**1D, 2D, MultiDimensional Array Assignments**

**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 arraystyle syntax

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

**Ans:**yes the array can be used in pointer style

Q3: is ptr++ valid?

**Ans:**ptr++ is valid

Q4: is array++ valid?

**Ans:**no its not valid we cant increment the array

Q5: what is sizeof(array)?

**Ans:**array of size of array depend on the compiler.In most of the cases it will give the output

Q6: what is sizeof(ptr)?

**Ans:**the value of size of ptr is the size of pointer that is 4bytes on 32 bit and 8 bytes on 64 bit system

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];

}

**Ans:**

arr[MAX]:elements 3 to 99 are unintialized .

arr1[MAX]:all the elment are intailized to 0 due to the initialization list {0}

arr2[Max];being declared as static the elements are initialized to 0 default

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

Description automatically generated

A screenshot of a computer

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.

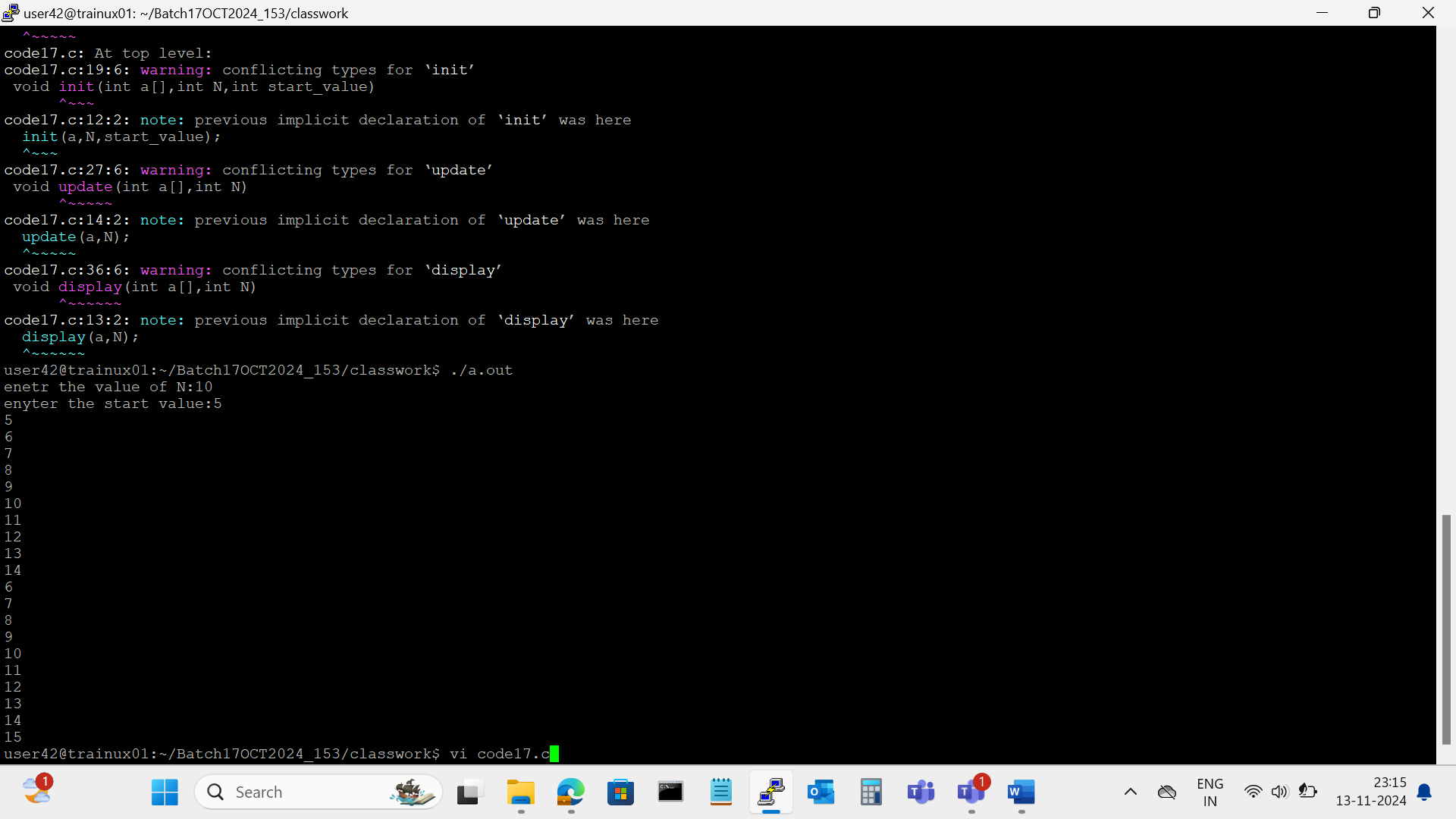
a. init() - Use the inputs to initialize the first N elements of the array with N consequetive values starting with given start value .

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 screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

2. Refer the code snippet below.

int main()

{

char arr[][3] = {

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

return 0;

}

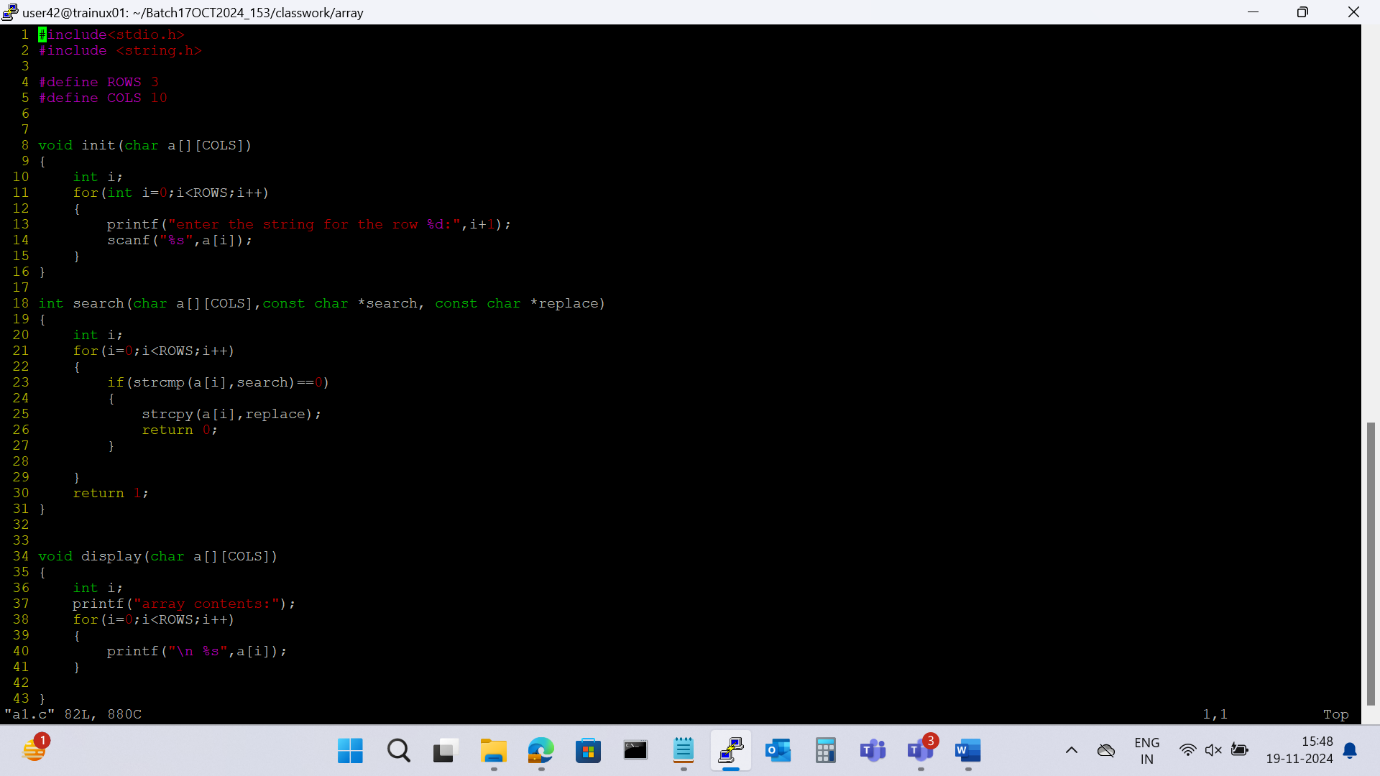
Allow user to perform the following operations.

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 screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated