sum of elements from index 0 to (idx – 1)](https://t.me/campusdrive)

[int left\_side\_sum(int a[], int n, int idx)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int sum = 0, i;](https://t.me/campusdrive)

[for(i = 0; i < idx; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[sum += a[i];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return sum;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [// Return the sum of elements from index (idx + 1) to (n – 1)](https://t.me/campusdrive)

[int right\_side\_sum(int a[], int n, int idx)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int sum = 0, i;](https://t.me/campusdrive)

[for(i = idx + 1; i < n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[sum += a[i];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return sum;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [// returns -1 if no equilibrium index found](https://t.me/campusdrive)

[int findEquilibriumIndex(int a[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[// Type your code here](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main() {](https://t.me/campusdrive)

[//code int a[10], n, i;](https://t.me/campusdrive)

[// get the elements count scanf(“%d”, &n);](https://t.me/campusdrive)

[// get the array elements for(i=0; i<n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[scanf(“%d”, &a[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int equiIndex = findEquilibriumIndex(a, n); if(equiIndex != -1) {](https://t.me/campusdrive)

[printf(“%d”, a[equiIndex]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[**// Return the sum of elements from index 0 to (idx – 1)**](https://t.me/campusdrive)

[int left\_side\_sum(int a[], int n, int idx)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int sum = 0, i;](https://t.me/campusdrive)

[for(i = 0; i < idx; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[sum += a[i];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return sum;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [// Return the sum of elements from index (idx + 1) to (n – 1)](https://t.me/campusdrive)

[int right\_side\_sum(int a[], int n, int idx)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int sum = 0, i;](https://t.me/campusdrive)

[for(i = idx + 1; i < n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[sum += a[i];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return sum;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [// returns -1 if no equilibrium index found](https://t.me/campusdrive)

[int findEquilibriumIndex(int a[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[// Type your code here](https://t.me/campusdrive)

## [int i;](https://t.me/campusdrive)

[**for(i = 0; i < n; i++)**](https://t.me/campusdrive)

## [{](https://t.me/campusdrive)

[**if(left\_side\_sum(a, n, i) == right\_side\_sum(a, n, i))**](https://t.me/campusdrive)

## [{](https://t.me/campusdrive)

[**return i;**](https://t.me/campusdrive)

## [}](https://t.me/campusdrive)

[**}**](https://t.me/campusdrive)

## [return -1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main() {](https://t.me/campusdrive)

[//code int a[10], n, i;](https://t.me/campusdrive)

[// get the elements count scanf(“%d”, &n);](https://t.me/campusdrive)

[// get the array elements for(i=0; i<n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[scanf(“%d”, &a[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int equiIndex = findEquilibriumIndex(a, n); if(equiIndex != -1) {](https://t.me/campusdrive)

[printf(“%d”, a[equiIndex]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Check for syntax error/ logical error and correct the error to get the desired output.](https://t.me/campusdrive)

[Given n, print from n to 0 int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n; scanf(“%d”, &n);](https://t.me/campusdrive)

[unsigned int i = n; while(i >= 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d\n”, i); i–;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Input: 4](https://t.me/campusdrive)

[**Output: Infinite loop**](https://t.me/campusdrive)

[**Answer:** Error – Logical error](https://t.me/campusdrive)

[unsigned int i = n; unsigned integer ranges from 0 to 65535, which will be taken in the cyclic order. So i– will keep repeating in a cyclic way. The loop will never be terminated. So it should be written as **int i = n;**](https://t.me/campusdrive)

## [Find the factorial of a given number.](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[long int fact = 1, n, i; scanf(“%d”, &n);](https://t.me/campusdrive)

[for(i =1; i <= n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[fact = fact \* i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, fact);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Input: 20](https://t.me/campusdrive)

[**Output: -2102132736**](https://t.me/campusdrive)

[**Answer:** Error – Logical error](https://t.me/campusdrive)

[The fact and n are declared as long int, so in scanf and printf %ld should be used in place of %d.](https://t.me/campusdrive)

## [Check whether the below program print the below pattern](https://t.me/campusdrive)

[1111](https://t.me/campusdrive)

[222](https://t.me/campusdrive)

[33](https://t.me/campusdrive)

[void main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, n; scanf(“%d”, &n); for(i = 1; i<n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(j = 1; j<n; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”, i);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Input: 3 Output:](https://t.me/campusdrive)

[**111**](https://t.me/campusdrive)

## [222](https://t.me/campusdrive)

[**333**](https://t.me/campusdrive)

[**Answer:** Error: Logical error](https://t.me/campusdrive)

[The inner for loop has to be written in this way: **for(j = i-1; j<n; j++)**](https://t.me/campusdrive)

## [Find the greatest of three numbers.](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int num1, num2, num3;](https://t.me/campusdrive)

[scanf(“%d %d %d”, &num1,&num2,&num3); if (num1 > num2) && (num1 > num3)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”, num1);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[elseif(num2>num3)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, num2)](https://t.me/campusdrive)

[printf(“%d”, num3);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[**Answer:** Error: Syntax error](https://t.me/campusdrive)

[if (num1 > num2) && (num1 > num3) à it has to be written as if ((num1 > num2) && (num1 > num3)) and this line **elseif**(num2>num3) should be rewritten as **else if**(num2>num3)](https://t.me/campusdrive)

## [Fix the error, recompile and match against the output provided.](https://t.me/campusdrive)

[int main(void)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“This is a \”buggy” program\n”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Corrected program:](https://t.me/campusdrive)

[int main(void)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“This is a \”buggy\” program\n”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

1. [**Code reuse:** Convert Binary to Decimal by using the existing function. void binarytodecimal(number)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[// Type your code here](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[void main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int num; scanf(“%d”, &num);](https://t.me/campusdrive)

[printf(“%d”, binarytodecimal(num);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Answer:](https://t.me/campusdrive)

[void binarytodecimal(number)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int dval=0, base=1, rem; while(number > 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[rem = number % 10; dval = dval + rem \* base;](https://t.me/campusdrive)

[num = number / 10; base = base \* 2;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return dval;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Print the prime numbers from an array up to given value n by using existing function.](https://t.me/campusdrive)

[int isprime(int num)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[// type your code here](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n, m, arr[100], size=0, i; scanf(“%d”, &n);](https://t.me/campusdrive)

[for(m = 2; m <= n; m++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(isprime(m))](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[arr[size++]= m;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[for(i = 0; i < size; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d\n”, arr[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Answer:](https://t.me/campusdrive)

[int isprime(int num)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i;](https://t.me/campusdrive)

[int isprime = 1;](https://t.me/campusdrive)

[for(i = 2; i <= num / 2; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(num % i == 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[isprime = 0; break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return isprime;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [– AMCAT automata questions:](https://t.me/campusdrive)

[**Find the number of all possible triplets in the array that can form the triangle (condition is a + b > c).**](https://t.me/campusdrive)

[#include”stdio.h”](https://t.me/campusdrive)

[int arr[100], n, n1, n2, i, j, k; int a,b,c;](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[scanf(“%d”,&n); for(i=0;i<n;i++) scanf(“%d”,&arr[i]); for(i=0;i<n-2;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[a=arr[i]; for(j=i+1;j<n-1;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[b=arr[j]; for(k=j+1;k<n;k++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[c=arr[k];](https://t.me/campusdrive)

[if( ((a + b)>c) && ((a + c)>b) && ((b + c)>a) )](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d %d %d “,a,b,c); printf(“Yes\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d %d %d “,a,b,c); printf(“No\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [– AMCAT automata question](https://t.me/campusdrive)

[**Print all the prime numbers which are below the given number separated by comma**](https://t.me/campusdrive)

[Input: 50](https://t.me/campusdrive)

[#include”stdio.h” int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n,i,j,ct=0; scanf(“%d”,&n); for(i=2;i<=n;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[ct=0; for(j=2;j<i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(i%j==0)](https://t.me/campusdrive)

[{ ct=1; break; }](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(ct==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(i>2) printf(“, “);](https://t.me/campusdrive)

[printf(“%d”,i);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[Output: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47](https://t.me/campusdrive)

## [– AMCAT automata questions](https://t.me/campusdrive)

[**Program to find the GCD of two Integers.**](https://t.me/campusdrive)

[#include”stdio.h” int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int m, n, i, gcd; scanf(“%d %d”, &m, &n);](https://t.me/campusdrive)

[for(i=1; i <=m && i <= n; ++i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[// Checks if i is factor of both integers if(m%i==0 && n%i==0)](https://t.me/campusdrive)

[gcd = i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“ %d”, gcd); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [– AMCAT automata questions](https://t.me/campusdrive)

[**Program to find out sum of digits of given number.**](https://t.me/campusdrive)

[#include “stdio.h” void main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[long n, t, digit;](https://t.me/campusdrive)

[int sum = 0; scanf(“%ld”, &n); t = n;](https://t.me/campusdrive)

[while (n > 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[digit = n % 10; sum = sum + digit; n /= 10;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, sum);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [– AMCAT automata questions Print the pattern If input is 5](https://t.me/campusdrive)

[**1**](https://t.me/campusdrive)

## [3\*2](https://t.me/campusdrive)

[**4\*5\*6**](https://t.me/campusdrive)

## [10\*9\*8\*7](https://t.me/campusdrive)

[**11\*12\*13\*14\*15**](https://t.me/campusdrive)

[#include ”stdio.h” int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, k, l=1, N, d, r, count=0; scanf(“%d”, &N);](https://t.me/campusdrive)

[for(i=1; i<=N; i++)](https://t.me/campusdrive)

[{ k=1;](https://t.me/campusdrive)

[d=i%2; r=l+i-1;](https://t.me/campusdrive)

[for(j=0;j<i;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(d==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,r); r–;](https://t.me/campusdrive)

[if(k<i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\*”); k=k+1;](https://t.me/campusdrive)

[} l++;](https://t.me/campusdrive)

[continue;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”,l); l++;](https://t.me/campusdrive)

[if(k<i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“\*”); k=k+1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [– AMCAT automata questions](https://t.me/campusdrive)

[Mooshak the mouse has been placed in a maze. There is a huge chunk of cheese somewhere in the maze. The maze is represented as a two-dimensional array of integers, where 0 represents walls.1 represents paths where Mooshak can move and 9 represents the huge chunk of cheese. Mooshak starts in the top left corner at 0.](https://t.me/campusdrive)

[Write a method is Path of class Maze Path to determine if Mooshak can reach the huge chunk of cheese. The input to is Path consists of a two-dimensional array and for the maze matrix. the method should return 1 if there is a path from Mooshak to the cheese and 0 if not Mooshak is not allowed to leave the maze or climb on walls.](https://t.me/campusdrive)

[EX: 8 by 8(8\*8) matrix maze where Mooshak can get the cheese. 1 0 1 1 1 0 0 1](https://t.me/campusdrive)

[1 0 0 0 1 1 1 1](https://t.me/campusdrive)

[1 0 0 0 0 0 0 0](https://t.me/campusdrive)

[1 0 1 0 9 0 1 1](https://t.me/campusdrive)

[1 1 1 0 1 0 0 1](https://t.me/campusdrive)

[1 0 1 0 1 1 0 1](https://t.me/campusdrive)

[1 0 0 0 0 1 0 1](https://t.me/campusdrive)

[1 1 1 1 1 1 1 1](https://t.me/campusdrive)

## [Test Cases:](https://t.me/campusdrive)

[**Test Case 1:**](https://t.me/campusdrive)

[Input: [[0,0,0],[9,1,1],[0,1,1]]](https://t.me/campusdrive)

[Expected return value: 0](https://t.me/campusdrive)

## [Explanation:](https://t.me/campusdrive)

[The piece of cheese is placed at(1,0) on the grid Mooshak can move from (0,0) to (1,0) to reach it or can move from (0,0) to (0,1) to (1,1) to (1,0)](https://t.me/campusdrive)

[#include”stdlib.h” #include”stdio.h”](https://t.me/campusdrive)

[int path(int maze[3][3]); int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i,j,maze[3][3], result=0; for(int i=0;i<3;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(int j=0;j<3;j++)](https://t.me/campusdrive)

[scanf (“%d”, &maze[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“input\n”); for(int i=0;i<3;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for(int j=0;j<3;j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “,maze[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“\n”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, path(maze)); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int path(int m[3][3])](https://t.me/campusdrive)

[{ for(static int i=0;i<3;i++)](https://t.me/campusdrive)

[{ for(int j=0;j<3;j++)](https://t.me/campusdrive)

[{ if((m[i][j]==1))](https://t.me/campusdrive)

[int path(m[3][3]);](https://t.me/campusdrive)

[if(m[i][j]==0)](https://t.me/campusdrive)

[return 0; if(m[i][j]==9) return 1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [– AMCAT automata questions](https://t.me/campusdrive)

[Program to print all distinct elements of given input arrays. Also print the total of the distinct elements.](https://t.me/campusdrive)

## [Input:](https://t.me/campusdrive)

[Arr1 = {1,2,3,4,5}](https://t.me/campusdrive)

[Arr 2 = {2,6,8,10}](https://t.me/campusdrive)

[#include”stdio.h”](https://t.me/campusdrive)

[int Not\_common (int \*arr1, int \*arr2, int l1, int l2)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int count =0, flag1, i, j; for(i=0; i<l1; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag1=0;](https://t.me/campusdrive)

[for(j=0; j<l2; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(arr1[i] == arr2[j])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag1=1; break;](https://t.me/campusdrive)

[}}](https://t.me/campusdrive)

[if(flag1 ==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[count++; printf(“%d,”, arr1[i]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[for(i=0; i<l2; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag1=0;](https://t.me/campusdrive)

[for(j=0; j<l1; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(arr2[i] == arr1[j])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag1=1; break;](https://t.me/campusdrive)

[}}](https://t.me/campusdrive)

[if(flag1 ==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[count++; printf(“%d,”, arr2[i]);](https://t.me/campusdrive)

[}}](https://t.me/campusdrive)

[return count;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int len1=3,len2=3, result, i, j; int arr1[10],arr2[10]; scanf(“%d %d”, &len1, &len2); for(i=0; i<len1; i++) scanf(“%d”, &arr1[i]);](https://t.me/campusdrive)

[for(i=0; i<len2; i++) scanf(“%d”, &arr2[i]);](https://t.me/campusdrive)

[result = Not\_common (arr1,arr2,len1,len2); printf(“\n %d”, result);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding questions – Important instructions](https://t.me/campusdrive)

## [Instructions](https://t.me/campusdrive)

1. [Only One question, 20 minutes.](https://t.me/campusdrive)
2. [Choice of C / C++ / Java / Perl / Python 2.7.](https://t.me/campusdrive)
3. [Provided an IDE to debug.](https://t.me/campusdrive)
4. [For Java, the class name should be named Maze.](https://t.me/campusdrive)
5. [Input to the program either through STDIN / Command line arguments, as per the instructions.](https://t.me/campusdrive)
6. [Program should write the output to STDOUT.](https://t.me/campusdrive)
7. [Public and private test cases based evaluation.](https://t.me/campusdrive)

## [Points to note](https://t.me/campusdrive)

1. [While printing the output no leading or trailing spaces should be printed.](https://t.me/campusdrive)
2. [Other than the required output, no other text should be printed.](https://t.me/campusdrive)
3. [If the output is a number, no leading sign must be printed unless it is a negative number.](https://t.me/campusdrive)
4. [No scientific notation (3.9265E + 2).](https://t.me/campusdrive)
5. [All floating point numbers must contain that many decimal places as mentioned in the question.](https://t.me/campusdrive)

# [TCS Ninja Mock test questions – Coding section](https://t.me/campusdrive)

[Consider the below series:](https://t.me/campusdrive)

[1, 2, 1, 3, 2, 5, 3, 7, 5, 11, 8, 13, 13, 17, …](https://t.me/campusdrive)

[This series is a mixture of 2 series – all the odd terms in this series form a Fibonacci series and all the even terms are the prime numbers in ascending order.](https://t.me/campusdrive)

[Write a program to find the Nth term in this series.](https://t.me/campusdrive)

[The value N is a Positive integer that should be read from STDIN. The Nth term that is calculated by the program should be written to STDOUT. Other than the value](https://t.me/campusdrive)

[of Nth term, no other characters/strings or message should be written to STDOUT.](https://t.me/campusdrive)

[For example, when N = 14, the 14th term in the series is 17. So only the value 17 should be printed to STDOUT.](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[#define MAX 1000 void fibonacci(int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, t1 = 0, t2 = 1, nextTerm; for (i = 1; i<=n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[nextTerm = t1 + t2; t1 = t2;](https://t.me/campusdrive)

[t2 = nextTerm;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, t1);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[void prime(int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, flag, count =0; for (i=2; i<=MAX; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag = 0;](https://t.me/campusdrive)

[for (j=2; j<i; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(i%j == 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag = 1; break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if (flag == 0) count++; if(count == n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”, i); break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main( )](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n; scanf(“%d”, &n); if(n%2 == 1)](https://t.me/campusdrive)

[fibonacci (n/2 + 1); else](https://t.me/campusdrive)

[prime(n/2); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 1:](https://t.me/campusdrive)

[Factorial program in c using command line arguments.](https://t.me/campusdrive)

[**Explanation:** Factorial of a non-negative integer n, denoted by n!, is the product of all positive integers less than or equal to n. For example, The value of 5! is 5\*4\*3\*2\*1](https://t.me/campusdrive)

[= 120](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[int main(int a, char \*b[]) //command line arguments](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int x,y,f=1;](https://t.me/campusdrive)

[x=atoi(b[1]); //atoi function is to convert a character to integer for(i=1;i<=x;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[f=f\*i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”,f); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 2:](https://t.me/campusdrive)

[Write a c program, to find the area of a circle when the diameter is given, using command line arguments. The input diameter is an integer and the output area should be a floating point variable with 2 point precision.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #define PI 3.14](https://t.me/campusdrive)

[int main(int a, char \*b[]) //command line arguments](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int d; float area =0; d= atoi(argv[1]);](https://t.me/campusdrive)

[area =(float) PI\*(d/2)\*(d/2);](https://t.me/campusdrive)

[printf(“%0.2f”, area); //%0.2f is to print the answer with 2 values after decimal point. return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 3:](https://t.me/campusdrive)

[Write a c program, to check whether the given year is a leap year or not using command line arguments. A leap year is a calendar year containing one additional day (Feb 29th) added to keep the calendar year synchronized with the astronomical year.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[int main(int a, char\*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int year; year=atoi(b[1]); if(year%100==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(year%400==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“LEAP YEAR”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else{](https://t.me/campusdrive)

[printf(“NOT LEAP YEAR”); } }](https://t.me/campusdrive)

[else if(year%4==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“LEAP YEAR”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else{](https://t.me/campusdrive)

[printf(“NOT LEAP YEAR”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0; }](https://t.me/campusdrive)

# [TCS Ninja Coding question 4:](https://t.me/campusdrive)

[Write a c program, to find the GCD of the given 2 numbers, using command line arguments. The input is 2 integer and the output GCD also should be an integer value.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[int main(int x, char \*y[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[inta,b,small,i; a=atoi(y[1]);](https://t.me/campusdrive)

[b=atoi(y[2]); small=a>b?b:a; for(i=small;i>=1;i–)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if((a%i==0)&&(b%i==0))](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,i); break;](https://t.me/campusdrive)

[} }](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 5:](https://t.me/campusdrive)

[C Program to check whether a given number is a prime number or not. The given number N, a positive integer, will be passed to the program using the first command line parameter. If it is a prime number the output should be the square root of the number up to 2 decimal point precision, If it is not a prime number then print 0.00 to stdout.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[#include int main(int a, char \*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int number,i,flag = 1; number = atoi(b[1]); for(i=2; i<number; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(number%i == 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[flag = 0; break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(flag == 1) printf(“%.2f”,sqrt(number)); else](https://t.me/campusdrive)

[printf(“0.00”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 6:](https://t.me/campusdrive)

[C Program to check whether a given number is a strong number or not. The given number N, a positive integer, will be passed to the program using the first command line parameter. If it is a strong number, the output should be “YES”, If it is not a prime](https://t.me/campusdrive)

[number then output should be “NO” to stdout. Other than YES or NO, no other extra information should be printed to stdout.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[int main(int a, char \*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int number, i, temp, sum = 0, factorial = 1; number = atoi(b[1]);](https://t.me/campusdrive)

[temp = number; while(number != 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int rem = number%10; for(i=2; i<=rem; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[factorial = factorial \* i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[sum = sum + factorial; number = number/10; factorial = 1;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(temp == sum) printf(“YES”); else printf(“NO”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 7:](https://t.me/campusdrive)

[Write a C program which will convert a given decimal integer number N to its binary equivalent. The given number N, a positive integer, will be passed to the program using the first command line parameter. Print the equivalent binary number to stdout. Other than the binary number, no other extra information should be printed to stdout Example: Given input “19”, here N=19, expected output 10011](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[int main(int a, char \*argv[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int number, count, i; int b[32];](https://t.me/campusdrive)

[number = atoi(argv[1]); count = 0;](https://t.me/campusdrive)

[while(number != 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[b[count]=number%2; number = number/2; count++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[for(i=(count-1); i>=0; i–)](https://t.me/campusdrive)

[printf(“%d”, b[i]); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 8:](https://t.me/campusdrive)

[Write a c program that will find the sum of all prime numbers in a given range. The range will be specified as command line parameters. The first command line parameter, N1 which is a positive integer, will contain the lower bound of the range. The second command line parameter N2, which is also a positive integer will contain the upper bound of the range. The program should consider all the prime numbers within the range, excluding the upper bound and lower bound. Print the output in integer format to stdout. Other than the integer number, no other extra information should be printed to stdout. Example Given inputs “7” and “24” here N1= 7 and N2=24, expected output as 83.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[int main(int argc, char \*argv[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int N1, N2, j, i, count, sum = 0; N1 =atoi(argv[1]);](https://t.me/campusdrive)

[N2 =atoi(argv[2]); for(i=N1+1; i<N2; ++i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[count = 0;](https://t.me/campusdrive)

[for(j=2; j<=(i/2); j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(i%j==0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[count++; break;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(count==0) sum = sum + i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”,sum); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 9:](https://t.me/campusdrive)

[Write a C program to check whether the given number is a perfect square or not using command line arguments.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[int main(int a, char \*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n, i;](https://t.me/campusdrive)

[n= atoi(b[1]);](https://t.me/campusdrive)

[for(i = 0; i <= n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(n == i \* i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“YES”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“NO”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 10:](https://t.me/campusdrive)

[Write a C program to check whether the given number is Palindrome or not using command line arguments.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[int main(int a,int \*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int number, rem, sum = 0; number = atoi(b[1]);](https://t.me/campusdrive)

[int copy = number; while(number != 0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[rem =number%10; sum = sum \* 10 + rem; number = number/10;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(copy == sum) printf(“Palindrome”); else](https://t.me/campusdrive)

[printf(“Not Palindrome”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 11:](https://t.me/campusdrive)

[Write a C program to convert the vowels to an uppercase in a given string using command line arguments.](https://t.me/campusdrive)

[Example: if the input is tata, then the expected output is tAtA.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include](https://t.me/campusdrive)

[int main(int argc, char \*argv[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[char \*str = argv[1]; int i;](https://t.me/campusdrive)

[for(i =0; str[i] !=’\0′; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(str[i] == ‘a’ || str[i] == ‘e’ || str[i] == ‘i’ || str[i] == ‘o’ || str[i] == ‘u’)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[str[i] = str[i] – 32;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%s”, str); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 12:](https://t.me/campusdrive)

[Write a C program to find the hypotenuse of a triangle using command line arguments.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include int main(int a, char\*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[float hyp;](https://t.me/campusdrive)

[int opp=atoi(b[1]); int adj=atoi(b[2]);](https://t.me/campusdrive)

[hyp=sqrt((opp\*opp)+(adj\*adj)); printf(“%0.2f”,hyp);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 13:](https://t.me/campusdrive)

[Write a C program to find whether the given number is an Armstrong number or not using command line arguments.](https://t.me/campusdrive)

[An Armstrong number of three digits is an integer such that the sum of the cubes of its digits is equal to the number itself. For example, 371 is an Armstrong number since 3\*\*3 + 7\*\*3 + 1\*\*3 = 371.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[#include int main(int a, char\*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n;](https://t.me/campusdrive)

[n= atoi(b[1]); int sum=0; int temp=n;](https://t.me/campusdrive)

[int cnt=0; while(n!=0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[n=n/10; cnt++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[n=temp; while(n!=0)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int rem=n%10; sum=sum+pow(rem,cnt); n=n/10;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(temp==sum)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“yes”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[else](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“no”);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [TCS Ninja Coding question 14:](https://t.me/campusdrive)

[Write a program to generate Fibonacci Series.](https://t.me/campusdrive)

## [Solution:](https://t.me/campusdrive)

[#include #include](https://t.me/campusdrive)

[int main(int a, char \*b[])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, n, t1 = 0, t2 = 1, nextTerm; n=atoi(b[1]);](https://t.me/campusdrive)

[for (i = 1; i <= n; ++i)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d “, t1); nextTerm = t1 + t2; t1 = t2;](https://t.me/campusdrive)

[t2 = nextTerm;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# [Cocubes Coding Questions with Answers](https://t.me/campusdrive)

1. [**Count the number of co-prime pairs in an array. (**Any two numbers whose GCD is 1 are be called as co-prime)](https://t.me/campusdrive)

## [Input:](https://t.me/campusdrive)

[The first line contains an integer T, total number of elements. Then follow T elements.](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[Count the number of co-prime pairs in an array.](https://t.me/campusdrive)

## [Constraints](https://t.me/campusdrive)

[1 ≤ T ≤ 25](https://t.me/campusdrive)

[1 ≤ elements ≤ 100](https://t.me/campusdrive)

## [Sample Input and Output:](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[1 2 3](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[Here, Co-prime pairs are (1, 2), (2, 3), (1, 3)](https://t.me/campusdrive)

## [Input:](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

[4 8 3 9](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

[Here, Co-prime pairs are (4, 3), (8, 3), (4, 9 ), (8, 9 )](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include<stdio.h>](https://t.me/campusdrive)

[int coprime(int a, int b)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int gcd;](https://t.me/campusdrive)

[while ( a != 0 )](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[gcd = a; a = b%a; b = gcd;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(gcd == 1)](https://t.me/campusdrive)

[return 1; else return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int count\_pairs(int arr[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int count = 0;](https://t.me/campusdrive)

[for (int i = 0; i < n – 1; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (int j = i + 1; j < n; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (coprime(arr[i], arr[j])) count++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return count;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n; scanf(“%d”, &n); int a[25], i; for(i=0; i<n; i++)](https://t.me/campusdrive)

[scanf(“%d”, &a[i]); printf(“%d”, count\_pairs(a, n)); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Search for Nth Occurrence](https://t.me/campusdrive)

[Given an array, number to search (say e1), and occurrence (say n), print the index of the nth occurrence of e1 in the array. If e1 does not occur n times, then print the index as -1.](https://t.me/campusdrive)

## [Input and Output:](https://t.me/campusdrive)

[Get the size of an array and get elements one by one. Input the number to be searched and occurrence. For example, 7 => Size of an array 1 4 6 7 6 3 6 => array elements 6 => number to be searched 3 => 3rd occurrence of number 6 Output: 6 Explanation: Number 6, 3rd occurrence position is 6](https://t.me/campusdrive)

## [Sample Input and Output:](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[7](https://t.me/campusdrive)

[1 4 6 7 6 3 6](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include<stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int a[100],n,i,e1,size,count=0; scanf(“%d”,&size); for(i=0;i<size;i++) scanf(“%d”,&a[i]);](https://t.me/campusdrive)

[scanf(“%d”,&e1);](https://t.me/campusdrive)

[scanf(“%d”,&n); for(i=0;i<size;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(e1==a[i]) count++;](https://t.me/campusdrive)

[//If ‘n’th occurrence found then print it’s index and exit. if(count==n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,i); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[//If ‘n’ occurrence not found then print ‘-1’. printf(“%d”,-1);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Search for an element in an array:](https://t.me/campusdrive)

[Program to search for an element in the given array.](https://t.me/campusdrive)

## [Input and Output:](https://t.me/campusdrive)

[The input consists of n + 2 lines. The first line consists a single integer n, The next n lines consists of 1 integer element part of the array. The last line consists of an integer to be searched. Output found or missing based on whether the element is present in the array or not. Note: max value of n is 100.](https://t.me/campusdrive)

## [Sample Input and Output: Input 1:](https://t.me/campusdrive)

[**3**](https://t.me/campusdrive)

[1 2 3](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

[**Output 1:** Missing](https://t.me/campusdrive)

## [Input 1:](https://t.me/campusdrive)

[**3**](https://t.me/campusdrive)

[1 2 3](https://t.me/campusdrive)

[2](https://t.me/campusdrive)

[**Output 2:** Found](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include <stdio.h> #define MAX\_SIZE 20 int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n, i, j, min\_index, array[MAX\_SIZE], x; scanf(“%d”, &n);](https://t.me/campusdrive)

[for(i = 0; i < n; i++) scanf(“%d”, &array[i]);](https://t.me/campusdrive)

[scanf(“%d”, &x); for(i = 0; i < n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(x == array[i])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“Found\n”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“Missing\n”);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Second largest number – Input:](https://t.me/campusdrive)

[The first line contains an integer T, total number of elements. Then follow T integers.](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[Display the second largest among the given T integers.](https://t.me/campusdrive)

## [Constraints](https://t.me/campusdrive)

[1 ≤ T ≤ 1000](https://t.me/campusdrive)

[1 ≤ integers ≤ 1000000](https://t.me/campusdrive)

## [Sample Input and Output: Input:](https://t.me/campusdrive)

[7](https://t.me/campusdrive)

[23 45 7 34 25 25 89](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[45](https://t.me/campusdrive)

[**Program:** #include<stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int a[50], size, i, j = 0, big, sec\_big; scanf(“%d”, &size);](https://t.me/campusdrive)

[for(i=0; i<size; i++) scanf(“%d”, &a[i]); big=a[0]; for(i=1;i<size;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(big<a[i])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[big=a[i]; j = i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[sec\_big =a[size-j-1]; for(i=1;i<size;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(sec\_big <a[i] && j != i) sec\_big =a[i];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, sec\_big); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Search index in a sorted array:](https://t.me/campusdrive)

[Program to find the target value in a two-dimensional matrix.](https://t.me/campusdrive)

## [Input and Output:](https://t.me/campusdrive)

[Get a target element and return its coordinates. If the value didn’t exist, the program had to return (-1,-1). First line of input is size of row and column,](https://t.me/campusdrive)

[followed rxc elements. The third line of input is the element to be searched in the rxc matrix.](https://t.me/campusdrive)

## [Sample Input and Output: Input 1:](https://t.me/campusdrive)

[4 2](https://t.me/campusdrive)

[0 9 8 7 6 5 4 3](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Output 1:](https://t.me/campusdrive)

[(3, 1)](https://t.me/campusdrive)

[**Program:** #include <stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, count = 0;](https://t.me/campusdrive)

[int arr[10][10], search, r, c;](https://t.me/campusdrive)

[scanf(“%d %d”, &r, &c); for (i = 0; i < r; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (j = 0; j < c; j++) scanf(“%d”, &arr[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[scanf(“%d”, &search);](https://t.me/campusdrive)

[for (i = 0; i < r; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (j = 0; j < c; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (arr[i][j] == search)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“(%d , %d)\n”, i, j); count++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if (count == 0) printf(“(-1,-1)”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

# cocubes coding questions with answers[Cocubes Coding Questions with Answers](https://t.me/campusdrive)

1. [**Count the number of co-prime pairs in an array. (**Any two numbers whose GCD is 1 are be called as co-prime)](https://t.me/campusdrive)

## [Input:](https://t.me/campusdrive)

[The first line contains an integer T, total number of elements. Then follow T elements.](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[Count the number of co-prime pairs in an array.](https://t.me/campusdrive)

## [Constraints](https://t.me/campusdrive)

[1 ≤ T ≤ 25](https://t.me/campusdrive)

[1 ≤ elements ≤ 100](https://t.me/campusdrive)

## [Sample Input and Output:](https://t.me/campusdrive)

[**Input:**](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[1 2 3](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

[Here, Co-prime pairs are (1, 2), (2, 3), (1, 3)](https://t.me/campusdrive)

## [Input:](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

[4 8 3 9](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[4](https://t.me/campusdrive)

[Here, Co-prime pairs are (4, 3), (8, 3), (4, 9 ), (8, 9 )](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include<stdio.h>](https://t.me/campusdrive)

[int coprime(int a, int b)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int gcd;](https://t.me/campusdrive)

[while ( a != 0 )](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[gcd = a; a = b%a; b = gcd;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if(gcd == 1)](https://t.me/campusdrive)

[return 1; else return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int count\_pairs(int arr[], int n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int count = 0;](https://t.me/campusdrive)

[for (int i = 0; i < n – 1; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (int j = i + 1; j < n; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (coprime(arr[i], arr[j])) count++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[return count;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n; scanf(“%d”, &n); int a[25], i; for(i=0; i<n; i++)](https://t.me/campusdrive)

[scanf(“%d”, &a[i]); printf(“%d”, count\_pairs(a, n));](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Search for Nth Occurrence](https://t.me/campusdrive)

[Given an array, number to search (say e1), and occurrence (say n), print the index of the nth occurrence of e1 in the array. If e1 does not occur n times, then print the index as -1.](https://t.me/campusdrive)

## [Input and Output:](https://t.me/campusdrive)

[Get the size of an array and get elements one by one. Input the number to be searched and occurrence. For example, 7 => Size of an array 1 4 6 7 6 3 6 => array elements 6 => number to be searched 3 => 3rd occurrence of number 6 Output: 6 Explanation: Number 6, 3rd occurrence position is 6](https://t.me/campusdrive)

## [Sample Input and Output: Input:](https://t.me/campusdrive)

[7](https://t.me/campusdrive)

[1 4 6 7 6 3 6](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

[**Program:** #include<stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int a[100],n,i,e1,size,count=0; scanf(“%d”,&size); for(i=0;i<size;i++) scanf(“%d”,&a[i]);](https://t.me/campusdrive)

[scanf(“%d”,&e1);](https://t.me/campusdrive)

[scanf(“%d”,&n); for(i=0;i<size;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(e1==a[i]) count++;](https://t.me/campusdrive)

[//If ‘n’th occurrence found then print it’s index and exit. if(count==n)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“%d”,i); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[//If ‘n’ occurrence not found then print ‘-1’. printf(“%d”,-1);](https://t.me/campusdrive)

[return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Search for an element in an array:](https://t.me/campusdrive)

[Program to search for an element in the given array.](https://t.me/campusdrive)

## [Input and Output:](https://t.me/campusdrive)

[The input consists of n + 2 lines. The first line consists a single integer n, The next n lines consists of 1 integer element part of the array. The last line consists of an integer to be searched. Output found or missing based on whether the element is present in the array or not. Note: max value of n is 100.](https://t.me/campusdrive)

## [Sample Input and Output: Input 1:](https://t.me/campusdrive)

[**3**](https://t.me/campusdrive)

[1 2 3](https://t.me/campusdrive)

[6](https://t.me/campusdrive)

[**Output 1:** Missing](https://t.me/campusdrive)

## [Input 1:](https://t.me/campusdrive)

[**3**](https://t.me/campusdrive)

[1 2 3](https://t.me/campusdrive)

[2](https://t.me/campusdrive)

[**Output 2:** Found](https://t.me/campusdrive)

## [Program:](https://t.me/campusdrive)

[#include <stdio.h> #define MAX\_SIZE 20 int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int n, i, j, min\_index, array[MAX\_SIZE], x; scanf(“%d”, &n);](https://t.me/campusdrive)

[for(i = 0; i < n; i++) scanf(“%d”, &array[i]);](https://t.me/campusdrive)

[scanf(“%d”, &x); for(i = 0; i < n; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(x == array[i])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“Found\n”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“Missing\n”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Second largest number – Input:](https://t.me/campusdrive)

[The first line contains an integer T, total number of elements. Then follow T integers.](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[Display the second largest among the given T integers.](https://t.me/campusdrive)

## [Constraints](https://t.me/campusdrive)

[1 ≤ T ≤ 1000](https://t.me/campusdrive)

[1 ≤ integers ≤ 1000000](https://t.me/campusdrive)

## [Sample Input and Output: Input:](https://t.me/campusdrive)

[7](https://t.me/campusdrive)

[23 45 7 34 25 25 89](https://t.me/campusdrive)

## [Output:](https://t.me/campusdrive)

[45](https://t.me/campusdrive)

[**Program:** #include<stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int a[50], size, i, j = 0, big, sec\_big;](https://t.me/campusdrive)

[scanf(“%d”, &size); for(i=0; i<size; i++) scanf(“%d”, &a[i]); big=a[0]; for(i=1;i<size;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(big<a[i])](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[big=a[i]; j = i;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[sec\_big =a[size-j-1]; for(i=1;i<size;i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if(sec\_big <a[i] && j != i) sec\_big =a[i];](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[printf(“%d”, sec\_big); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

## [Search index in a sorted array:](https://t.me/campusdrive)

[Program to find the target value in a two-dimensional matrix.](https://t.me/campusdrive)

## [Input and Output:](https://t.me/campusdrive)

[Get a target element and return its coordinates. If the value didn’t exist, the program had to return (-1,-1). First line of input is size of row and column,](https://t.me/campusdrive)

[followed rxc elements. The third line of input is the element to be searched in the rxc matrix.](https://t.me/campusdrive)

## [Sample Input and Output: Input 1:](https://t.me/campusdrive)

[4 2](https://t.me/campusdrive)

[0 9 8 7 6 5 4 3](https://t.me/campusdrive)

[3](https://t.me/campusdrive)

## [Output 1:](https://t.me/campusdrive)

[(3, 1)](https://t.me/campusdrive)

[**Program:** #include <stdio.h> int main()](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[int i, j, count = 0;](https://t.me/campusdrive)

[int arr[10][10], search, r, c;](https://t.me/campusdrive)

[scanf(“%d %d”, &r, &c); for (i = 0; i < r; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (j = 0; j < c; j++) scanf(“%d”, &arr[i][j]);](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[scanf(“%d”, &search); for (i = 0; i < r; i++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[for (j = 0; j < c; j++)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[if (arr[i][j] == search)](https://t.me/campusdrive)

[{](https://t.me/campusdrive)

[printf(“(%d , %d)\n”, i, j); count++;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[}](https://t.me/campusdrive)

[if (count == 0) printf(“(-1,-1)”); return 0;](https://t.me/campusdrive)

[}](https://t.me/campusdrive)