2. Let Learn Arrays-1

03 August 2024 09:37

Array → Packet / Container / Bucket

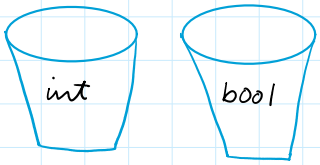
↓

which store

↓

Same type of element / item.

To store karega unn salika datatype same rahega.



Creation

Variable →

- int a;
- int a = 5;

When we talk about array, we can store multiple items

If we want to store 1 Lakh no's then

- we make 1 Lakh variables ✗
- we make 1 Lakh integer ✗
- we have to write 1 Lakh line of code ✗

We make array which can store 1 Lakh numbers.

Array Creation

Syntax

```
int arr[10000];
```

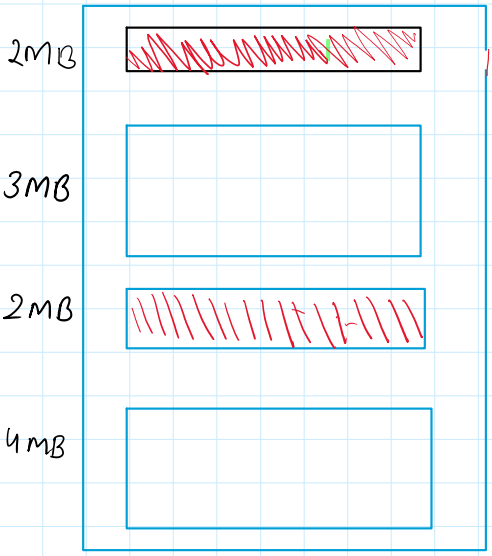
datatype — Array name — Size

```
int a[5];
```

Similar type of element store

similar type of element store

Cond^ → Contiguous memory hai.



Q) Create an array of 5MB space

Empty / Free Space = 3+4 = 7MB

But the array can't be stored because the available required space is not contiguous.

✳ GALAT

int a;

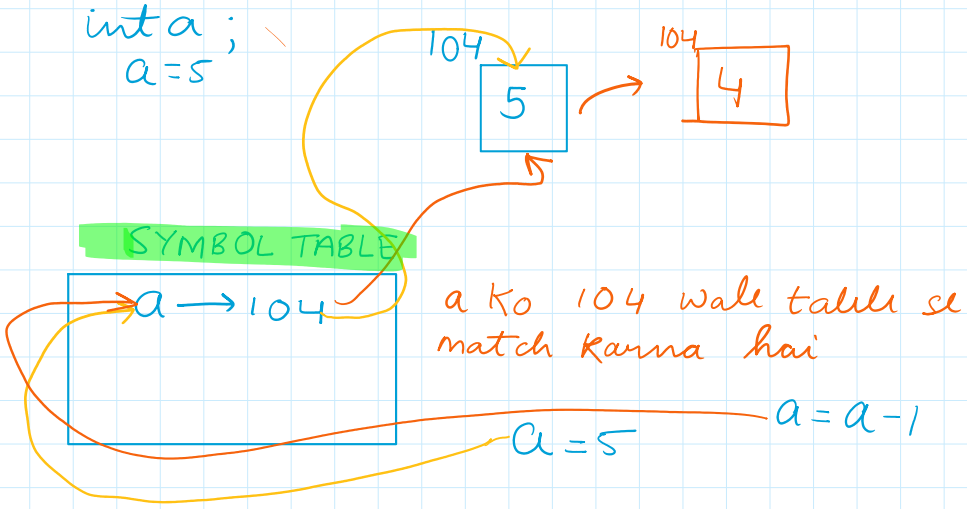
a

memory block name 'a' created

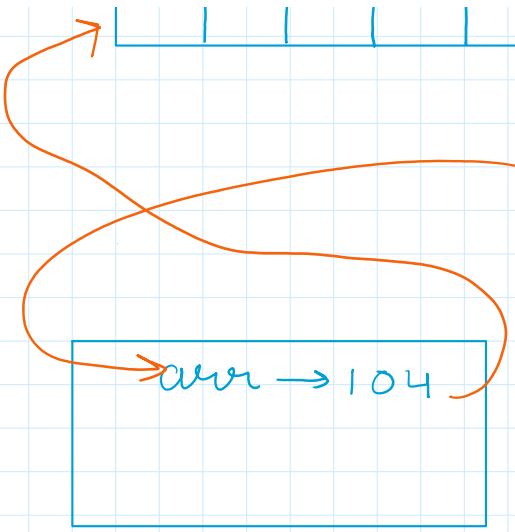
Only one thing is attached with memory i.e address.

Reality

int a;
a = 5;



int arr[5];



☆ Memory block se koi name attached nahi hota woh sirf samjhane ke liye bolte hai

Create

- 10 bool → bool arr [10];
- 15 char → Char arr [15];
- 1 int → int arr [5];
- String

int arr [5]; → Garbage value

Initialisation

int arr [5] = {10, 20, 30, 40, 50};

10	20	30	40	50

int arr [5] = {10, 20};

10	20	0	0	0

Imp:- Agar size se kam value de rakhi hai toh Rest bachchi hui space par 0 aajayega.

int arr [4] = {};

0	0	0	0

int arr [3] = {10}

10	0	0

arr → Each member ko → 10 Initialise

arr → Each member ko → 10 Initialise

fill loop

Indexing

0 Based Numbering / Indexing

int marks [6];

104 → Starting / Base address

marks [1]

marks [5]

arr size → 6
Index
↓
0-5

Important

arr size → n

index → {0-(n-1)}

int arr [3] = {10, 20, 30};

Cout << arr [4];

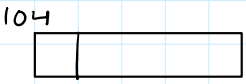
→ Garbage Value

→ Executes memory depends on compiler to compiler

Fill Function

Segment table

arr → 104



int arr [3];

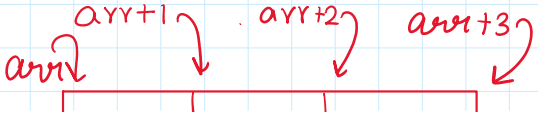
Jo bhi size hai plus kardo.

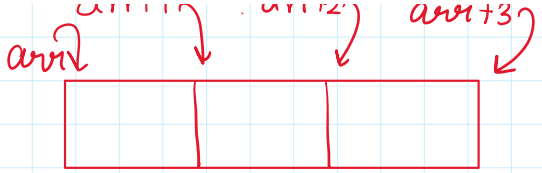
fill (arr, arr+3, 23)

Starting address

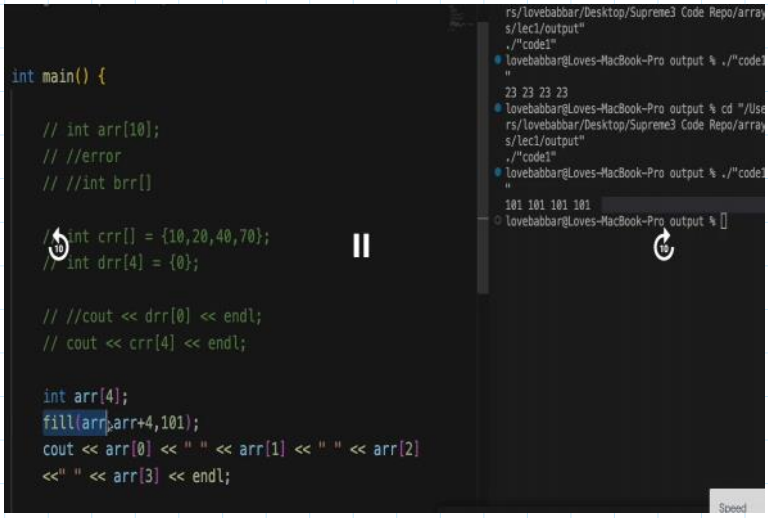
Ending address

Value of Initialisation





After running all value should be 23



Input

int a ⟶ input ⟶ cin >> a;

int arr [5]

↳ 5 Blocks

arr[0] ⟶ cin >> arr[0];

⋮

arr[4] ⟶ cin >> arr[4];

Loop

n = 5

```
for(int i=0; i<5; i++) {
    cin >> arr[i];
}
```

Why array?

↳ Jab bahut saare same type of data store karna hai toh array.

Because 100 No. ke liya 100 Variable

Because 100 No. ke liya 100 variable name hum create nahi karenge.

int size;
cin >> size;
int arr [size];
↳ Bad Practice
↳ why → Study in Dynamic Memo. allocation

#Doubt
↳ Size

int a = 5;
sizeof(a) → o/p → 4

arr → | 10 | 20 | 30 | 40 |

$\frac{\text{Size of (arr)}}{\text{Size of (int)}} = \frac{16}{4} = 4$

Number of Blocks of array

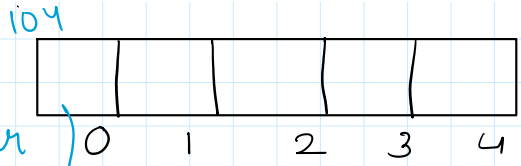
Imp It don't tell no. of elements present in an array

We don't have any formula

Only find by using another variable separately.

Q Why indexing start from 0?

104



Base address ↓ 104

arr[0]

Here, size ↓ DT ↓ Size ↓ 4 (int)

arr[i] → Value present at [Base address + Size × i]

arr[0] = ,, ,, ,, [104 + 4 × 0]

DocScanner 07-Aug-2...

Imp.

arr[i] → Value at (arr + (Size × i))

i[arr] → Sun Lo run hoga

dono chalte hai

arr[i] → i[arr]

Both Equal.

Behind scene Yeh chalta raha hota hai

*(arr+i)

*(i+arr)

Value present at address

Case-1

arr[i] → Compiler understand it as *(arr+i)

Value present at (arr+i)

i=3

val present at (104+3)

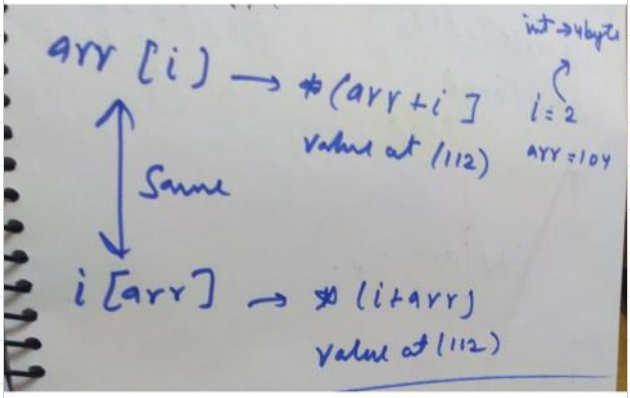
(104+3) → (int size × 3) → (11C)

Case-2

i[arr] → *(i+arr)

Value present (at 104)

i[arr] → *(i+arr)
Value present at (104) (116)



Functions in an array

```
int main() {
   int a = 5;
int c = solve(a);
}
```

```
int solve (int a) {
}
```

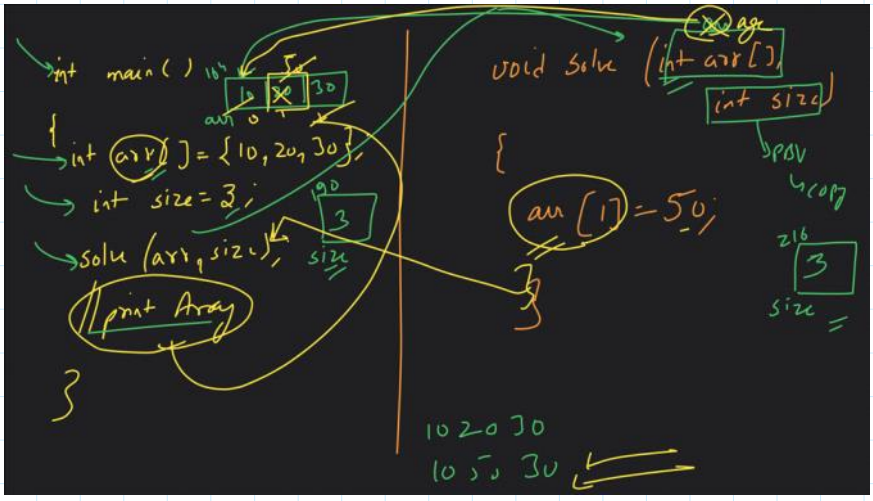
```
int main ()
{
 int arr[5] = {0};
 solve (arr, n)
```
n → Size

↳ Solve (arr, Size)

```
int solve (int arr[], int size) {
}
```

```cpp
int main() {

    // int arr[10];
    // //error
    // //int brr[]

    // int crr[] = {10,20,40,70};
    // int drr[4] = {0};

    // //cout << drr[0] << endl;
    // cout << crr[4] << endl;

    int arr[4];
    fill(arr,arr+4,101);
    cout << arr[0] << " " << arr[1] << " " << arr[2]
    <<" " << arr[3] << endl;
```

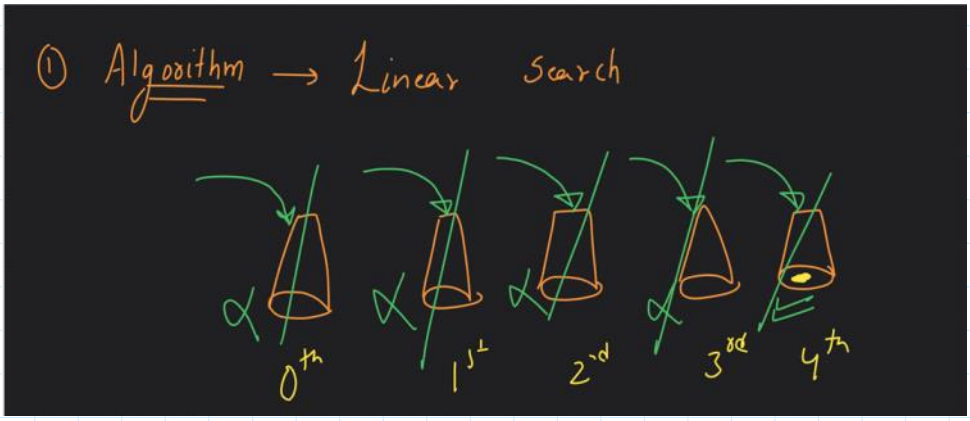
```
rs/lovebabbar/Desktop/Supreme3 Code Repo/array
s/lec1/output"
./"code1"
lovebabbar@Loves-MacBook-Pro output % ./"code1
"
23 23 23 23
lovebabbar@Loves-MacBook-Pro output % cd "/Use
rs/lovebabbar/Desktop/Supreme3 Code Repo/array
s/lec1/output"
./"code1"
lovebabbar@Loves-MacBook-Pro output % ./"code1
"
101 101 101 101
lovebabbar@Loves-MacBook-Pro output %
```



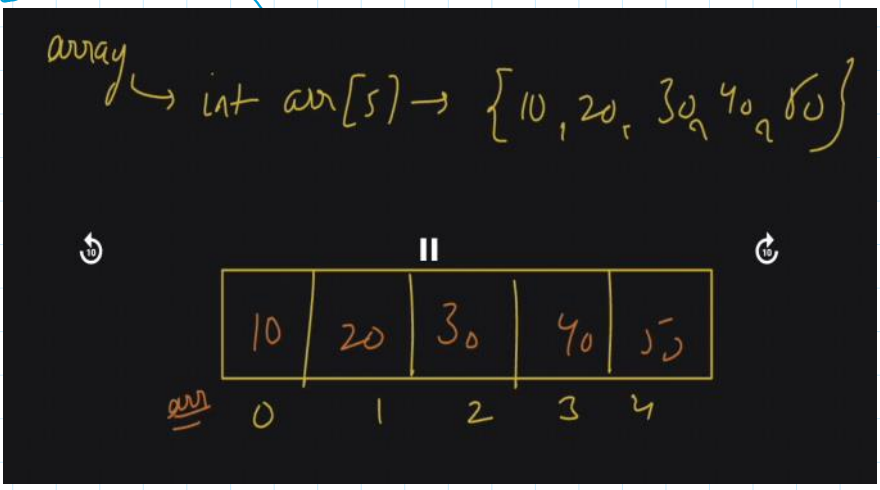

```cpp
int main ()
{
    int arr[4] = {0};
    int size = 4;

    Solve (arr, size)

    for(i=0→ i<4)
    { cout << arr[i];
    }
    return 0;
}

void Solve (int num[],
            int n)
{
    for (i=0→ i<n)
    {
        num[i] = num[i] + 10;
    }
    return;
}
```

Array → Pass By Reference

① Algorithm → Linear Search



ek ke baad ek par jana isse linear search kehte hai



Find target element present or not

if found returns true → element found

& if not present return → Element not found.

```cpp
int main() {

    int arr[] = {10,20,30,40,50};
    int size = 5;
    int target = 55;

    bool ans = findTarget(arr,size,
    target);
    cout << "ans: " << ans << endl;
    // solve1(arr, size);
    // print(arr,size);
```

```cpp
bool findTarget(int arr[], int size, int target) {
    //traverse the entire array
    for(int i=0; i<size; i++) {

        int currentElement = arr[i];

        if(currentElement == target) {
            //found
            return true;
        }
    }
    ///agar aap yaha tk pohoch gye toh
    //iska mtlb poora loop chal chuka hai
    //iska matlab poore loop me kahin bhi target nahi mila
    //iska mtlb element not found
    //iskat matlab return false;
    return false;
}
```

```cpp
bool checkTarget (arr,size,target)
{
    for (i=0; i<size; i++)
    {
        if(arr[i] == target)
        { return true;
        }
    }
    return false;
}
```

13	92	53	47
0	1	2	3

arr

47

target

Integer

↓

Range

Range

-2^{31} to $2^{31}-1$

INT_MIN INT_MAX

arr	13	21	14	17
	0	1	2	3

To Find

→ Max^M No. → ans = INT_MIN

Min^M No. → ans = INT_MAX

arr[i] > ans → ans = arr[i]

arr[i] < ans → ignore

```
int findMax(int arr[], int size) {
    int maxAns = INT_MIN;
    for(int i=0; i<size; i++) {
        maxAns = max(maxAns, arr[i]);
        // if(arr[i] > maxAns) {
        //     maxAns = arr[i];
        // }
    }
}
```

learn.codehelp.in
yashchanda2004@gmail.com
+917877058701

If Karke bhi compare Kar Sakte hai

We also have Max^M build in functions

To use INT_MIN & INT_MAX → use #include <limit.h>

```
int findMax(arr, n)
{
    int maxAns = INT_MIN;
    for(int i=0; i<n; i++)
    {
        maxAns = max(maxAns, arr[i]);
    }
    return maxAns;
}
```

arr	0	1	2	3	4
	13	42	55	70	88

maxAns = -2^{31}

n = 5

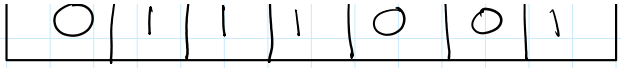
```
int findMax(arr, n)
{
    int maxAns = INT_MIN;
    for(i=0; i<n; i++)
    {
        if(arr[i] > maxAns)
        { maxAns = arr[i];
        }
    }
    return
```

```
int findMin(arr, n)
{
    int minAns = INT_MAX;
    for(i=0; i<n; i++)
    {
        minAns = min(minAns, arr[i])
    }
    return minAns;
}
```


Q. i/p mai array given hai

↳ Count → 0's / 1's

0	1	1	1	0	0	1

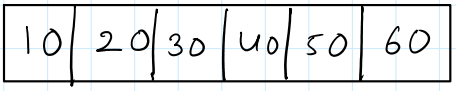


Zero_count = 0;
one_count = 0;

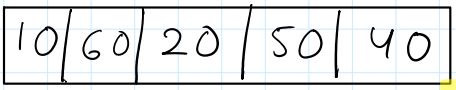
```cpp
void printZeroesAndOnes(int arr[], int n) {
    int zeroCount = 0;
    int oneCount = 0;

    //traverse Array
    for(int i=0; i<n; i++) {
        int currElement = arr[i];

        if(currElement == 0) {
            zeroCount++;
        }
        if(currElement == 1) {
            oneCount++;
        }
    }
    cout << "Total Zeroes: " << zeroCount << endl;
    cout << "Total Ones: " << oneCount << endl;
}
```

Q. Extreme Printing:

i/p



o/p



2 Pointer Method

Approach

i = 0
j = n + 1

cout << i;
i++
cout << j;
j--

Stop j < i

```cpp
void extremePrint(int arr[], int n) {
    int i = 0;
    int j = n-1;

    while(i < j) {
        cout << arr[i] << " ";
        i++;
        cout << arr[j] << " ";
        j--;
    }
}
```

Not Work For Odd No's

↓

Change Condition ($i \leq j$)

↓

It will print middle Element 2 times

Method-1

Use `if (j==i) { cout<<i<<" "; }`

`else { cout<<i<<" "; i++; cout<<j<<" "; j--; }`

Method-2

```
cout << i << " ";
i++;
if (j>i) {
    cout<<arr[j]<<" ";
    j--;
}
```

HOMEWORK QUESTIONS

Important

Q1) Swap int a → 5
int b → 4

↳ Various Methods

((+,-), temp, ^ (XOR), swap)

o/p → a → 4
b → 5

Q2) Reverse an Array ?