**Introduction to Pointers Assignments**

Mandatory

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

{

char arr=”hello hi “;

int \*ptr = arr;

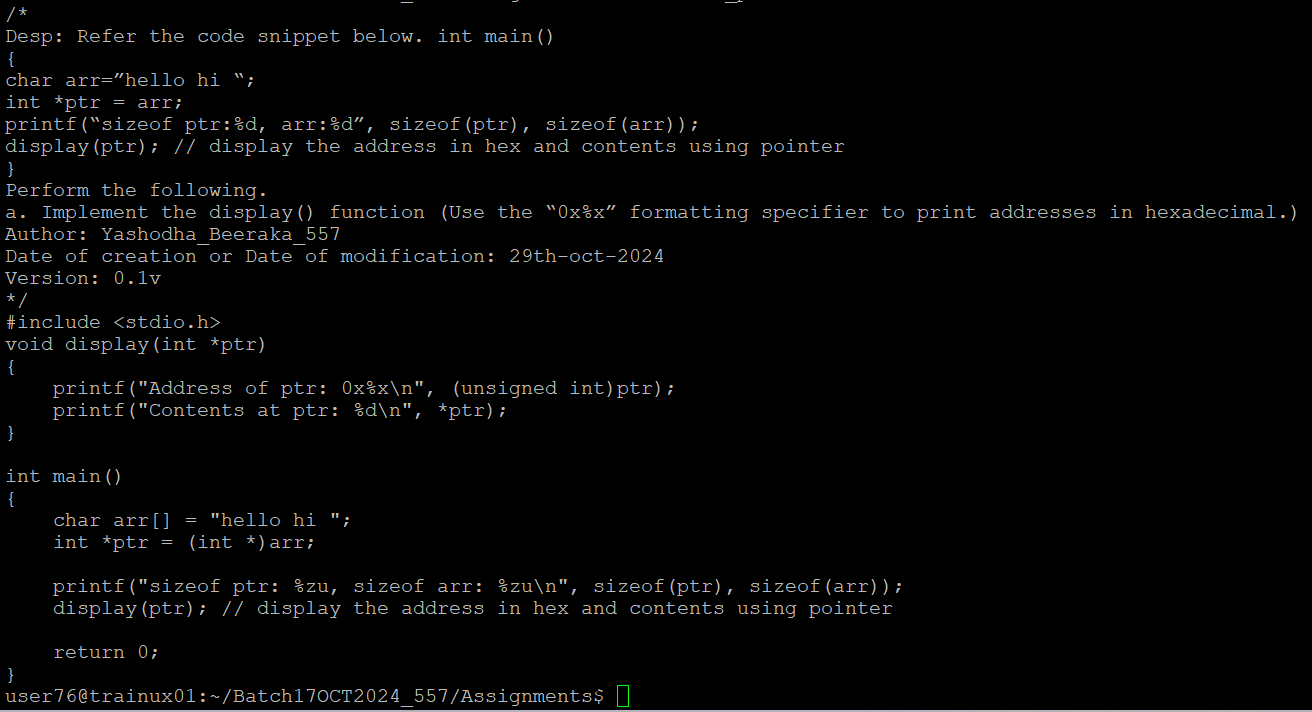
printf(“sizeof ptr:%d, arr:%d”, sizeof(ptr), sizeof(arr));

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

}

Perform the following.

a. Implement the display() function (Use the “0x%x” formatting specifier to print addresses in hexadecimal.)



b. comment on the sizeof(ptr) and sizeof(arr)

**Ans:**

sizeof(ptr): This will give the size of the pointer ptr. On most systems, this will be 8 bytes on a 64-bit system and 4 bytes on a 32-bit system. This is because ptr is a pointer to an integer, and the size of any pointer is determined by the architecture of the system (64-bit or 32-bit).

sizeof(arr): This will give the size of the array arr. Since arr is an array of characters with the string "hello hi ", it includes the null terminator \0 at the end. Therefore, the size will be 10 bytes (9 characters + 1 null terminator).

2. 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.

a. 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]

**Ans:** The append function will concatenate appendstr to the string pointed to by ptr. It will return SUCCESS if the operation is successful and FAILURE if there is an error (e.g., if the resulting string would exceed the maximum size). The display function will print the address and contents of the pointer.

3. 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.

**Ans:** the pointers by reference (i.e., pass pointers to the pointers).