**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 be used in array style syntax

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

A : yes , array can be used in pointers

Q3: is ptr++ valid?

A: yes, ptr++ is valid used in memory size increment

Q4: is array++ valid?

A; No, array++ is invalid

Q5: what is sizeof(array)?

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

Q6: what is sizeof(ptr)?

A: gives the size of the pointer itself

2. 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];

}

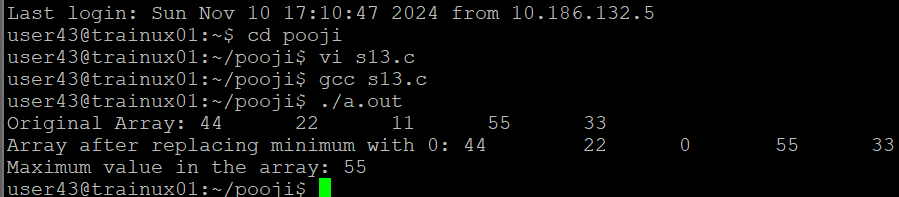
A : Normally, when an array is declared inside a function (local variable), it is created when the function is called and destroyed when the function exits. This means the array is re-initialized every time the function is called, and its values are lost between calls.

When you declare an array as static inside a function, its values are preserved between function calls. The array is not destroyed when the function exits. It retains its values and exists for the entire duration of the program (i.e., its lifetime is the same as the program's)

3. 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 program

Description automatically generated



4. 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.

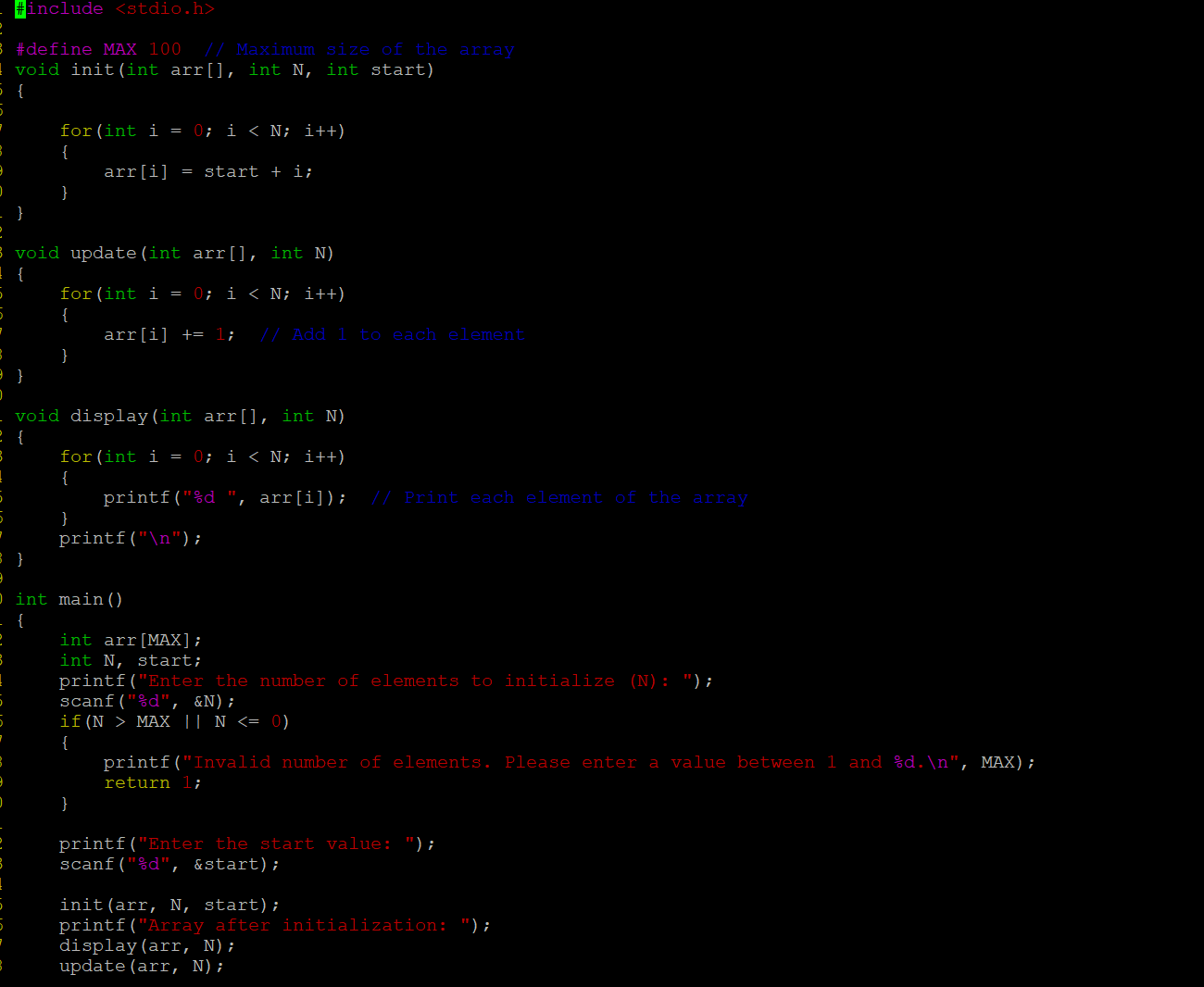
a. init() - Use the inputs to initialize the first N elements of the array with N consequetive values starting with given start value .

b. update() – increment value of every element in the array

c. display() – display the contents of array

A screenshot 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 code on a black background

Description automatically generated:

A screen shot of a computer screen

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.

a. init() - initialize the array and return 0

b. search\_update() – search for a given element in array and if found update it to given value and return 0 else return 1

c. display() – traverse and display array contents

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

A computer screen shot of many colorful text

Description automatically generated

