**INTRODUCTION TO POINTERS**

**Mandatory**

1. Refer the code snippet below. int main()

{

char arr=”hello hi “;

int \*ptr = arr;

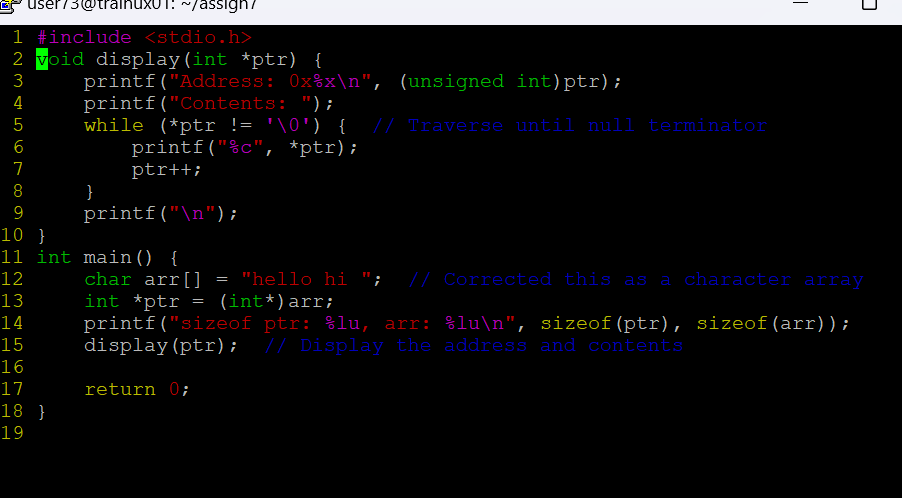
printf(“sizeof ptr:%d, arr:%d”, sizeof(ptr), sizeof(arr));

display(ptr); // display the address in hex and contents using pointer

}

Perform the following.

1. Implement the display() function (Use the “0x%x” formatting specifier to print addresses in hexadecimal.)
2. comment on the sizeof(ptr) and sizeof(arr)



A black background with white text

Description automatically generated

1. Refer the code snippet below. int main()

#define MAX 100

#define SUCCESS 0

#define FAILURE 1

int main()

{

char arr[MAX] = “Learning C“;

char\*ptr = arr;

char appendstr[3]= “in my org”;

printf(“Address of ptr:%x”, ptr);

int ret = append(ptr, appendstr);// append the string

printf(“Address of ptr:%x”, ptr);

if (ret == SUCCESS)

{

display(ptr); // display the address in hex and contents using pointer

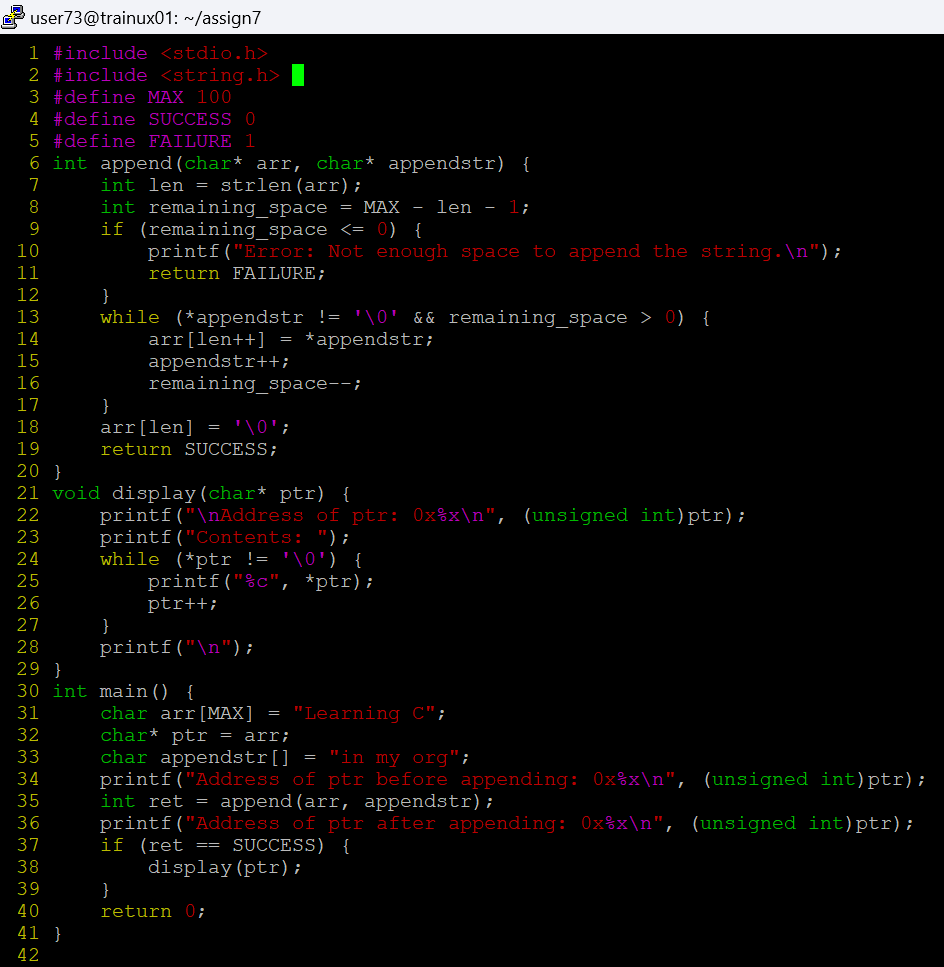
}

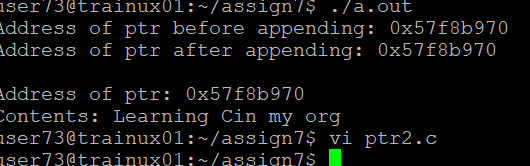
}

Perform the following.

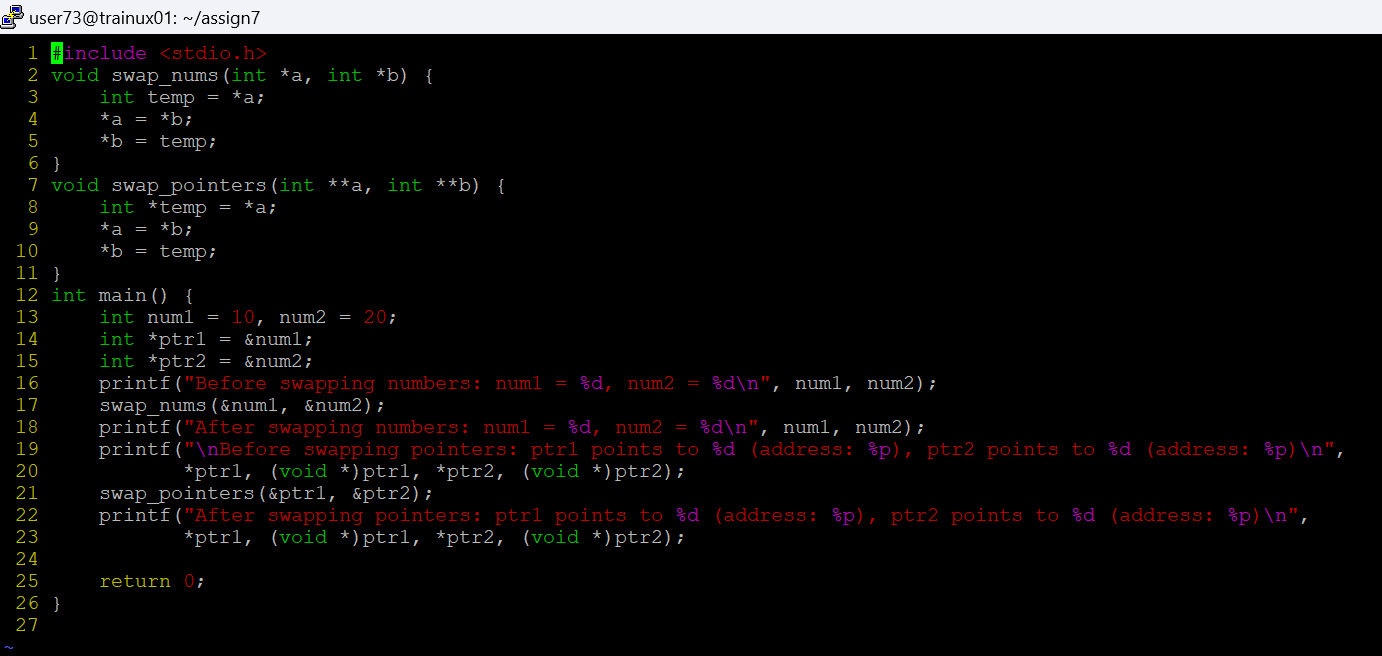
1. Implement the append() function to append the contents of the appendstr[] to arr using pointer.

[Note: append() should only use its content and not manipulate it. Contents should be retained even after the call]





1. Refer the code in “pointer\_prg.c”. The functions swap\_nums() and swap\_pointers() are expected to swap the numbers and pointers respectively. But swap\_pointers() is currently not giving the expected results. Analyse and the fix the issue.



Here initially there is no change in address because the values are only passed and changed not the address. To swap the address we need to pass by reference for that we need to take a double pointer to catch the address, then we get the required result

A black background with colorful text

Description automatically generated with medium confidence