# 1D, 2D, MultiDimensional Array Assignments

**Mandatory**

**1D Array**

1. Refer the code snippet and answer the queries

int main()

{

int array[100];

int \*ptr;

// do something

}

**Q1: Can pointer be used in Array-style syntax? e.g. ptr[10], ptr[0]**

A : Yes, the pointer can be used in Array style syntax

**Q2: Can Array be used in Pointer-style syntax? e.g. \*array, \*(array + 0), \*(array + 10)**

A: Yes, an array can be used in pointer style syntax

**Q3: is ptr++ valid?**

A: Yes, it is valid. It moves to the next address

**Q4: is array++ valid?**

A: It is invalid. Array address cannot be changed

**Q5: what is sizeof(array)?**

A: Sizeof(array) gives the total size of array

**Q6: what is sizeof(ptr)?**

A: Gives the sizeof ptr generally 4 bytes

1. **Refer the code snippet below. Comment on the other elements (other than those that are explicitly initialized) of all array variables in code snippet below.**

**#define MAX 100**

int main()

{

int arr[MAX] = {11,22,33};

int arr1[MAX]={0};

static int arr2[MAX];

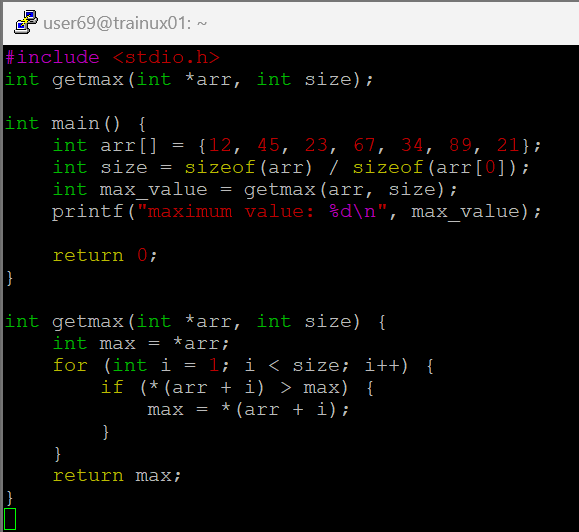
}

Arr[3] to arr[99] à initialized to 0

Arr[0] to arr[99] à initialized to 0

All elements are initialized to 0 as they are static

1. **Refer the program “array\_pointer.c”. Add a function getmax() to find the maximum in the array and call in main() and display the result.**

****

**A screenshot of a computer

Description automatically generated**

1. **Extend the code given below to read N and a start value from the user to perform the given operations.**

**#define MAX 100**

int main()

{

int arr[MAX] = {11,22,33};

}

**Add the following functions choosing proper input, output and return.**

1. **init() - Use the inputs to initialize the first N elements of the array with N consequetive values starting with given start value .**
2. **update() – increment value of every element in the array**
3. **display() – display the contents of array**

A screen shot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

**2D, MultiDimensional Arrays**

1. Implement sort() to sort a given array. Refer the code snippet below.

int main()

{

char arr[]= “xaybz”;

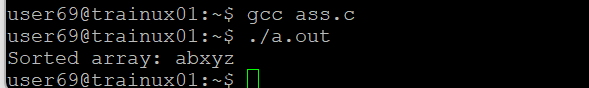
sort(arr, sizeof(arr)/sizeof(arr[0]);

return 0;

}

A screenshot of a computer program

Description automatically generated



**2. Refer the code snippet below.**

int main()

{

char arr[][3] = {

sort(arr, sizeof(arr)/sizeof(arr[0]);

return 0;

}

**Allow user to perform the following operations.**

* 1. **init() - initialize the array and return 0**
  2. **search\_update() – search for a given element in array and if found update it to given value and return 0 else return 1**
  3. **display() – traverse and display array contents**

**For the functions, pass array and other required arguments to functions and return as per requirement**

A screen shot of a computer program

Description automatically generated

