**C Programming Topic Wise**

**Questions**

By

Kitflix.com

Table of Contents

1. [Introduction to C Programming 1](#_Toc72692884)
2. [Variables in C 3](#_Toc72692885)
3. [Conditional Execution in C (if else/switch) 4](#_Toc72692886)
4. [Loops in C 6](#_Toc72692887)
5. [Arrays in C 8](#_Toc72692888)
6. [Arrays in C (string / text array) 15](#_Toc72692889)
7. [Pointers in C 21](#_Toc72692890)
8. [Functions in C 24](#_Toc72692891)

# Introduction to C Programming

1. Write a C program to print a string without quote in the program.

Hint: Think out of the box

1. Write a C program to take a one-line text as input in string s from user and print Hello, World! followed by input string provided by user.

**Example  
s=”** **Life is beautiful** “

The required output is:

Hello, World!

Life is beautiful

**Input:**

Welcome to C programming.

**Output:**

Hello, World!

Welcome to C programming.

1. Write a C program to swap two numbers without using the third variable.

**Input**:

Enter two numbers:

x=50

y=60

**Output**:

x=60

y=50

1. Write a C program to add to number without using “+” operator.

**Input:**

Enter two numbers:

20

40

**Output:** 60

1. Write a C program find the maximum & minimum of two numbers in a single line without using any condition & loop.

**Input:**

30

15

**Output:**

max=30, min=15

1. Write a C Program to Check Whether a Number is Palindrome or Not

**Input**:

12121

**Output**:

True

**Input**:

12121

**Output**:

False

1. Write a C program to find the parity of an unsigned integer. Parity of a number refers to whether it contains an odd or even number of 1-bits. The number has “odd parity”, if it contains odd number of 1-bits and is “even parity” if it contains even number of 1-bits.

**Input:**

n =7

**Output:**

Parity= 1

Variables in C

Concept**:** Basic understanding

1. Write a C Program to illustrate the following global variables and local variables
2. Write a C program to convert a string to an unsigned long integer.

**Input**: 25

**Output:** 25

1. Write a C program to convert float number to string.

In program it should print with %s

**Input**: 23.34

**Output:**

The string for the num is 23.34

# Conditional Execution in C (if else/switch)

1. **Write a C program to print “Hello World” without using semicolon even once?**

Hint: Think out of the box

**Input**

Input alphabet: c

**Output**

'c' is consonant

1. Write a C program to input an alphabet and check whether it is vowel or consonant using switch case.

**Input:**

Enter a number: 1996

**Output:**

mmxii

1. **Write a C program to convert integer to roman.**
2. Given an integer N and a pizza which can be cut into pieces, each cut should be a straight line going from the center of the pizza to its border. Also, the angle between any two cuts must be a positive integer. Two pieces are equal if their appropriate angles are equal.

The given pizza can be cut in following three ways:

* Cut the pizza into **N equal pieces**.
* Cut the pizza into **N pieces of any size**.
* Cut the pizza into **N pieces** such that **no two of them are equal**.

The task is to find if it is possible to cut the pizza in the above ways for a given value of **N**. Print **1** if **possible** else **0** for all the cases i.e. print 111 if all the cases are possible.

**Input:** N = 4  
**Output:** 1 1 1  
Case 1: All four pieces can have angle = 90  
Case 2: Same cut as Case 1  
Case 3: 1, 2, 3 and 354 are the respective angles of the four pieces cut.

**Input:** N = 7  
**Output:** 0 1 1

# Loops in C

1. In the below code, change/add only one character and print ‘\*’ exactly 20 times.   
    int main (){

int i, n = 20;

for (i = 0; i < n; i--)

printf ("\*");

getchar ();

return 0;

}

1. Write a C program to get the sum of digit of given number in single statement line.

**Input :** n = 687

**Output:** 21

**Input :** n = 12

**Output**: 3

1. Write a C program to Count number of bits to be flipped to convert A to B

**Input** : a = 10, b = 20

**Output** : 4

Binary representation of a is 000**0101**0

Binary representation of b is 00010100

We need to flip highlighted four bits in a

to make it b.

**Input**: a = 7, b = 10

**Outpu**t : 3

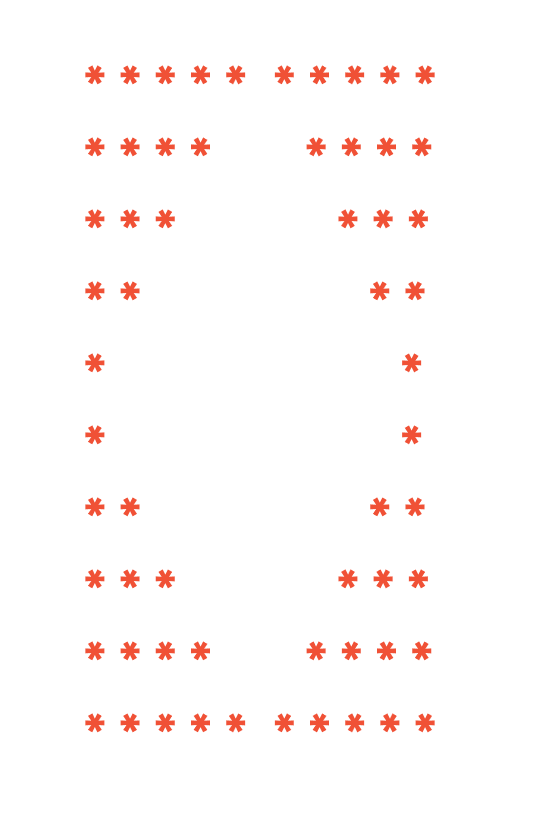
Binary representation of a is 0000**01**1**1**

Binary representation of b is 00001010

We need to flip highlighted three bits in a

to make it b.

1. **Write a C program to create below pattern**



Hint: Think out of the box

# Arrays in C

1. Given an array nums with n objects colored red, white, or blue, sort them [in-place](https://en.wikipedia.org/wiki/In-place_algorithm)so that objects of the same color are adjacent, with the colors in the order red, white, and blue.

We will use the integers 0, 1, and 2 to represent the color red, white, and blue, respectively.

**Example 1:**

**Input:** nums = [2,0,2,1,1,0]

**Output:** [0,0,1,1,2,2]

**Example 2:**

**Input:** nums = [2,0,1]

**Output:** [0,1,2]

**Example 3:**

**Input:** nums = [0]

**Output:** [0]

**Example 4:**

**Input:** nums = [1]

**Output:** [1]

1. **Write a C program to find the maximum amount of water that C program to find maximum amount of water that can be trapped within given set of bars.** block height are given as input in an array and assuming each block of size 1m.

**Input 0**:

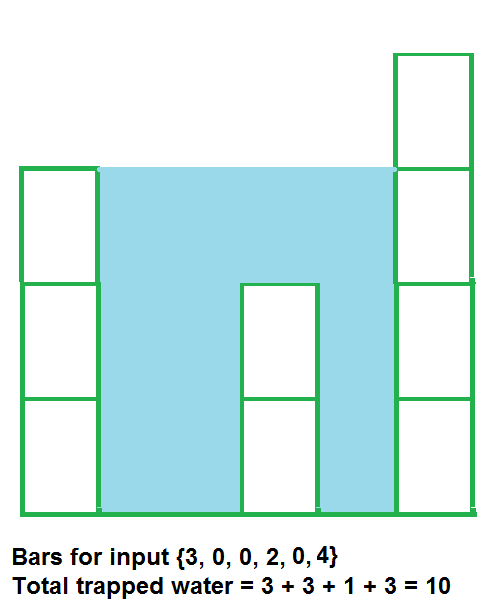
N = 4

arr [] = {7,4,0,9}

**Output 0**:

10

**Explanation:**



Water trapped by above

block of height 4 is 3 units and above

block of height 0 is 7 units. So, the

total unit of water trapped is 10 units.

**Input 1**:

N = 3

arr [] = {6,9,9}

**Output 1**:

0

**Explanation:** No water trapped

1. Given a square matrix of size N\*N, return an array of its anti-diagonals. For better understanding let us look at the image given below

**Input:**



**Output:**

1

2 4

3 5 7

6 9 8

9

1. Find the count of the number of ways to reach from source to destination in a given a maze with obstacles, count number of paths to reach rightmost-bottommost cell from topmost-leftmost cell. A cell in given maze has value -1 if it is a blockage or dead end, else 0.  
   From a given cell, we are allowed to move to cells (i+1, j) and (i, j+1) only.

**Input:** maze[R][C] = {{0, 0, 0, 0},

{0, -1, 0, 0},

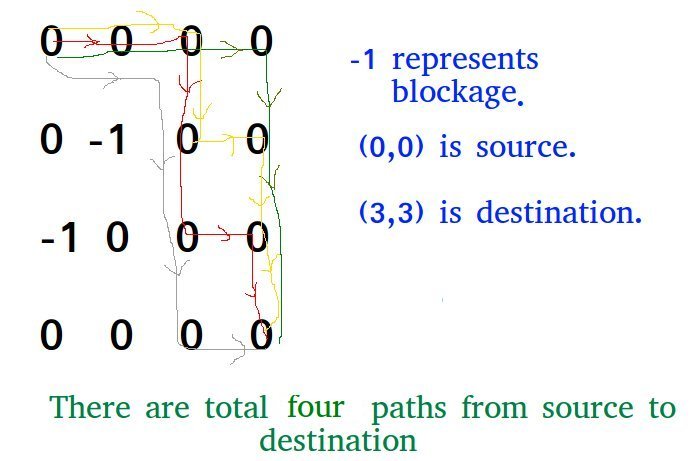
{-1, 0, 0, 0},

{0, 0, 0, 0}};

**Output:** 4

There are four possible paths as shown in

below diagram.



1. Given a square matrix, turn it by 90 degrees in anti-clockwise direction without using any extra space.

**Input:**

1 2 3

4 5 6

7 8 9

**Output:**

3 6 9

2 5 8

1 4 7

**Rotated the input matrix by**

**90 degrees in anti-clockwise direction**.

1. Write a C program to implement iterative Binary Search. A iterative binary search function. It returns location of x in given array arr[l..r] if present, otherwise -1.
2. Write a C program that, given an array A[] of n numbers and another number x, determines whether or not there exist two elements in S whose sum is exactly x.

**Enter number of elements**:

5

**Enter 5 integers:**

1  
9  
22  
24  
46

**Input:**

**Enter the value to find:**

24

**Output:4**

24 is present at index 4.

**Input:**

**Enter the value to find:**

4

**Output: -1**

4 is not present

**Input:** arr[] = {0, -1, 2, -3, 1}

sum = -2

**Output:** -3, 1

If we calculate the sum of the output,

1 + (-3) = -2

**Input:** arr[] = {1, -2, 1, 0, 5}

sum = 0

**Output:** -1

No valid pair exists.

# Arrays in C (string / text array)

1. **Write a C program to reverse a String in C**

**Input:** ANU

**Output**: UNA

1. **Write a C program to implement following string library operations on given string s1, s2 and character ch accordingly.**
2. strlen
3. strcmp
4. strcpy
5. strcat
6. strstr
7. strchr
8. **Write a C program** find the first occurrence of a character in a given **string with the help of library function.**

**Input**: str[] = 'This is a string' , ch =’a’

**Outpu**t : 9

**Input** : str[] = 'My name is Anu', ch =’a’

**Outpu**t : 5

1. **Write a C** program to check if strings are rotations of each other or not. Given a string s1 and a string s2, write a snippet to say whether s2 is a rotation of s1?

**Input:** str1 =ABCD, str2= CDAB

**Output**: True

**Explanation:** ABCD and CDAB are rotation of each other

**Input**: str1 =ABCD, str2= ACBD

**Output**: False

1. **Write a C** program to check if strings are anagrams of each other or not. Given a string s1 and a string s2, write a snippet to say whether s2 is an anagrams of s1?An anagram of a string is another string that contains the same characters, only the order of characters can be different. For example, “abcd” and “dabc” are an anagram of each other.

**Input:** str1 =LISTEN, str2= SILENT

**Output**: True

**Input**: str1 =abcd, str2= pqrs

**Output**: False

1. Write a C Program to Sort set of strings in alphabetical order string taken as input from user**.**

**Input:** 4

Nanu

Manu

Tanu

Anu

**Output:**

Anu

Manu

Nanu

Tanu

1. **Write a C program** given a string containing lowercase characters. The task is to print the maximum occurring character in the input string. If 2 or more characters appear the same number of times, print the lexicographically (alphabetically) lowest (first) character

**Input:** test sample

**Output**: e

‘t’, ‘e’ and ‘s’ appears 2 times, but ‘e’ is the lexicographically smallest character.

**Input**: sample program

**Output**: a

1. Given a string s we need to tell minimum characters to be appended (insertion at end) to make a string palindrome.

**Input:** s = "abede"

**Output**: 2

We can make string palindrome as "abede**ba**"

by adding **ba** at the end of the string.

**Input:** s = "aabb"

**Output**: 2

We can make string palindrome as "aabb**aa**"

by adding **aa** at the end of the string.

1. Given a string s input it from user, write a program to find the length of the longest substring

**Example 1:**

Input: s = "abcabcbb"

Output: 3

Explanation: The answer is "abc", with the length of 3.

**Example 2:**

Input: s = "bbbbb"

Output: 1

Explanation: The answer is "b", with the length of 1.

**Example 3:**

Input: s = "pwwkew"

Output: 3

Explanation: The answer is "wke", with the length of 3.

Notice that the answer must be a substring, "pwke" is a subsequence and not a substring.

**Example 4:**

Input: s = ""

Output: 0 without repeating characters

1. Write code to convert a given 4-digit number into words.

**Input:** 1234

**Output:** one thousand two hundred thirty four

1. Write a program to Validate an IPv4 Address. [IPv4 addresses](http://en.wikipedia.org/wiki/IP_address)are canonically represented in dot-decimal notation, which consists of four decimal numbers, each ranging from 0 to 255, separated by dots, e.g., 172.16.254.1

**Input:** 128.0.0.1

**Output:** valid

**Explanation:**

This is a valid IP address.

**Input:** "125.512.100.abc"

**Output:** Invalid

**Explanation:**

invalid IP address with this string.

1. Write a program to generate random password, which fulfills all the criteria for strong password randomly. (given n as length of password)

**Input:** 4

**Output:** $7fK

Criteria for strong password is as follows:   
A password is strong if it has:   
2. At least one special char   
3. At least one number   
4. At least one upper and one lower case char. 

1. Write a C program to find the occurrence of the word and replace a word in a text by another given word.

Given three strings ‘str’, ‘oldW’ and ‘newW’. The task is find all occurrences of the word ‘oldW’ and replace them with word ‘newW’.  
**Input**: str [] = " All is well",

oldW [] = "is",

newW [] = "izzz"

**Output:** All izzz well

1. Given two strings where first string may contain wild card characters and second string is a normal string. Write a function that returns true if the two strings match. The following are allowed wild card characters in first string.

\* --> Matches with 0 or more instances of any character or set of characters.

? --> Matches with any one character.

**Example 1:**

**Input:** s = "aa", p = "a"

**Output:** false

**Explanation:** "a" does not match the entire string "aa".

**Example 2:**

**Input:** s = "aa", p = "\*"

**Output**: true

**Explanation:**'\*' matches any sequence.

**Example 3:**

**Input**: s = "cb", p = “? a"

**Output:** false

**Explanation**: '?' matches 'c', but the second letter is 'a', which does not match 'b'.

**Example 4:**

**Input**: s = "adceb", p = "\*a\*b"

**Output**: true

**Explanation**: The first '\*' matches the empty sequence, while the second '\*' matches the substring "dce".

**Example 5:**

**Input**: s = "acdcb", p = "a\*c? b"

**Output**: false

# Pointers in C

1. Write a program in C to store n elements in an array and print the elements using pointer

Input the number of elements to store in the array :5  
Input 5 number of elements in the array :  
element - 0 : 5  
element - 1 : 7  
element - 2 : 2  
element - 3 : 9  
element - 4 : 8  
**Expected Output** :

The elements you entered are :

element - 0 : 5

element - 1 : 7

element - 2 : 2

element - 3 : 9

element - 4 : 8

1. **Write a C program to implement strstr using pointers (take two string s1 and s2, if first string s1 is substring of second string s2 then return index number else it will return -1 if string is not matching)**

**Input:** s1 = "for", s2 = "formula"

**Output:** 5

**Explanation:**

String "for" is present as a substring

of s2.

**Input:** s1 = "practice", s2 = "learn"

**Output:** -1.

**Explanation:**

There is no occurrence of "practice" in

"learn"

1. **Write a C program to compare two string using pointers (take two string s1 and s2 print equal if both the strings are equal else print not equal)**

**Input:** s1 = "for", s2 = "formula"

**Output:** not equal

**Explanation:**

Strings are not equal

**Input:** s1 = "learn", s2 = "learn"

**Output:** equal

**Explanation:**

Strings are equal

1. **Write a C program to** Count of number of given string in 2D character array

Given a 2-Dimensional character array and a string, we need to find the given string in 2-dimensional character array such that individual characters can be present left to right, right to left, top to down or down to top.

**Input** : a = {

{B,B,M,B,B,B},

{C,B,A,B,B,B},

{I,B,G,B,B,B},

{G,B,I,B,B,B},

{A,B,C,B,B,B},

{M,C,I,G,A,M}

}

str= "MAGIC"

**Output :**4

1. Write an efficient C function that takes two strings as arguments and removes the characters from first string which are present in second string (mask string) using pointers.

**Input:** s1 = "for", s2 = "formula"

**Output:** mula

**Explanation:**

Removed for from formula

1. Write a C program to group the anagrams together in given an array of strings strs. You can return the answer in any order.

**Example 1:**

**Input:** strs = ["eat","tea","tan","ate","nat","bat"]

**Output:** [["bat"],["nat","tan"],["ate","eat","tea"]]

**Example 2:**

**Input:** strs = [""]

**Output:** [[""]]

**Example 3:**

**Input:** strs = ["a"]

**Output:** [["a"]]

# Functions in C

**Input: 1.67**

**Output: 2**

1. Write a one-line C function to round floating point numbers
2. Write a one-line C function to return position of first 1 from right to left, in binary representation of an Integer.

**Input:**18, Binary Representation 010010

**Output**:2

**Input:**19, Binary Representation 010011

**Output:**1

1. Write a C function to find count of square in given a m x n rectangle.

**Input:** m = 2, n = 2

Output: 5

There are 4 squares of size 1x1 + 1 square of size 2x2.

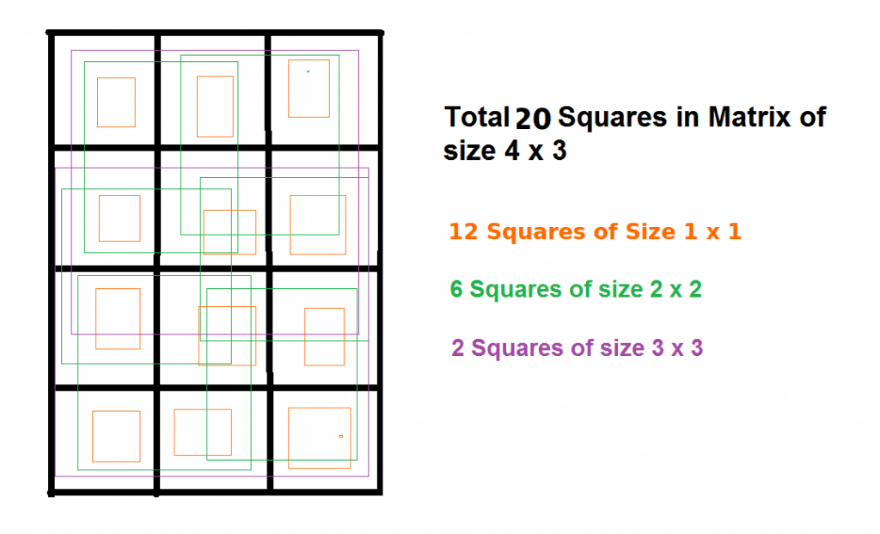
**Input:** m = 4, n = 3

**Output:** 20

There are 12 squares of size 1x1 +

6 squares of size 2x2 +

2 squares of size 3x3



1. **Write a C program to print numbers from 1 to 100 without using a loop**

Hint: Think out of the box

1. Write a C program for a recursive function for Tower of Hanoi

void towerOfHanoi (int n, char from\_rod, char to\_rod, char aux\_rod)

where n will be the number of disc, given as input

and A, B, C are the rods

void towerOfHanoi (int n, A, B, C)

simple rules:

* Only one disk can be moved at a time.
* Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
* No disk may be placed on top of a smaller disk



**Input 0: 2**

**Output 0:**

Disk 1 moved from A to B

Disk 2 moved from A to C

Disk 1 moved from B to C

**Input 1: 3**

**Output 1:**

Disk 1 moved from A to C

Disk 2 moved from A to B

Disk 1 moved from C to B

Disk 3 moved from A to C

Disk 1 moved from B to A

Disk 2 moved from B to C

Disk 1 moved from A to C

1. **Write a C program to print** Count of strings that can be formed using a, b and c under given constraints. Given a length n, count the number of strings of length n that can be made using ‘a’, ‘b’ and ‘c’ with at-most one ‘b’ and two ‘c’s allowed using recursive function.

**Input:** n = 3

**Output**: 19

Below strings follow given constraints:

aaa aab aac aba abc aca acb acc baa

bac bca bcc caa cab cac cba cbc cca ccb

**Input**: n = 4

**Output**: 39

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*END\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*