# Command Line Argument, Variable Argument Handling

1. Write a program to
   1. read a name(of max length 40 characters), ip address (as char \* string in dotted notation) and port number (unsigned short) of the cloud server as command line arguments.
   2. Validate if the required number of arguments have been received before proceeding. Else report error and return.
   3. Validate every argument received for valid range of values.

[Refer [ip address range](https://en.wikipedia.org/wiki/IP_address#:~:text=IPv4%20addresses%20are%20usually%20represented,%2C%20e.g.%2C%20172.16.254.1.), [port range](https://www.sciencedirect.com/topics/computer-science/registered-port#:~:text=Port%20numbers%20can%20run%20from,application%20processes%20on%20other%20hosts.) to do validations]

* 1. Store the values in a data structure and display using a function passing data structure

void display(struct server \*servercfg);

* 1. Implement a function update() to prompt user, to modify all the server attributes and to display the updated configuration.

// to read, update configuration and return status as SUCCESS/FAILURE

Int update(struct server \*servercfg);

* 1. Specify atleast 6 test cases (positive and negative ) to test command line inputs and update operations
  2. Check for memory leaks and fix them.

A screen shot of a computer

Description automatically generated

A:A screen shot of a computer program

Description automatically generated

A screen shot of a computer program

Description automatically generated

1. Implement a log() with signature as below to display all the input arguments as per their type. [Hint: In log() , use vfprintf() to display the received inputs]

void log(const char \*format, …);

For e.g.

int main()

{

int count = 10;

char prefix = ‘h’;

char label[] = “India”;

…

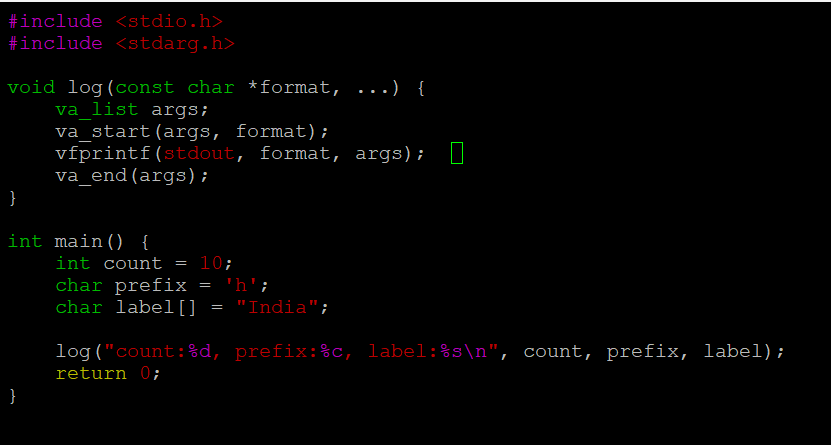
log(“count:%d, prefix:%c, label:%s”, count, prefix, label);

…

}

**Expected Output:**

count:10,prefix:h,label:India

A:

A black screen with white text

Description automatically generated

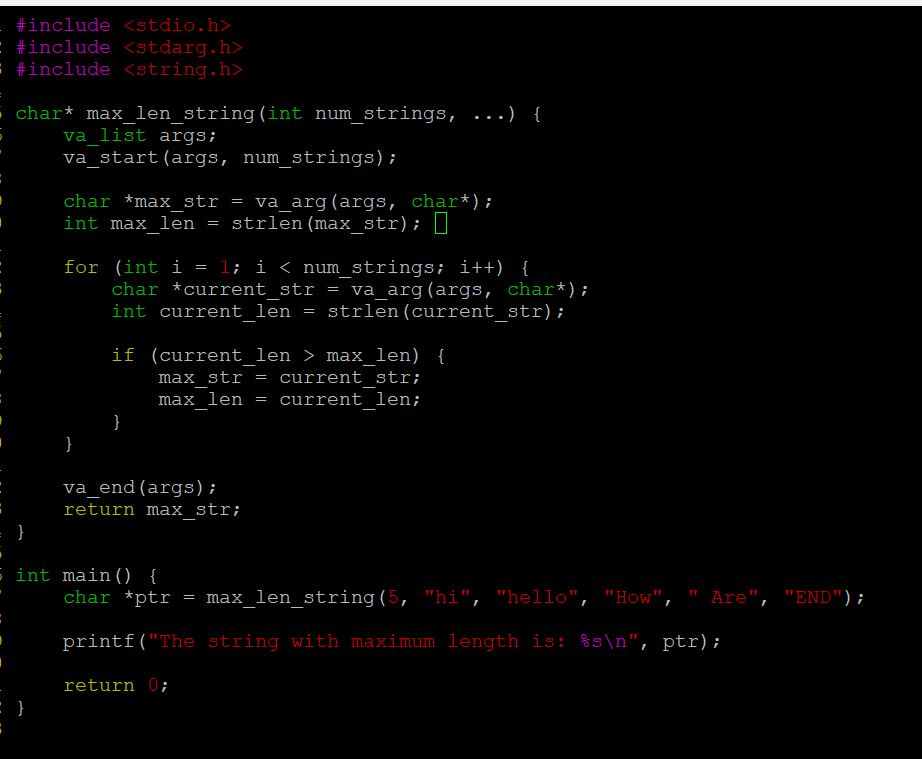
1. Refer the code “find\_max.c”. Add a function below to accept variable number of strings and to return the string with maximum length to the caller. In case of strings with same length, return the first string in the input sequence

max\_len\_string(<variable number of arguments>)

Eg. Code below shoud output “hello”

char \*ptr = max\_len\_string(“hi”, “hello”, “How”, “ Are”, “END”);

printf(“%s”, ptr);

A:

A black background with white text

Description automatically generated