

Documentation for CA304

Jamie Clarke

18398783

For this project we had to design an IP address calculator, a Subnet calculator and a Supernet calculator. Before beginning this we had to download FastAPI and uvicorn for testing the post requests.

IP Calculator

Functions:

The first function we had to design was an IP address calculator. We called this function ipcalc and its role was to take in a post request which contained an IP address in decimal dot notation. Once it received the Ip address from the post request, it then returned 5 different things:

- What class the IP address belonged to
- The number of networks available to this class of address
- The number of hosts available to this class of address
- The first IP address of this class
- The last IP address of this class

Inputs :

The inputs for this calculator were quite simple. All they involved were one IP address in the format of 4 numbers between the range 0-256 and a decimal point separating each of the 4 numbers.

Outputs:

The Output for this calculator depended on the IP address entered. These were:

The class which the IP address belonged to, The number of networks for the **class** of the address, The number of hosts for the **class** of the address, The first IP address for this class and finally the last IP address for this class. For

both class D and E, the number of networks and number of hosts, we returned "N/A" as there are no host addresses in these classes.

The class which the IP address belonged to

There are 5 different classes in which an IP address can belong to, ranging from A-E.

The number of networks for the **class** of the address

Depending on which class the IP address belonged to, this determined how many networks were available to the users. This was N/A in classes D and E due to them not having host addresses.

The number of hosts for the **class** of the address.

Depending on which class the IP address belonged to, this determined how many hosts were available to the users. This was N/A in classes D and E due to them not having host addresses.

The first IP address for this class.

Each class has its own first IP address which is always the same no matter what.

the last IP address for this class.

Each class has its own last IP address which will always stay the same.

Examples:

An example of an input would be the address 192.168.0.256

The screenshot shows a REST client interface with a light green header bar. The header bar contains the text "POST /ipcalc ipcalc" on the left and a small upward-pointing arrow on the right. Below the header bar is a section titled "Parameters" with a "Cancel" button in the top right corner. The "Parameters" section contains a table with two columns: "Name" and "Description". The table has one row with the following data:

Name	Description
address * required string (query)	192.168.0.256

At the bottom of the interface, there are two buttons: "Execute" (a blue button) and "Clear" (a white button with a grey border).

The output of this being entered is:

```
Response body
{
  "Class": "C",
  "num_networks": "16777214",
  "num_hosts": "254",
  "first_address": "192.0.0.0",
  "last_address": "223.255.255.255"
}
Response headers
```

As you can see this example address belongs to class C.
It has a total of 16777214 total networks available.
It has 254 total hosts available.
The first IP address of a class C is 192.0.0.0
And the last IP address of a class C is 223.255.255.255

Subnet Calculator

Functions:

The next function we needed to design was a subnet calculator which would take both an IP address and a mask. This function could only take a class B or C address as both D and E networks do not have a subnet mask. Once again we take the IP address in decimal dot notation. Once it receives both the Ip address and the mask from a post request, it must return six outputs, These are:

- The Ip address in CIDR notation
- The number of subnets on the network
- The number of addressable hosts per subnet
- The valid subnets
- The broadcast address of each subnet
- The valid hosts on each subnet

Inputs:

The inputs for this calculator included an IP address and a subnet mask. The subnet mask is used to divide the IP address up into 2 parts, one for identifying the host and the other for identifying the network it belongs to. The IP address that must be inputted is just in the normal format.(4 numbers between 0.0.0.0 - 256.256.256.256).

Outputs:

For the subnet calculator we had to return seven outputs. The Ip address followed by the cidr, the number of subnets, the addressable hosts per subnet, all the valid subnets, all the broadcast addresses and both the first and last addresses.

Address_cidr: This is just the ip address followed by a "/" followed by a number. This number represents the number of addresses in the range of IP addresses a network uses.

Num_subnets: This is found by counting the number of bits by which the initial mask was extended. This is also known as subnet bits.

Addressable_hosts_per_subnets: This is calculated by multiplying the number of subnet bits by 2 and then taking 2 from this number.

Valid_subnets: A valid subnet is one in which the address quoted is the lowest possible in the range. As stated in your notes we count up in blocks of 64 from 0 to get the valid subnets.

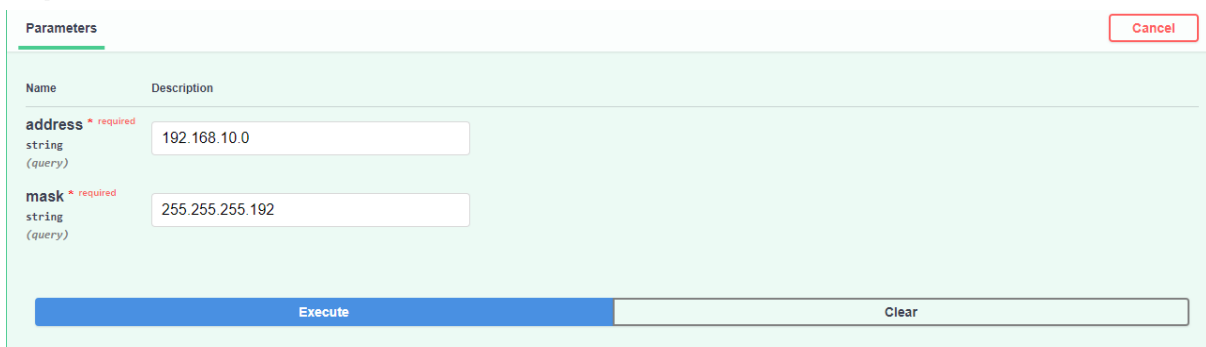
Broadcast_addresses: These are just the addresses before the next subnet.

First_addresses: These are just the first addresses of each subnet

Last_addresses: These are just the last addresses of each subnet.

Examples:

Input



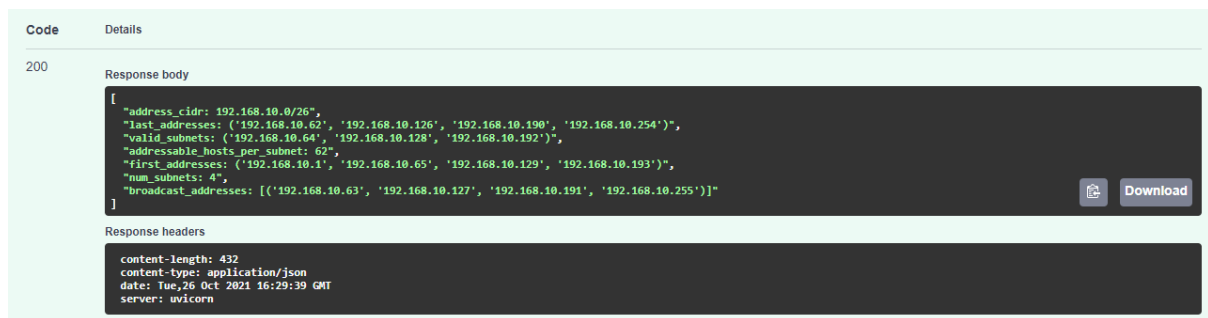
The screenshot shows a web-based form titled "Parameters" with a "Cancel" button in the top right corner. The form has two input fields: "address" and "mask". Both fields are marked as "required" with a red asterisk. The "address" field contains the value "192.168.10.0" and is labeled with "string" and "(query)". The "mask" field contains the value "255.255.255.192" and is also labeled with "string" and "(query)". At the bottom of the form, there are two buttons: "Execute" (in blue) and "Clear" (in light gray).

An example input IP address would be 192.168.10.0 and a mask of 255.255.255.192

I used these as a simple comparison as this is what's given as an example input in the specifications.

Output :

This is the output for the above entered



The screenshot shows a web interface with two tabs: 'Code' and 'Details'. The 'Code' tab is active, displaying a 200 status code and a 'Response body' containing a JSON object. The JSON object has the following structure:
- 'address_cidr': '192.168.10.0/26'
- 'last_addresses': an array of four IP addresses: ['192.168.10.62', '192.168.10.126', '192.168.10.190', '192.168.10.254']
- 'valid_subnets': an array of two CIDR notations: ['192.168.10.64', '192.168.10.128', '192.168.10.192']
- 'addressable_hosts_per_subnet': 62
- 'first_addresses': an array of four IP addresses: ['192.168.10.1', '192.168.10.65', '192.168.10.129', '192.168.10.193']
- 'num_subnets': 4
- 'broadcast_addresses': an array of four IP addresses: ['192.168.10.63', '192.168.10.127', '192.168.10.191', '192.168.10.255']
Below the JSON, the 'Response headers' are shown:
- content-length: 432
- content-type: application/json
- date: Tue, 26 Oct 2021 16:29:39 GMT
- server: unicorn
A 'Download' button is visible next to the JSON response body.

```
200

Response body

[
  "address_cidr: 192.168.10.0/26",
  "last_addresses: ('192.168.10.62', '192.168.10.126', '192.168.10.190', '192.168.10.254')",
  "valid_subnets: ('192.168.10.64', '192.168.10.128', '192.168.10.192')",
  "addressable_hosts_per_subnet: 62",
  "first_addresses: ('192.168.10.1', '192.168.10.65', '192.168.10.129', '192.168.10.193')",
  "num_subnets: 4",
  "broadcast_addresses: [('192.168.10.63', '192.168.10.127', '192.168.10.191', '192.168.10.255')]"
]

Response headers

content-length: 432
content-type: application/json
date: Tue, 26 Oct 2021 16:29:39 GMT
server: unicorn
```

Supernet Calculator

I was unable to get this working as I found this too difficult to understand.

Functions:

This calculator was to take a list of contiguous class C addresses which then needed to be supernetted. It would take a list of IP addresses and return just the address with the cidr notation and the supernet mask.

Input:

The input was just a list of IP addresses in the normal format of 4 numbers in the range 0.0.0.0 - 256.256.256.256. This would be contained within a string.

Output:

The function would return both an address and a subnet mask. The address would also contain the CIDR notation at the end.

Address:

This would be the IP address followed by a "/" and then the CIDR notation for this.

Mask:

This would be the supernet mask. These masks are a 32-bit number where all the fixed bits of the network are represented by 1s.