# 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, we can used pointer in array-style syntax ptr[10] or ptr[0] can be written as \*(ptr+10) and \*(ptr).

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

Ans) Yes.

Q3: is ptr++ valid?

Ans) Yes, it is valid. We can move the pointer to the next memory location i.e incrementing the ptr+1.

Q4: is array++ valid?

Ans) No, It’s not valid. It is a constant pointer which is not changeable.

Q5: what is sizeof(array)?

Ans) It will give the size of the array in bytes i.e array[100] is 400 bytes.(100\*4).

Q6: what is sizeof(ptr)?

Ans) Size of ptr is 4/8 bytes depends on the system.

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)- int arr[Max]={11,22,33};

arr[0] to arr[2] : the values are initialized as 11,22,33.(non-static)

arr[3] to arr[99] : will allocate a garbage values which are uninitialized.(non-static)

* arr1[MAX] = {0}

All initialized to 0(non-static)

* static int arr[MAX]

No elements are initialized and it is a static array automatically set to 0.

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.

Ans)

A computer screen shot of white text

Description automatically generated

A black screen with white text

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()

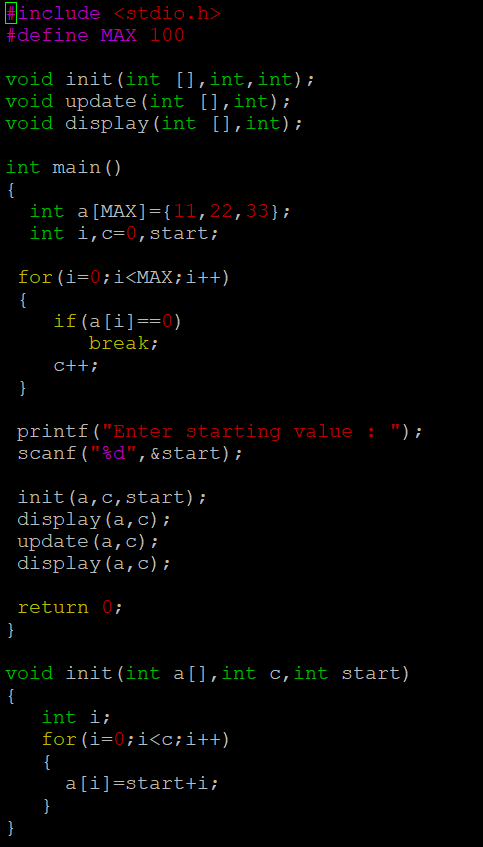
{

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 screen shot of a computer

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 computer screen shot of a program code

Description automatically generated

A screen shot of a computer 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. 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

A computer screen shot of a program code

Description automatically generated

A computer screen shot of a program code

Description automatically generated

A screenshot of a computer program

Description automatically generated

A screen shot of a computer

Description automatically generated