**C Data Structure**

Sponsored Links

* **[Stack Data Structure](http://cprogramminglanguage.net/c-stack-data-structure.aspx" \o "Stack Data Structure)**

Here you will learn what stack data structure is, how to implement a stack using C array, linked representation and applications of stack.

* **[Queue](http://cprogramminglanguage.net/queue-data-structure.aspx" \o "Queue)**

Provides the definition of queue data structure - one of the most common uses data structure in computer world. It also gives varies implementations of queue using array, link with source code.

* **[Linked List](http://cprogramminglanguage.net/linked-list-data-structure-c-source-code.aspx" \o "Linked List)**

Provides all common linked list data structures such as: singly-linked list, doubly-linked list,singly-circularly-linked list and Doubly-circularly-linked list with clear explanation and source code in C.

* **[AVL Tree](http://cprogramminglanguage.net/avl-tree.aspx" \o "AVL Tree)**

An AVL tree is a self-balancing binary search tree, and it is the first such data structure to be invented.

* **[Binary Search Tree](http://cprogramminglanguage.net/binary-search-tree-c-code.aspx" \o "Binary Search Tree)**

Binary search tree (BST) is a node based binary tree data structure.

* **[Binary Heap](http://cprogramminglanguage.net/binary-heap-c-code.aspx" \o "Binary Heap)**

A binary heap is a heap data structure created using a binary tree.

**Source code of simple Selection sort implementation using array ascending order in c programming language**

|  |  |
| --- | --- |
| #include <stdio.h> | |
|  | #include <stdlib.h> | |

|  |  |
| --- | --- |
| 03 |  |
| 04 | void swap(int \*x,int \*y) | |

|  |  |
| --- | --- |
| 05 | { |
| 06 | int temp; | |

|  |  |  |
| --- | --- | --- |
| 07 | temp = \*x; | |
| 08 | \*x = \*y; |

|  |  |  |
| --- | --- | --- |
| 09 | \*y = temp; | |
| 10 | } |

|  |  |
| --- | --- |
| 11 |  |
| 12 | void selection\_sort(int list[], int n) | |

|  |  |
| --- | --- |
| 13 | { |
| 14 | int i, j, min; | |

|  |  |
| --- | --- |
| 15 |  |
| 16 | for (i = 0; i < n - 1; i++) | |

|  |  |
| --- | --- |
| 17 | { |
| 18 | min = i; | |

|  |  |  |
| --- | --- | --- |
| 19 | for (j = i+1; j < n; j++) | |
| 20 | { |

|  |  |  |
| --- | --- | --- |
| 21 | if (list[j] < list[min]) | |
| 22 | { |

|  |  |  |
| --- | --- | --- |
| 23 | min = j; | |
| 24 | } |

|  |  |
| --- | --- |
| 25 | } |
| 26 | swap(&list[i], &list[min]); | |

|  |  |  |
| --- | --- | --- |
| 27 | } | |
| 28 | } |

|  |  |
| --- | --- |
| 29 |  |
| 30 | void printlist(int list[],int n) | |

|  |  |
| --- | --- |
| 31 | { |
| 32 | int i; | |

|  |  |
| --- | --- |
| 33 | for(i=0;i<n;i++) |
| 34 | printf("%d\t",list[i]); | |

|  |  |  |
| --- | --- | --- |
| 35 | } | |
| 36 |  |

|  |  |  |
| --- | --- | --- |
| 37 | void main() | |
| 38 | { |

|  |  |  |
| --- | --- | --- |
| 39 | const int MAX\_ELEMENTS = 10; | |
| 40 | int list[MAX\_ELEMENTS]; |

|  |  |
| --- | --- |
| 41 |  |
| 42 | int i = 0; | |

|  |  |
| --- | --- |
| 43 |  |
| 44 | // generate random numbers and fill them to the list | |

|  |  |  |
| --- | --- | --- |
| 45 | for(i = 0; i < MAX\_ELEMENTS; i++ ){ | |
| 46 | list[i] = rand(); |

|  |  |
| --- | --- |
| 47 | } |
| 48 | printf("The list before sorting is:\n"); | |

|  |  |  |
| --- | --- | --- |
| 49 | printlist(list,MAX\_ELEMENTS); | |
| 50 |  |

|  |  |
| --- | --- |
| 51 | // sort the list |
| 52 | selection\_sort(list,MAX\_ELEMENTS); | |

|  |  |
| --- | --- |
| 53 |  |
| 54 | // print the result | |

|  |  |  |
| --- | --- | --- |
| 55 | printf("The list after sorting:\n"); | |
| 56 | printlist(list,MAX\_ELEMENTS); |

|  |  |
| --- | --- |
| 57 | } |

////////////////////////////////////////////////////////

**Source code of simple insertion sort implementation using array in ascending order in c programming language**

#include<stdio.h>

int main(){

  int i,j,s,temp,a[20];

  printf("Enter total elements: ");

  scanf("%d",&s);

  printf("Enter %d elements: ",s);

  for(i=0;i<s;i++)

      scanf("%d",&a[i]);

  for(i=1;i<s;i++){

      temp=a[i];

      j=i-1;

      while((temp<a[j])&&(j>=0)){

      a[j+1]=a[j];

          j=j-1;

      }

      a[j+1]=temp;

  }

  printf("After sorting: ");

  for(i=0;i<s;i++)

      printf(" %d",a[i]);

  return 0;

}

Output:

Enter total elements: 5

Enter 5 elements: 3 7 9 0 2

After sorting:  0 2 3 7 9

/////////////////////////////////////////////////////////////////////

**Source code of simple quick sort implementation using array ascending order in c programming language**

#include<stdio.h>

void quicksort(int [10],int,int);

int main(){

  int x[20],size,i;

  printf("Enter size of the array: ");

  scanf("%d",&size);

  printf("Enter %d elements: ",size);

  for(i=0;i<size;i++)

    scanf("%d",&x[i]);

  quicksort(x,0,size-1);

  printf("Sorted elements: ");

  for(i=0;i<size;i++)

    printf(" %d",x[i]);

  return 0;

}

void quicksort(int x[10],int first,int last){

    int pivot,j,temp,i;

     if(first<last){

         pivot=first;

         i=first;

         j=last;

         while(i<j){

             while(x[i]<=x[pivot]&&i<last)

                 i++;

             while(x[j]>x[pivot])

                 j--;

             if(i<j){

                 temp=x[i];

                  x[i]=x[j];

                  x[j]=temp;

             }

         }

         temp=x[pivot];

         x[pivot]=x[j];

         x[j]=temp;

         quicksort(x,first,j-1);

         quicksort(x,j+1,last);

    }

}

Output:

Enter size of the array: 5

Enter 5 elements: 3 8 0 1 2

Sorted elements: 0 1 2 3 8

**Source code of simple merge sort implementation using array in ascending order in c programming language**

#include<stdio.h>

#define MAX 50

void mergeSort(int arr[],int low,int mid,int high);

void partition(int arr[],int low,int high);

int main(){

    int merge[MAX],i,n;

    printf("Enter the total number of elements: ");

    scanf("%d",&n);

    printf("Enter the elements which to be sort: ");

    for(i=0;i<n;i++){

         scanf("%d",&merge[i]);

    }

    partition(merge,0,n-1);

    printf("After merge sorting elements are: ");

    for(i=0;i<n;i++){

         printf("%d ",merge[i]);

    }

   return 0;

}

void partition(int arr[],int low,int high){

    int mid;

    if(low<high){

         mid=(low+high)/2;

         partition(arr,low,mid);

         partition(arr,mid+1,high);

         mergeSort(arr,low,mid,high);

    }

}

void mergeSort(int arr[],int low,int mid,int high){

    int i,m,k,l,temp[MAX];

    l=low;

    i=low;

    m=mid+1;

    while((l<=mid)&&(m<=high)){

         if(arr[l]<=arr[m]){

             temp[i]=arr[l];

             l++;

         }

         else{

             temp[i]=arr[m];

             m++;

         }

         i++;

    }

    if(l>mid){

         for(k=m;k<=high;k++){

             temp[i]=arr[k];

             i++;

         }

    }

    else{

         for(k=l;k<=mid;k++){

             temp[i]=arr[k];

             i++;

         }

    }

    for(k=low;k<=high;k++){

         arr[k]=temp[k];

    }

}

**Sample output:**

Enter the total number of elements: 5

Enter the elements which to be sort: 2 5 0 9 1

After merge sorting elements are: 0 1 2 5 9

### Find g.c.d of two number using c program

**Definition of HCF (Highest common factor):**

HFC is also called greatest common divisor (gcd). HCF of two numbers is a largest positive numbers which can divide both numbers without any remainder.  For example HCF of two numbers 4 and 8 is 2 since 2 is the largest positive number which can dived 4 as well as 8 without a remainder.

**Logic of HCF or GCD of any two numbers:**

In HCF we try to find any largest number which can divide both the number.

For example: HCF or GCD of 20 and 30

Both number 20 and 30 are divisible by 1, 2,5,10.

HCF=max (1, 2, 3, 4, 10) =10

**Logic for writing program:**

It is clear that any number is not divisible by greater than number itself. In case of more than one numbers, a possible maximum number which can divide all of the numbers must be minimum of all of that numbers.

For example: 10, 20, and 30

Min (10, 20, 30) =10 can divide all there numbers. So we will take one for loop which will start form min of the numbers and will stop the loop when it became one, since all numbers are divisible by one. Inside for loop we will write one if conditions which will check divisibility of both the numbers.

Program:

**#include**<stdio.h>

**int** **main**(){

**int** x,y,m,i;

    printf("Insert any two number: ");

    scanf("%d%d",&x,&y);

**if**(x>y)

         m=y;

**else**

         m=x;

**for**(i=m;i>=1;i--){

**if**(x%i==0&&y%i==0){

             printf("\nHCF of two number is : %d",i) ;

**break**;

         }

    }

**return** 0;

}

**Other logic : HCF  program with two numbers in c**

#include<stdio.h>

int main(){

int n1,n2;

printf("\nEnter two numbers:");

scanf("%d %d",&n1,&n2);

while(n1!=n2){

if(n1>=n2-1)

n1=n1-n2;

else

n2=n2-n1;

}

printf("\nGCD=%d",n1);

return 0;

}

**HCF  program with multiple numbers in c**

**#include**<stdio.h>

**int** **main**(){

**int** x,y=-1;

    printf("Insert numbers. To exit insert zero: ");

**while**(1){

         scanf("%d",&x);

**if**(x<1)

**break**;

**else** **if**(y==-1)

             y=x;

**else** **if** (x<y)

             y=gcd(x,y);

**else**

             y=gcd(y,x);

    }

    printf("GCD is %d",y);

**return** 0;

}

**int** **gcd**(**int** x,**int** y){

**int** i;

**for**(i=x;i>=1;i--){

**if**(x%i==0&&y%i==0){

**break**;

         }

    }

**return** i;

}

**LCM program in c with two numbers :**

#include<stdio.h>

int main(){

  int n1,n2,x,y;

  printf("\nEnter two numbers:");

  scanf("%d %d",&n1,&n2);

  x=n1,y=n2;

  while(n1!=n2){

      if(n1>n2)

           n1=n1-n2;

      else

      n2=n2-n1;

  }

  printf("L.C.M=%d",x\*y/n1);

  return 0;

}

**LCM program in c with two numbers (Other logic) :**

#include<stdio.h>

int lcm(int,int);

int main(){

    int a,b,l;

    printf("Enter any two positive integers ");

    scanf("%d%d",&a,&b);

    if(a>b)

         l = lcm(a,b);

    else

         l = lcm(b,a);

    printf("LCM of two integers is %d",l);

    return 0;

}

int lcm(int a,int b){

    int temp = a;

    while(1){

         if(temp % b == 0 && temp % a == 0)

             break;

         temp++;

    }

   return temp;

}

**LCM program in c with multiple numbers :**

#include<stdio.h>

int lcm(int,int);

int main(){

    int a,b=1;

    printf("Enter positive integers. To quit press zero.");

    while(1){

         scanf("%d",&a);

         if(a<1)

             break;

         else if(a>b)

             b = lcm(a,b);

         else

             b = lcm(b,a);

    }

    printf("LCM is %d",b);

    return 0;

}

int lcm(int a,int b){

    int temp = a;

    while(1){

if(temp % b == 0 && temp % a == 0)

  break;

         temp++;

    }

    return temp;

}

**Definition of LCM (Least common multiple):**

LCM of two integers is a smallest positive integer which is multiple of both integers that it is divisible by the both of the numbers.

For example: LCM of two integers 2 and 5 is 10 since 10 is the smallest positive numbers which is divisible by both 2 and 5.

### BINARY SEARCH USING C PROGRAM

1. **Write a simple code for binary search in c programming language**

2. **Wap a c program to search an element in an array using binary search**

This method is very fast and efficient. This method requires that the list of the elements be in sorted order. In this method to search an element we compare it with the element present at the center of the list. If it matches then the search is successful. Otherwise the list is divided into two halves. One from 0th element to the center element (first half) and another from center element to last element(second half).As a result all the element in the first half are smaller than the center element whereas all the element in the second half are greater than the center element.

[Reply With Quote](http://forums.techarena.in/newreply.php?do=newreply&p=5161925)

void main( )

{

int arr[10] = { 1, 2, 3, 9, 11, 13, 17, 25, 57, 90 } ;

int mid, lower = 0 , upper = 9, num, flag = 1 ;

clrscr( ) ;

printf ( "Enter number to search: " ) ;

scanf ( "%d", &num ) ;

for ( mid = ( lower + upper ) / 2 ; lower <= upper ;mid = ( lower +

upper ) / 2 )

{

if ( arr[mid] == num )

{

printf ( "The number is at position %d in the array.", mid ) ;

flag = 0 ;

break ;

}

if ( arr[mid] > num )

upper = mid - 1 ;

else

lower = mid + 1 ;

}

if ( flag )

printf ( "Element is not present in the array." ) ;

getch( ) ;}

#include<stdio.h>

int main(){

    int a[10],i,n,m,c=0,l,u,mid;

    printf("Enter the size of an array: ");

    scanf("%d",&n);

    printf("Enter the elements in ascending order: ");

    for(i=0;i<n;i++){

         scanf("%d",&a[i]);

    }

    printf("Enter the number to be search: ");

    scanf("%d",&m);

    l=0,u=n-1;

    while(l<=u){

         mid=(l+u)/2;

         if(m==a[mid]){

             c=1;

             break;

         }

         else if(m<a[mid]){

             u=mid-1;

         }

         else

             l=mid+1;

    }

    if(c==0)

         printf("The number is not found.");

    else

         printf("The number is found.");

    return 0;

}

Sample output:

Enter the size of an array: 5

Enter the elements in ascending order: 4 7 8 11 21

Enter the number to be search: 11

The number is found.

**1. Write a simple code for linear search in c programming language**

**2. Wap a c program to search an element in an array using linear search**

The simplest algorithm to search a dictionary for a given key is to test successively against each element.

#include<stdio.h>

int main(){

    int a[10],i,n,m,c=0;

    printf("Enter the size of an array: ");

    scanf("%d",&n);

    printf("Enter the elements of the array: ");

    for(i=0;i<=n-1;i++){

         scanf("%d",&a[i]);

    }

    printf("Enter the number to be search: ");

    scanf("%d",&m);

    for(i=0;i<=n-1;i++){

         if(a[i]==m){

             c=1;

             break;

         }

    }

    if(c==0)

         printf("The number is not in the list");

    else

         printf("The number is found");

    return 0;

}

**Sample output:**

Enter the size of an array: 5

Enter the elements of the array: 4 6 8 0 3

Enter the number to be search: 0

The number is found

**Relevant answers:**

* [Role of **data** **structure** to insert an element in the **data** **structure**?](http://wiki.answers.com/Q/Role_of_data_structure_to_insert_an_element_in_the_data_structure)

using data structure an element can insert at any position easily. with out traversing through the entire list.

* [What are **data** **structures**?](http://wiki.answers.com/Q/What_are_data_structures)

DATA STRUCTURES AS ITS NAME IMPLIES DATA MEANS "VALUE' AND STRUCTURE MEANS THE WAY IT IS ORGANISED AND THE WAY IT IS ARRANGED INTO MATHEMATICAL AND LOGICAL WAY.

* [Can a **data** type be called a **data** **structure**?](http://wiki.answers.com/Q/Can_a_data_type_be_called_a_data_structure)

Yes, they can be used interchangeably, but they usually mean separate things. A type of data is something like an integer, or string. While a data structure usually refers to a linked list or tree of...

* [What is Elementary **Data** Organization in **Data** **Structures**?](http://wiki.answers.com/Q/What_is_Elementary_Data_Organization_in_Data_Structures)

elementry data organisation is a way to put the position of special element

* [Difference between **structure** and **data** **structure**?](http://wiki.answers.com/Q/Difference_between_structure_and_data_structure)

Structure is basicly a way of grouping servarl piece of information under a single name while Data structure menas with the orgnizing the data into a way by which it can be processed at any time and...

Read more: <http://wiki.answers.com/Q/What_is_searching_in_data_structure#ixzz1gaTTsMck>