2. Explain the concept of one dimensional array with example, diagrammatic layout.

ANS: A one-dimensional array is a collection of elements of the same data type stored in contiguous memory locations.

Example:- int Arr[5] = {10, 20, 30, 40, 50}

	0	1	2	3	4
iArr	10	20	30	40	50

3. Explain the concept of 2D Array with example and diagramatic layout.

ANS: A 2D array is a collection of elements of the same data type stored in a table-like structure with rows & columns.

Example:- int Arr[3][3]

		Columns		
	0	1	2	3
Row	1	4	5	6
	2	7	8	9

4. Write a Program to display the content of below array. draw its diagrammatic layout

Const int arr[6] = {10, 20, 30, 40, 50, 60};

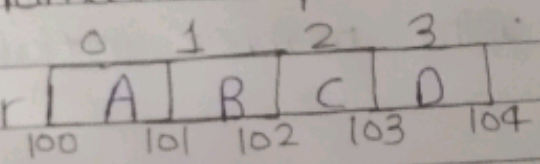
5. Write a program to create an array of all the Primitive data types, draw diagrammatic layout

Ans:-

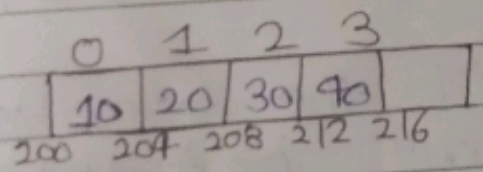
```
#include<stdio.h>
int main()
{
    char cArr [4] = { 'A', 'B', 'C', 'D' };
    int iArr [4] = { 10, 20, 30, 40 };
    float fArr [4] = { 10.5, 20.5, 30.5, 40.5 };
    double dArr [4] = { 10.0, 20.0, 30.0, 40.0 };

    return 0;
}
```

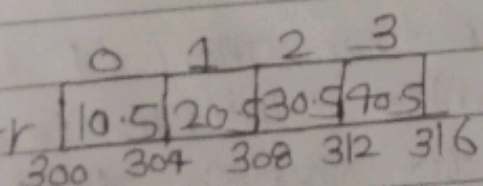
Diagramatic Layout :-



sizeof(cArr) = 4 bytes
sizeof(cArr) = 1 byte
cArr[2] = C
cArr = 100



sizeof(iArr) = 16 bytes
size (iArr[2]) = 4 byte
iArr[2] = 30
iArr = 200



sizeof(fArr) = 16 bytes
sizeof(fArr[2]) = 4 byte
fArr[2] = 30.5
iArr = 300

4. Write a program to display the content of below array, draw its diagrammatic layout

const int arr[6] = { 10, 20, 30, 40, 50, 60 };

```
#include<stdio.h>
int main()
{
    const int Arr[6] = { 10, 20, 30, 40, 50, 60 };

    for (int i = 0; i<6 ; i++) {
      printf( %d , arr[ i ]);
}

    return 0;
}
```

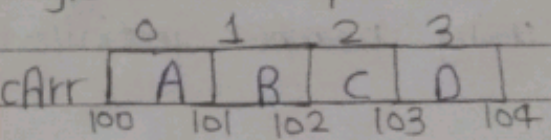
5. Write a program to create an array of all the Primitive data types, draw diagrammatic layout

Ans:-

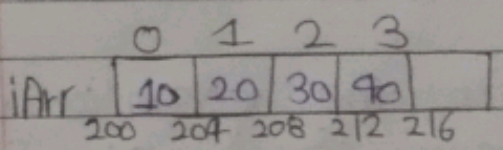
```
#include<stdio.h>
int main()
{
    char cArr [4] = {'A', 'B', 'C', 'D'};
    int iArr [4] = {10, 20, 30, 40};
    float fArr [4] = {10.5, 20.5, 30.5, 40.5};
    double dArr [4] = {10.0, 20.0, 30.0, 40.0};

return 0;
}
```

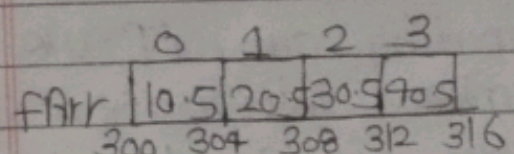
Diagramatic Layout :-



sizeof(cArr) = 4 bytes
sizeof(cArr) = 1 byte
cArr[2] = C
cArr = 100



sizeof(iArr) = 16 bytes
size(iArr[2]) = 4 byte
iArr[2] = 30
iArr = 200



sizeof(fArr) = 16 bytes
sizeof(fArr[2]) = 4 byte
fArr[2] = 30.5
iArr = 300

	0	1	2	3
dArr	10.0	20.0	30.0	40.0
	400	408	416	424	432

sizeof(dArr) = 32 bytes
sizeof(dArr[2]) = 8 byte.
dArr[2] = 30.0
dArr = 400

6. Explain the different approaches (ways) to create the array in c/c++.

Ans :-

There are multiple ways to create the array in c/c++.

1. With size & initialization list.

We can create the array by specifying the length of array as well as initializing the array immidiately using member initialization list.

Syntax :-

Data types _ Array Name [Length] =
{ Value1, Value2, Value3 --- Valuen };

2. Array initialization with the declaration with the declaration & without size & just without length.

At the time of creating array if we initialize it with member initialization list then its optional to specify the length of array because the length of array is automatically deduced by compiler.

E.x. int Brr = [10, 20, 30];
Syntax :- Data type -- Array name = [Value1, Value2 Value3 ---- Valuen]

3. declaration & initialization of array with less numbers.

It is also possible to provide to specify the length of array with some value but initialize less number of elements in it.

E.x. int crr[7] = {10, 20, 30, 40};
Size of crr = 24 bytes

4. We can also create the array & initialize it by using the concept of member by member initlalization.

5 We can also create the array of constant array.

7. Draw the diagramatic layout & the Matrix of below array.

int arr[3][3] = {10, 20, 30, 40, 50, 60, 70, 80, 90};

Diagramatic layout :-

		Columns	
	0	10	20	30
Row	1	40	50	60
	2	70	80	90

Matrix representation :-

[0][0]		[0][1]		[0][2]	
[1][0]		[1][1]		[1][2]	
[2][0]		[2][1]		[2][2]	

8 Note down the properties of Array in C/C++;

Ans:

The properties of Array in C/C++ is as follows:

1. Fixed size:-

a. It is a fixed size.

b. Length of array must be known at compile time.

c. Size of array can't change at runtim

2. Homogenius:-

Array can store one type of data elements.

3. Array indexing:-

a. Array index always starts from 0.

b. Array is last element is n-1 whe n is the length of array.

4. Dimensions of Array:-

1d, 2d, 3d, multiD where stat for dimensions.

5. Continuous Storage:-

All the array elements are st continuously after another me

6. Random Access:-

We access the array element using subscript operator

7. No check of index out of bounds :-

a. In C, C++ there is no provision for index out of bound checking.
It can compile the program but produces unexpect output runtime.

8. Length of Array in C :-
In C programing language we don't have any inbuilt function find the length of array.

a. using size of operator
b. pointer arithmetic
c. using loops

9. Explain the concept of length of Array, size of Array.

Ans:

Length of Array :
The length of an array refers to the number of elements it can hold. It is also known as the array's capacity. The length is determined when the array is created and can't be changed later.

Size of an array :
The size of an array refers to the actual number of elements currently stored in the array. It can be less than or

equal to the length of array.

10. Write notes on Data type modifiers and Data type Qualifiers.

Ans:

Data type Modifiers

Data type modifiers alter the size or range of a data type.
In case of data type modifier there are 2 subtypes.

1. The modifier which affects the sign
2. The modifier which affects the size

There are 2 subtypes of sign & size.

1. sign - It can store +ve, -ve & 0 value.
2. unsign - It can store only +ve value.

1. short - Reduces the size of the data type
2. Long - Increases the size of the data type

Data type Qualifiers

Data type qualifiers specify additional properties of a data type

In case of Data type qualifier there are 2 subtypes. 2.

1. const :
2. Volatile :

The value of constant data type is not modified.