Session 10

Thursday, 24 July 2025 9:35 AM

① Amstrong?

WA function which finds the Given Number is Amstrong?

x = 153;

$(1)^3 + (5)^3 + (3)^3 \Rightarrow 1 + 125 + 27 \Rightarrow 153$

then only the the Entered Number is Amstrong.

② find the Super factorial.;

x = 145 = $(1)! + 4! + 5!$

= $1 + 24 + 120 \Rightarrow 145$

the Entered Number is Super factorial

x = 25 → $2! + 5!$

= $2 + 120 \Rightarrow 122$

25 != 122 No the Entered Number is Not Super factorial.

```
int main(){
  int x,y,result=0;
  printf("Enter the Number");
  scanf("%d",&x);
  y=x;
  while(y!=0)
  {
    result = factorial(y%10) + result;
    y=y/10;
  }
  if(result == x);
    printf("The Given Value is Super fact");
  else
    printf("The Given Value is Not factorial");
}
```

145 → 145%10 ⇒ 5

result = factorial(y%10) + result;

Amstrong

result = power((y%10), 3) + result;

③ I want to store 10 Digit then I want to print those Value in Sorted order!

Variables:- We are using Variables to Store Values;

```
int x, y, z, a, b, c, d, e, f, g;
printf( )
scanf("%d %d %d %d %d %d %d");
```

42 4 19 2 7 21 24 25

Array:-

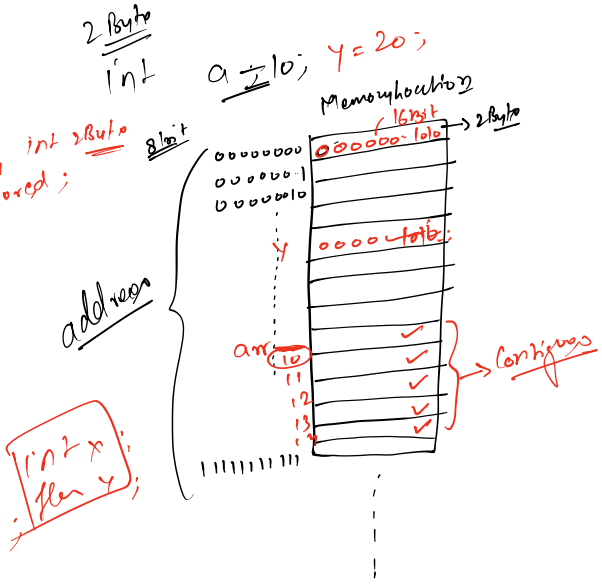
An array in C is a collection of data items of the same type, stored at contiguous memory locations. It allows you to store and manage multiple values under a single variable name.

Key characteristics of arrays in C:

- **Homogeneous:**
  All elements within an array must be of the same data type (e.g., all integers, all characters, all floats).
- **Fixed Size:**
  The size of an array must be defined at the time of declaration and cannot be changed dynamically during program execution.
- **Contiguous Memory Allocation:**
  Elements are stored in consecutive memory locations, which enables efficient access and manipulation.
- **Zero-Based Indexing:**
  Elements are accessed using an index, which starts from 0 for the first element and goes up to size - 1 for the last element.

Declaration and Initialization:

Arrays are declared by specifying the data type, the array name, and the size in square brackets. They can be initialized at the time of declaration or later.



int arr[5]; 16 Byte. Size of the array cannot be altered;

How to create/declare an Array: int x; float y;

datatype Nameofthearray [Size];

Size of the array

int arr[5];

arr can hold the starting address of allocated memory which is constant; allocated to arr can't

arr [10] the address Changed.

```
#include<stdio.h>
int main(){
    int arr[5];
    for(int i=0; i<5; i++)
    {
        printf("Enter the Value");
        scanf("%d", &arr[i]);
    }
    for(int i=0; i<5; i++)
    {
        printf("The Value is [%d]", arr[i]);
    }
}
```

0	1	2	3	4
12	14	5	2	7

i = 0 1 2 3 4 5

arr

5 = 0 — 4

6 = 0 — 5

11 = 0 — 10

```
/*Write a Program To Take
Five Input From the User in
Integer array and Print Those values*/
#include <stdio.h>
void inputArray(int a[], int size)
{
    for (int i = 0; i < 5; i++)
    {
        printf("Enter The Value Of a[%d] ", i);
        scanf("%d", &a[i]);
    }
}
void printArray(int a[], int size)
{
    // Print the Array
    for (int i = 0; i < 5; i++)
    {
        printf("The [%d]th Value Of a is %d \n", i, a[i]);
    }
}
int main()
{
    int a[5];
    inputArray(a, 5);
    printArray(a, 5);
    return 0;
}
```

int x;

scanf("%d", &x)

int x=10;

x [10]

=> Array

int arr[5];

int arr[5]={1,2,3,4,5};

Store → 10 Values in Array. and want to perform addition.

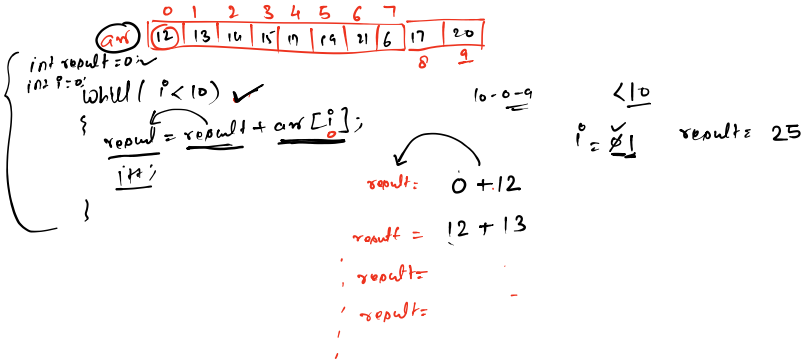
int arr[10]; → input from User.

int Addfunction (int arr[]);

{

→ { return result;

}



Array ?

→ How to take input from User in Array?

→ How to print the array?

→ How to perform addition in Array?

→ Search and item in Array ?

int array[5]: { 12, 14, 17, 19, 21 };

item[14]

return position

return -1;

int item;

printf("Enter the Value to Search");

&scanf("%d", &item);

while (i<size)

{

if (item == arr[i]);

return i;

}

return -1;

post.to.teach.to.tech@gmail.com

Sort the Array

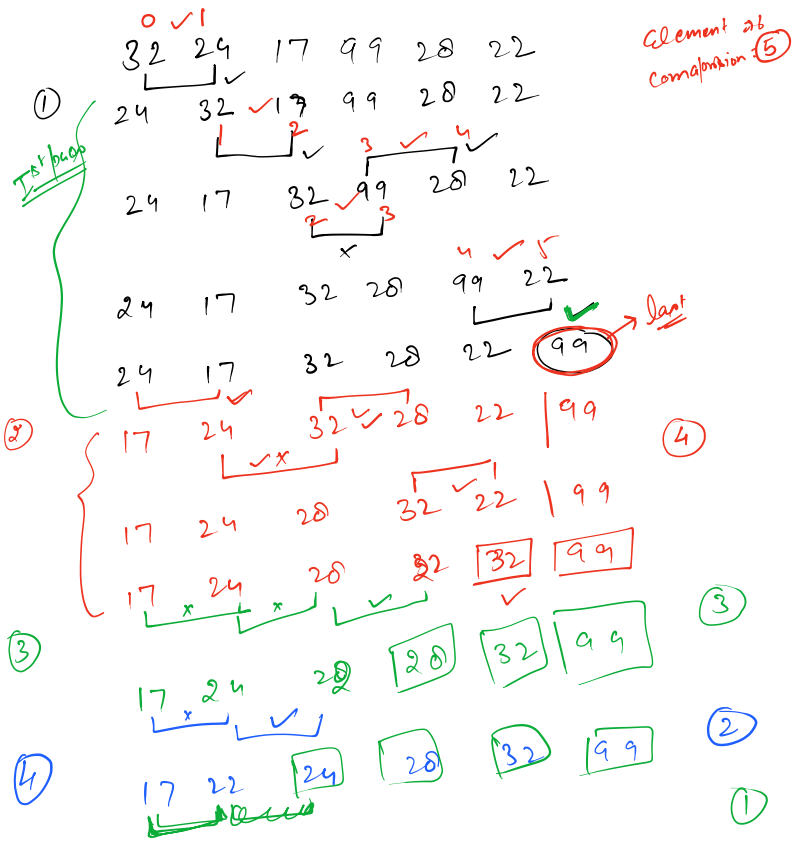
and

32	24	17	99	28	22
0	1	2	3	4	5

Ascending → Sorting

17	22	24	28	32	99
0	1	2	3	4	5

Descending (उतरते)



1. Bubble Sort → Learn and Try to Code
2. Insertion Sort —
3. Selection Sort —
4. Binary Search