**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.

For example, ptr[10] and ptr[0] are valid if ptr points to a valid memory location.

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.

For example, \*array, \*(array + 0), and \*(array + 10) are valid and equivalent to array[0] and array[10].

Q3: is ptr++ valid?

Ans: Yes, ptr++ is valid. It increments the pointer to point to the next integer in memory.

Q4: is array++ valid?

Ans: No, array++ is not valid. Arrays are not modifiable lvalues, so you cannot increment the array name.

Q5: what is sizeof(array)?

Ans: sizeof(array) is the size of the entire array in bytes.

For int array[100], it would be 100 \* sizeof(int).

Q6: what is sizeof(ptr)?

Ans: sizeof(ptr) is the size of the pointer itself, not the size of the array it points to.

It typically returns the size of a pointer on your system (e.g., 4 or 8 bytes).

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];

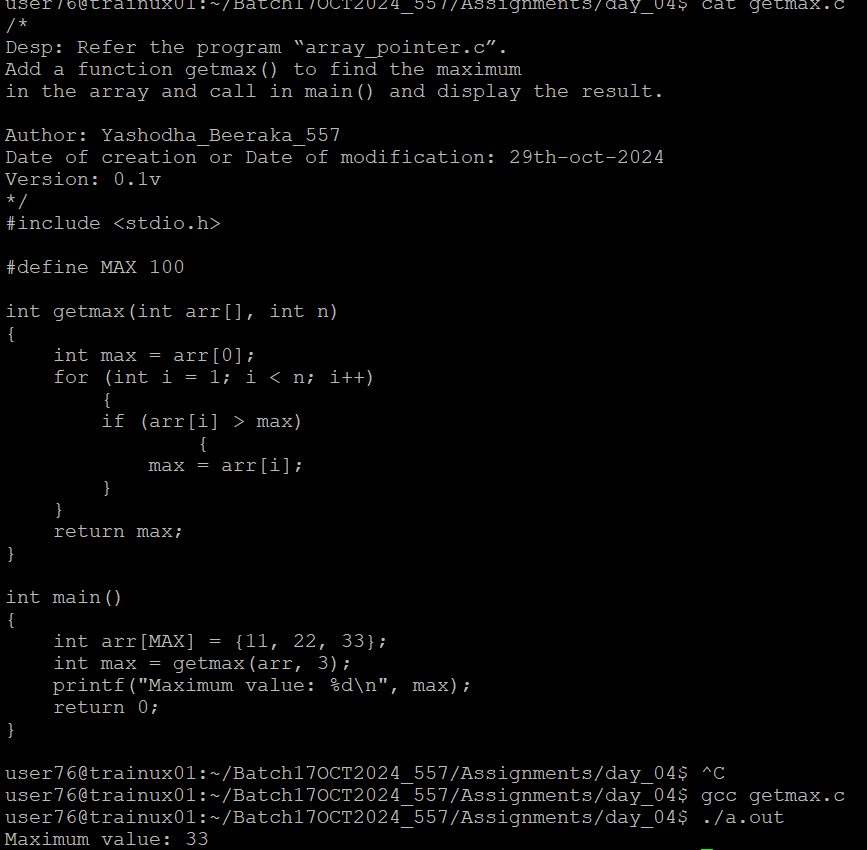
}

Ans:

* arr[MAX]: Elements not explicitly initialized will be set to 0.
* arr1[MAX]: All elements are explicitly initialized to 0.
* arr2[MAX]: As a static array, elements not explicitly initialized will be set to 0.

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.

Ans:



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

Ans: A black screen with white text

Description automatically generated

**2D, MultiDimensional Arrays**

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

int main()

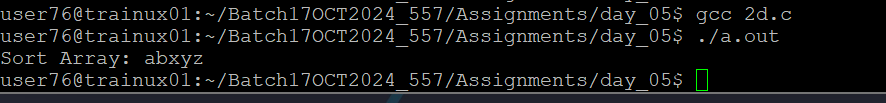
{

char arr[]= “xaybz”;

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

return 0;

}



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

Ans:

A computer screen with white text

Description automatically generated