LECTURE - 36

ARRAYS (PART 05)

PROGRAMMING IN 'C'

2-D array → [Row / Column]

int arr [x] [y] → Create an array of x arrays where each array consist 'y' elements

```
int arr [3] [4] = {
        { 10, 20, 30, 40},
        { 50, 60, 70, 80},
        { 90, 100, 110, 120}
        }
```

3 arrays

arr [0] ← { 10, 20, 30, 40},

arr [1] ← { 50, 60, 70, 80},

arr [2] ← { 90, 100, 110, 120}

arr [0] [0] = 10	20	30	40
50 = arr [1] [0]	60	70	80
90	100 = arr [2] [1]	110	120 = arr [2] [3]

printf ("%d", arr [0] [2]) ⇒ 30

Addressing of 2-D Array:

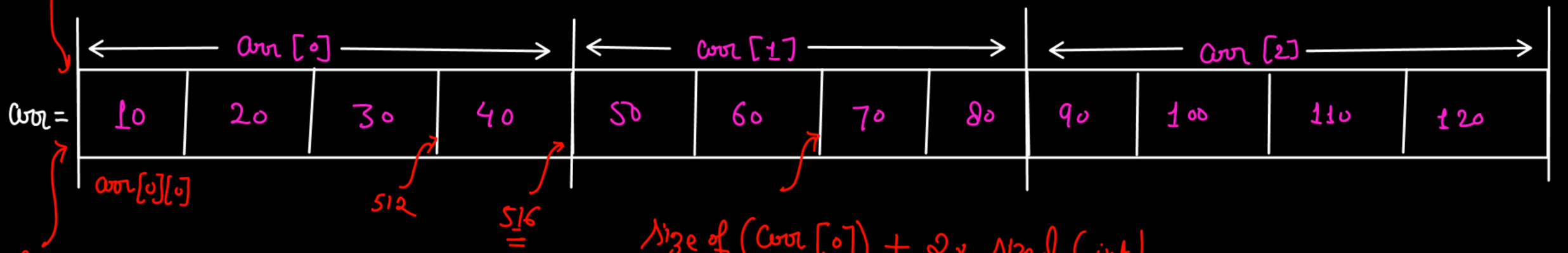
```
int arr [3] [4] = {
    { 10, 20, 30, 40 },   <- arr[0]
    { 50, 60, 70, 80 },   <- arr[1]
    { 90, 100, 110, 120 }
}
```

int → 4 Bytes; [3] → rows i; [4] → Col. j

Size of arr = 12 element x 4

Sizeof arr = ([3] x [4]) * 4

Size of 2D array = i x j * Size of (arr[0])



Assume = 500

Size of (arr [0]) + 2 x Size of (int)

= 12 Bytes + 2 * 4 = 12 + 8

= 20 Bytes

& arr[1][2] = 500 + 20

= 520

twoDarray.c

```c
#include<stdio.h>
#include<conio.h>
#include<math.h>
int main(){
    int arr[4][3] = {
        {10,20,30},
        {40,50,60},
        {70,80,90},
        {100,110,120}
    };
    printf("The address of array is: %d \n", arr);
    printf("The size of arr is: %d \n", sizeof(arr));
    for (int i = 0; i<4; i++){
        for (int j = 0; j<3; j++){
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
    return 0;
}
```

```
The address of array is: 6422000
The size of arr is: 48
10 20 30
40 50 60
70 80 90
100 110 120
PS>
```

Q.1. Write a Program to search an element in an array.

twoDarray.c ● searchingarray.c ×

DPP Conditional > arrays > searchingarray.c > main()

```c
#include<stdio.h>
int main(){
    int arr[] = {10,30,45,65,25,57,85,96,34,21,35,9,10};
    printf("Enter the number you want to find:\n");
    int n;
    scanf("%d", &n);
    int flag = 0;
    for (int i = 0; i< (sizeof(arr)/4); i++){
        if (n == arr[i]){
            flag = 1;
            break;
        }
    }
    if (flag == 1){
        printf("Found!!!");
    }
    else{
        printf("Not found");
    }

    return 0;
}
```

```
Enter the number you want to find:
99
Not found
PS>
```

Q.2. Write a program to Calculate sum and average of all elements in One-Dimensional array.

twoDarray.c ● searchingarray.c sumandavg.c ×

PP Conditional > arrays > sumandavg.c > ...

```c
#include<stdio.h>

int sum(int *arr, int size){
    int total = 0;
    for (int i=0; i<size; i++){
        total += arr[i];
    }
    return total;
}

float avg(int *arr, int size){
    float mean = sum(arr, size) / size;
    return mean;
}

int main(){
    int arr[] = {10,30,45,65,25,57,85,96,34,21,35,9,10};
    int size = sizeof(arr)/4;
    printf("The sum is: %d \n", sum(arr, size));
    printf("The average is: %f \n", avg(arr, size));
    return 0;
}
```

```
The sum is: 522
The average is: 40.000000
PS>
```

Q.3. Program to find maximum and minimum value from an array.

arr = [10, 9, 20, 8, 7, 15]

max = ~~0~~ ~~10~~ 20

arr[0] = 10

arr[1] = 9 → if arr[1] > max

then max = arr[1]

arr[2] = 20 , → if arr[2] > max

max = arr[2]

⋮

DPP Conditional > arrays > max.c > main()

```c
#include<stdio.h>

int main(){
    int arr[] = {10,30,45,65,25,57,85,96,34,21,35,9,10};
    int max = 0;
    int min = arr[0]; // first element (assume that first element is
    minimum)
    for (int i = 0; i< sizeof(arr)/4;  i++){
        if (arr[i]>max){
            max = arr[i];
        }
        if (arr[i]<min){
            min = arr[i];
        }
    }
    printf("The maximum value of array is: %d\n", max);
    printf("The minimum value of array is: %d", min);
    return 0;
}
```

```
The maximum value of array is: 96
The minimum value of array is: 9
PS>
```

Q.4. Program to reverse an array using stack.

arr1[] = [10, 20, 30, 40]

arr2[] = [arr[3], arr[2], arr[1], arr[0]]

Conditional > arrays > reversearr.c > main()

```c
int main(){
    int arr[] = {10,30,45,65,25,57};
    int reversed[6];
    int j = 0;
    for(int i = 5; i>= 0; i--){
        reversed[j] = arr[i];
        j++;
    }
      printf("Original array: \n");
    for (int i = 0; i<6; i++){
        printf("%d ", arr[i]);
    }
    printf("\nReversed array: \n");
    for (int i = 0; i<6; i++){
        printf("%d ", reversed[i]);
    }
    return 0;
}
```

```
Original array:
10 30 45 65 25 57
Reversed array:
57 25 65 45 30 10
PS>
```

Q.5. Program to find duplicate elements from an array.

twoDarray.c ● searchingarray.c sumandavg.c max.c reversearr.c findDuplicate.c ●

OPP Conditional > arrays > findDuplicate.c > main()

```c
#include<stdio.h>
int main(){
    int arr[] = {10,30,45,65,10,57,85,96,30,21,35,9,10};
    int len = sizeof(arr) / 4;

    for (int i = 0; i<len; i++){
        for (int j = 0; j<len; j++){
            if (arr[i] == arr[j]){
                if (i==j){
                    continue;
                }
                else{
                    printf("Duplicate: %d\n", arr[j]);
                }
            }
        }
    }
    return 0;
}
```

```
Duplicate: 10
Duplicate: 10
Duplicate: 30
Duplicate: 10
Duplicate: 10
Duplicate: 30
Duplicate: 10
Duplicate: 10
PS>
```

Tomorrow

To Structures ✓

[Array ✓ & Structures ✓]

⇓

DSA