Short-term Hands-on Supplementary Course on C programming

Session 4:
Arrays

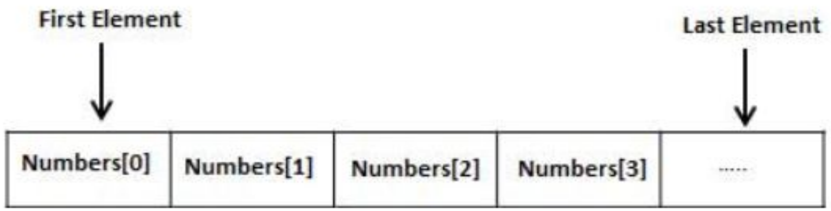
Nandakishor. V
Nitheesh Kumar. N

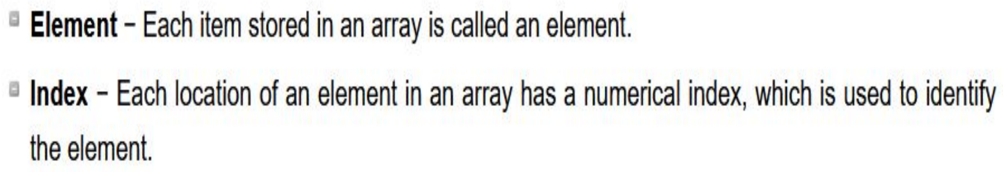
Agenda

- What are arrays?
- Declaring & Initializing arrays
- Traversing through an array
- Live Code Demo - Linear search
- Multidimensional array
- Matrix
- Live Code Demo - Matrix Addition and Multiplication
- Tutorial

What are arrays?

- Arrays a kind of data structure that can store a fixed-size sequential collection of elements of the same type.
- An array is used to store a collection of data, but it is often more useful to think of an array as a collection of variables of the same type.





- **Element** – Each item stored in an array is called an element.
- **Index** – Each location of an element in an array has a numerical index, which is used to identify the element.

Declaration arrays

Syntax-

```
type arrayName [ arraySize ];
```

Example-

```
double balance[10];
```

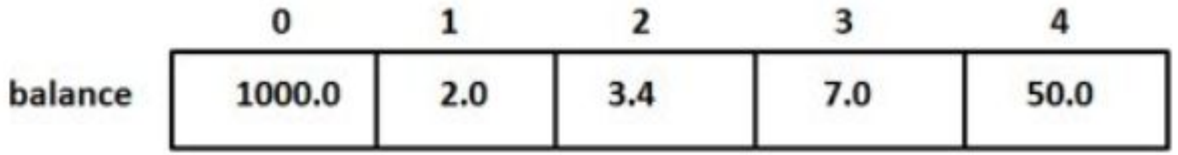
- This is called a single-dimensional array.
- The arraySize must be an integer constant greater than zero and type can be any valid C data type.

Initializing arrays

```
double balance[5] = {1000.0, 2.0, 3.4, 7.0, 50.0};
```

```
double balance[] = {1000.0, 2.0, 3.4, 7.0, 50.0};
```

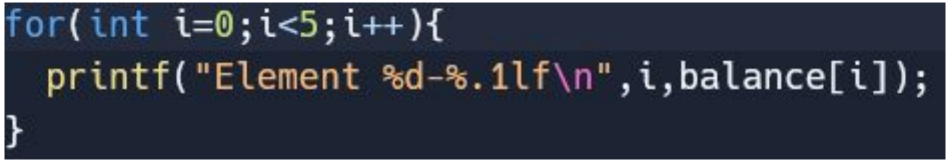
```
balance[4] = 50.0;
```



The number of values between braces { } cannot be larger than the number of elements that we declare for the array between square brackets [ ]

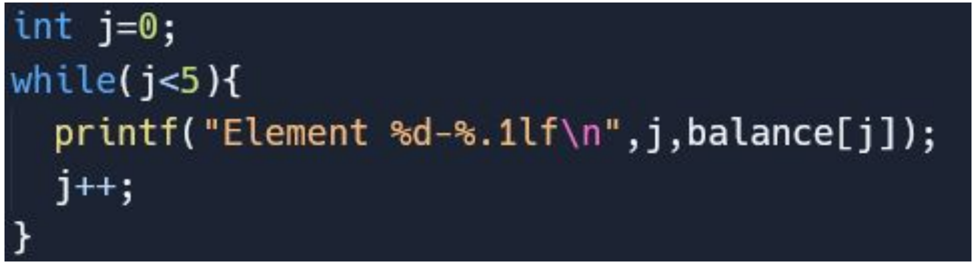
Traversing through an array

For loop-



```
for(int i=0;i<5;i++){
  printf("Element %d-%.1lf\n",i,balance[i]);
}
```

While loop-



```
int j=0;
while(j<5){
  printf("Element %d-%.1lf\n",j,balance[j]);
  j++;
}
```

- You can use any looping statement to traverse through an array.
- But for loop is mainly used.

Live code demo-Linear search

1) Searching an element in an array, if found return the index of the element, else output “Not found”.

Multidimensional arrays

Syntax-

```
type name[size1][size2]...[sizeN];
```

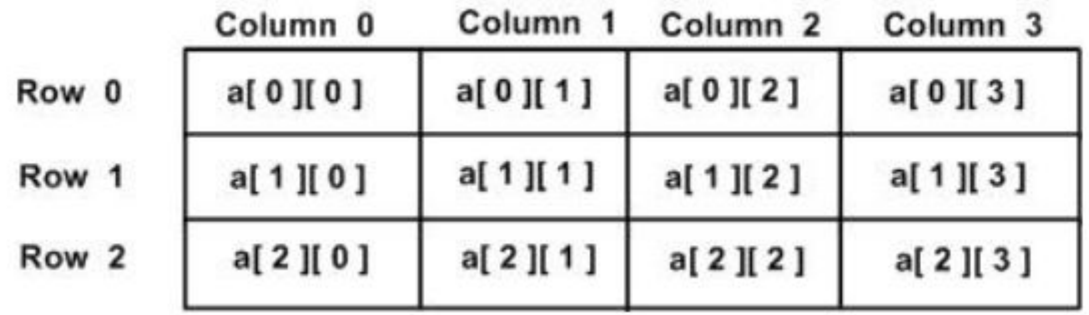
Example-

```
int threedim[5][10][4];
```

2D arrays- Matrix

Syntax-

```
type arrayName [ x ][ y ];
```



	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

- every element in the array a is identified by an element name of the form a[i][j].
- where 'a' is the name of the array, and 'i' and 'j' are the subscripts that uniquely identify each element in 'a'.

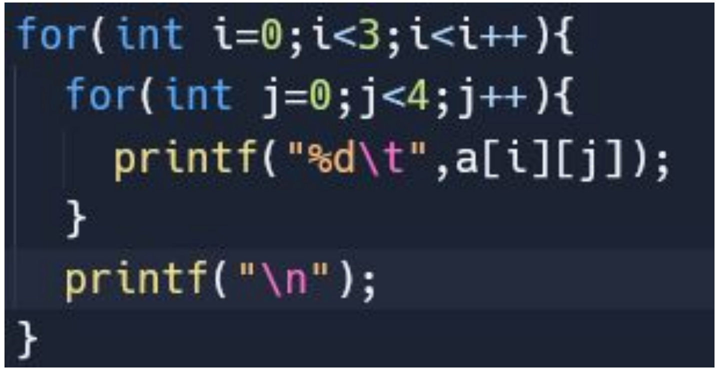
Initializing Matrix

```
int a[3][4] = {
   {0, 1, 2, 3} ,
   {4, 5, 6, 7} ,
   {8, 9, 10, 11}
};
```

Accessing an element in a matrix

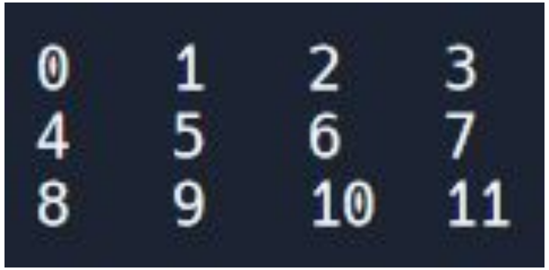
```
int val = a[2][3];
```

Traversing through a matrix



```
for(int i=0;i<3;i<i++){
  for(int j=0;j<4;j++){
    printf("%d\t",a[i][j]);
  }
  printf("\n");
}
```

Output-



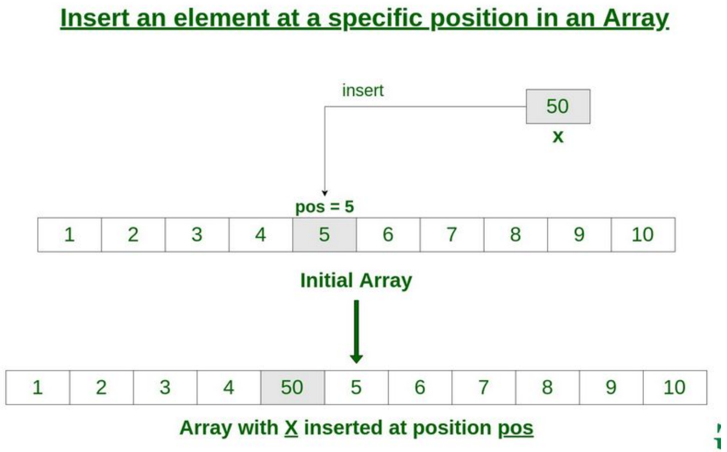
```
0    1    2    3
4    5    6    7
8    9    10   11
```

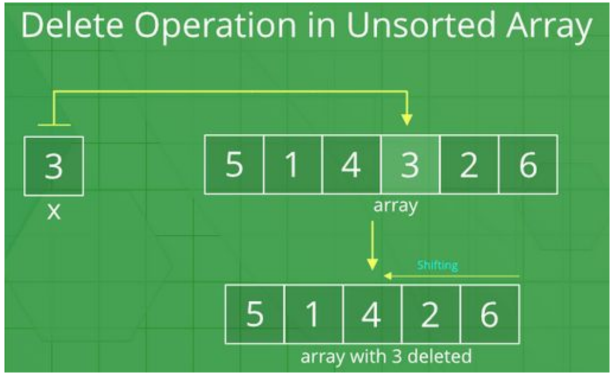
Live code demo- matrix addition and multiplication

1) Write a C program to implement matrix addition
2) Write a C program to implement matrix multiplication

Tutorial

1) Insert an element in the beginning, end and in between 2 elements in the given array.
2) Delete an element in an array.


Insert an element at a specific position in an Array
insert
50
x
pos = 5
1 2 3 4 5 6 7 8 9 10
Initial Array
1 2 3 4 50 5 6 7 8 9 10
Array with X inserted at position pos



Thank You for attending!

Contact us regarding any questions through email
nandakishor2010608@ssn.edu.in
nitheesh2010343@ssn.edu.in