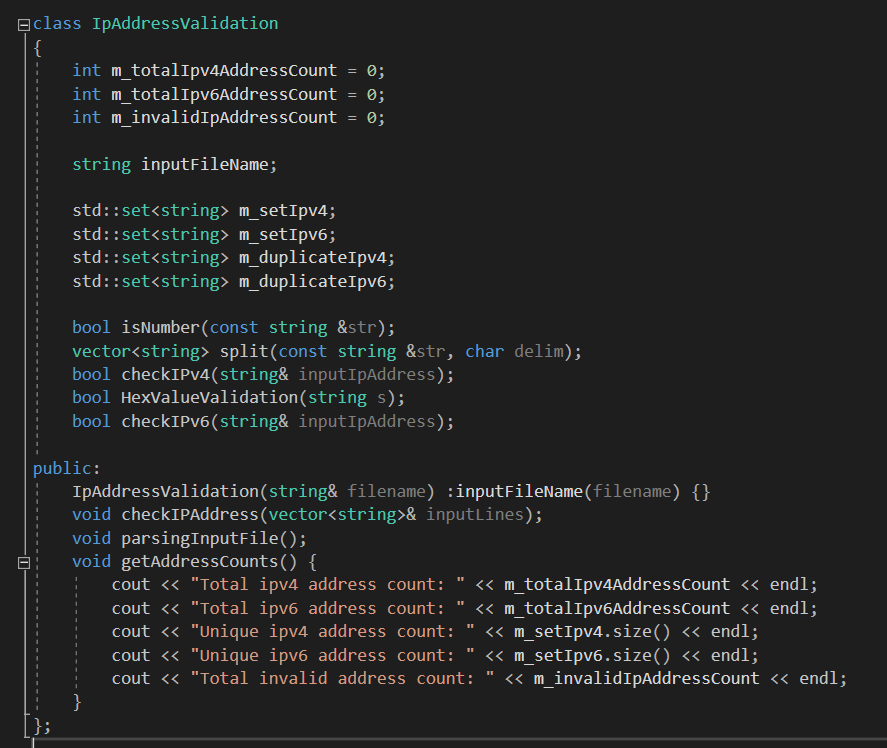
**IP Address Validation**

**Purpose:**There are millions of lines of IP addresses present in a file (input) which includes ipv4,ipv6,invalid addresses as well. There can be a chance the IP addresses can be redundant.   
My solution should be able to find out the total number of valid Ipv4,Ipv6 addresses, unique number of Ipv4 and Ipv6 addresses as well as invalid addresses.

**Class Design:**

1. Created a class named “**IpAddressValidation**” with various data members and member functions.
2. 

**Private Data Members and usage:**

1. int m\_totalIpv4AddressCount = 0;

To store total number of valid Ipv4 addresses.

1. int m\_totalIpv6AddressCount = 0;  
    To store total number of valid Ipv6 addresses.
2. int m\_invalidIpAddressCount = 0;

To store total number of invalid addresses.

1. string inputFileName;

To store the input file name.

1. std::set<string> m\_setIpv4;

set container to store all the unique Ipv4 addresses.

1. std::set<string> m\_setIpv6;

set container to store all the unique Ipv6 addresses.

1. std::set<string> m\_duplicateIpv4;

set container to store duplicate Ipv4 addresses.

1. std::set<string> m\_duplicateIpv6;

set container to store duplicate Ipv6 addresses.

**Private Member Functions and usage:**

1. bool isNumber(const string &str);   
   This function basically checks if an input string is a valid number or not.
2. vector<string> split(const string &str, char delim);  
   This function basically splits the entire input string based upon the input delimiter and put each value in a vector of string.
3. bool checkIPv4(string& inputIpAddress);  
   This function checks if the input string is a valid Ipv4 address or not.   
   There are two checks involved in this as below.
4. If each substring is a valid number or not?
5. If the number is less than equal to 255 or not?
6. bool HexValueValidation(string s);

This function checks if the input string is hexadecimal value or not by checking if each character is either 0-9 or A to F or a to f?

1. bool checkIPv6(string& inputIpAddress);

This function checks if the input string is a valid Ipv6 address or not.  
There are few checks involved in this as below.

1. If total number of substrings is 8 or not with delimiter ‘:’.
2. If total number of ‘:’ delimiter is 7 or not.
3. Each substring is an hexadecimal value or not.

**Public Member Functions and usage:**

1. IpAddressValidation(string& filename) :inputFileName(filename) {}

This is basically a constructor with input as the filename which we are going to parse through.

1. void parsingInputFile();  
   This function reads the input file, parse each line one by one and put in a vector until the vector size is 5000(configurable). Then it passes that vector with size 5000 to a method called checkIPAddess() and clears the vector. Will repeat the same until the end of the file.
2. void checkIPAddress(vector<string>& inputLines);  
   This function basically takes input a vector of size 5000 from parsingInputFiles() and then creates a thread. Each thread takes the vector and parse each ip address and checks for validity.
3. void getAddressCounts();  
   This function basically prints the number of valid and invalid IP addresses on the console.