**Code:**

|  |
| --- |
| #include <bits/stdc++.h>  using namespace std;  vector<string> split(const string &s, char delimiter)  {      vector<string> tokens;      string token;      istringstream tokenStream(s);      while (getline(tokenStream, token, delimiter))      {          tokens.push\_back(token);      }      return tokens;  }  int main()  {      string cidr;      string ip\_address;      cout << "Enter IP Address: " << endl;      cin >> ip\_address;      cout << "Enter the subnet mask in CIDR notation (e.g., /24):" << endl;      cin >> cidr;      // Split the IP address into octets      vector<string> ip\_octets = split(ip\_address, '.');      // Extract the CIDR prefix length      int cidr\_prefix\_length = stoi(cidr.substr(1));      // Calculate the subnet mask      uint32\_t subnet\_mask\_value = 0xFFFFFFFFU << (32 - cidr\_prefix\_length);      // Convert octets to integers      uint32\_t ip\_address\_value = (stoi(ip\_octets[0]) << 24) |                                  (stoi(ip\_octets[1]) << 16) |                                  (stoi(ip\_octets[2]) << 8) |                                  stoi(ip\_octets[3]);      // Calculate the network address      uint32\_t network\_address = ip\_address\_value & subnet\_mask\_value;      // Calculate the broadcast address      uint32\_t broadcast\_address = network\_address | (~subnet\_mask\_value);      int network\_octets[4];      int broadcast\_octets[4];      for (int i = 0; i < 4; ++i)      {          network\_octets[i] = (network\_address >> (24 - 8 \* i)) & 0xFF;          broadcast\_octets[i] = (broadcast\_address >> (24 - 8 \* i)) & 0xFF;      }      // Output the initial and end addresses      cout << "Initial Address: " << network\_octets[0] << "." << network\_octets[1] << "." << network\_octets[2] << "." << network\_octets[3] << endl;      cout << "End Address: " << broadcast\_octets[0] << "." << broadcast\_octets[1] << "." << broadcast\_octets[2] << "." << broadcast\_octets[3] << endl;      // cout << "Initial Address: " << network\_address<<endl;      // cout << "End Address: " << broadcast\_address<<endl;      return 0;  }  **Output:**  Enter IP Address:  201.1.1.0  Enter the subnet mask in CIDR notation (e.g., /24):  /27  Initial Address: 201.1.1.0  End Address: 201.1.1.31 |