# 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]

Ans) **Yes**, a pointer can be used in array-style syntax like ptr[10].

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

Ans) Yes, an array can be used in pointer-style syntax like \*array and \*(array + 10).

Q3: is ptr++ valid?

Ans) **Yes**, ptr++ is valid.

Q4: is array++ valid?

Ans) **No**, array++ is not valid because array is a constant pointer.

Q5: what is sizeof(array)?

Ans) sizeof(array) will give the size of the entire array, typically 400 bytes (for int of 4 bytes and 100 elements).

Q6: what is sizeof(ptr)?

Ans) sizeof(ptr) will give the size of the pointer, typically 4 or 8 bytes, depending on the system architecture.

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];

}

Ans)

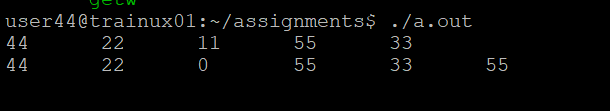
* arr: The first three elements are initialized to {11, 22, 33}, and the remaining elements are initialized to 0.
* arr1: All elements are initialized to 0, because only arr1[0] is explicitly set to 0.
* arr2: All elements are automatically initialized to 0, due to the static keyword.

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 computer code on a black background

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.

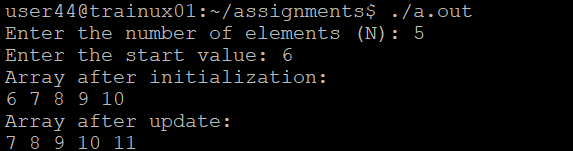
1. init() - Use the inputs to initialize the first N elements of the array with N consequetive values starting with given start value .
2. update() – increment value of every element in the array
3. display() – display the contents of array



A screen shot of a computer

Description automatically generated

Output:



**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 computer screen shot of a program code

Description automatically generated**

****

1. 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. iinit() - 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

A screen shot of a computer

Description automatically generated

Output:

A screen shot of a computer

Description automatically generated