# Function Parameter Passing and Return Assignments

**Mandatory**

1. Refer the code below and find the issue.

#include<stdio.h>

int \*func(void);

int main()

{

int num,\*ptr = NULL;

ptr = (int \*)func();

num = \*ptr;

return 1;

}

int \*func()

{

int local;

local = 10;

return(&local);

}

In above code is there a way(s) to return local variable address to caller?

ANS:  
The issue with code is trying to return the address of a local variable from the function func(). Local variables are stored on the stack, and once the function returns, the stack frame is destroyed, making the address invalid. This leads to undefined behavior.

Int \*func()

{

int local;

local =10;

return(&local);

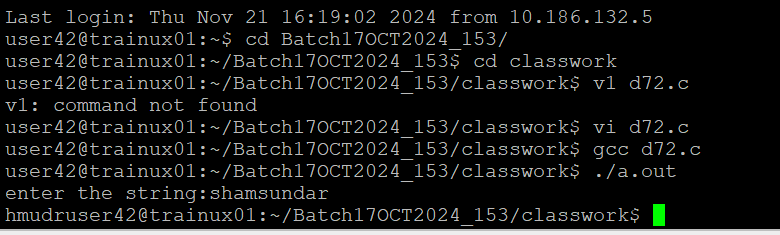
}

1. Write a program with a function ***read\_extract\_characters()*** to read a string (of max length 50 characters) from user, extract the characters at odd indices, store in an other array and return to the call. Caller should be able to read and display the extracted string.

[Note : do not return a local variable in function to caller]

A screenshot of a computer

Description automatically generated



1. Write below functions to extract and return the required inputs from an email id string of max length 80 characters. Program should be able to detect an invalid email id too based on following validations. Also value returned should be in scope and accessible in caller.
   1. valid email id will have username, host and domain name (as .com/.edu)

**Functions**:

*char \*getuser(char input[]); // return NULL or valid username from email id input*

*char \*gethost(char input[]); //return NULL or valid hostname from email id input*

*char \*getdomain(char input[]); //return NULL or valid domain name from email id input*

*int isValidDomain(char input[]); // return 1 if domain is “.com” or “.edu” else 0*

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A black screen with white text

Description automatically generated**