

Scilab Manual for
Computer Communication Networks
by Dr Sujata Shekhar Kulkarni
Others
Spit Mumbai¹

Solutions provided by
Dr Sujata Shekhar Kulkarni
Others
Spit Mumbai

May 19, 2022

¹Funded by a grant from the National Mission on Education through ICT, <http://spoken-tutorial.org/NMEICT-Intro>. This Scilab Manual and Scilab codes written in it can be downloaded from the "Migrated Labs" section at the website <http://scilab.in>

Contents

List of Scilab Solutions	3
1 HAMMING CODE	5
2 IP CHECKSUM	9
3 SHORTEST PATH DISTANCE ALGORITHM	14
4 CALCULATE FIRST AND LAST ADDRESS OF GIVEN NETWORK	20
5 DESIGN OF SUBNET IN CLASSFUL IP ADDRESS	24

List of Experiments

Solution 1.1	Hamming code for error detection and correction	5
Solution 2.2	Calculation of IP Header Checksum	9
Solution 3.3	Study and calculate shortest path distance using Dijkstras Algorithm	14
Solution 4.4	To determine First address and Last address and Number of addresses in the block from any given classless address	20
Solution 5.5	To determine the Class and first and last address of the class Number of networks and Number of hosts for a given classful address	24

List of Figures

1.1	Hamming code for error detection and correction	7
1.2	Hamming code for error detection and correction	8
2.1	Calculation of IP Header Checksum	10
3.1	Study and calculate shortest path distance using Dijkstras Algorithm	19
4.1	To determine First address and Last address and Number of addresses in the block from any given classless address	23
4.2	To determine First address and Last address and Number of addresses in the block from any given classless address	23
5.1	To determine the Class and first and last address of the class Number of networks and Number of hosts for a given classful address	27
5.2	To determine the Class and first and last address of the class Number of networks and Number of hosts for a given classful address	28

Experiment: 1

HAMMING CODE

Scilab code Solution 1.1 Hamming code for error detection and correction

```
1 //Note: Details of scilab software version and OS
   version used:
2 //Tested on OS: Ubuntu 14.04 LTS, 64 bit
3 //Scilab version: 5.5.0 (Tested on 64 bit version)
4 //Program Title: Study and calculate HAMMING CODE
   GENERATION AND CORRECTION
5 clear;
6 clc;
7 clear all;
8 close;
9 x=input('Enter 8 bit data : ');
10 p=[0 0 0 0];
11 p1=[0 0 0 0];
12 error_pos=0;
13 d=[p(1) p(2) x(1) p(3) x(2) x(3) x(4) p(4) x(5) x(6)
     x(7) x(8)];
14 p(1)=modulo((d(1)+d(3)+d(5)+d(7)+d(9)+d(11)),2);
15 p(2)=modulo((d(2)+d(3)+d(6)+d(7)+d(10)+d(11)),2);
16 p(3)=modulo((d(4)+d(5)+d(6)+d(7)+d(12)),2);
17 p(4)=modulo((d(8)+d(9)+d(10)+d(11)+d(12)),2);
```

```

18 d=[p(1) p(2) x(1) p(3) x(2) x(3) x(4) p(4) x(5) x(6)
      x(7) x(8)];
19 disp('Hamming code is : ');
20 disp(d);
21 d1=input('Enter 12 bit received data : ');
22 p1(1)=modulo((d1(1)+d1(3)+d1(5)+d1(7)+d1(9)+d1(11))
              ,2);
23 p1(2)=modulo((d1(2)+d1(3)+d1(6)+d1(7)+d1(10)+d1(11))
              ,2);
24 p1(3)=modulo((d1(4)+d1(5)+d1(6)+d1(7)+d1(12)),2);
25 p1(4)=modulo((d1(8)+d1(9)+d1(10)+d1(11)+d1(12)),2);
26 //disp("p is:");
27 //disp(p);
28 for i=1:4
29     if p1(i)==1 then
30         error_pos= error_pos+(2^(i-1) );
31     end
32 end
33 disp('error is in bit position-');
34 disp(error_pos);
35
36 d2=bitxor(d,d1);
37 disp('correct codeword is-');
38 d3=bitxor(d1,d2);
39 disp(d3);
40
41
42
43 //INPUT
44 //Enter 8 bit data[1 1 0 0 1 1 0 0]
45 //Enter 12 bit received data[ 1 1 1 1 1 0 0 0 1 1 0
    0]
46
47 //OUTPUT
48 //Enter 8 bit data : [1 1 0 0 1 1 0 0 ]
49 //
50 // Hamming code is :
51 //

```

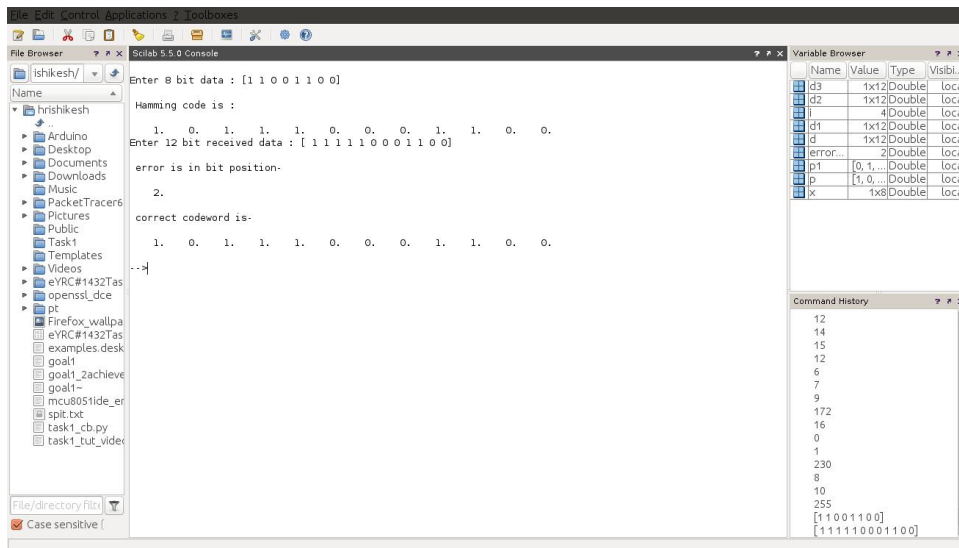


Figure 1.1: Hamming code for error detection and correction

```

52 //      1.      0.      1.      1.      1.      0.      0.      0.
           1.      1.      0.      0.
53 //Enter 12 bit received data : [1 1 1 1 1 0 0 0 1 1
           0 0 ]
54 //
55 // error is in bit position-
56 //
57 //      2.
58 //
59 // correct codeword is-
60 //
61 //      1.      0.      1.      1.      1.      0.      0.      0.
           1.      1.      0.      0.
62 //

```

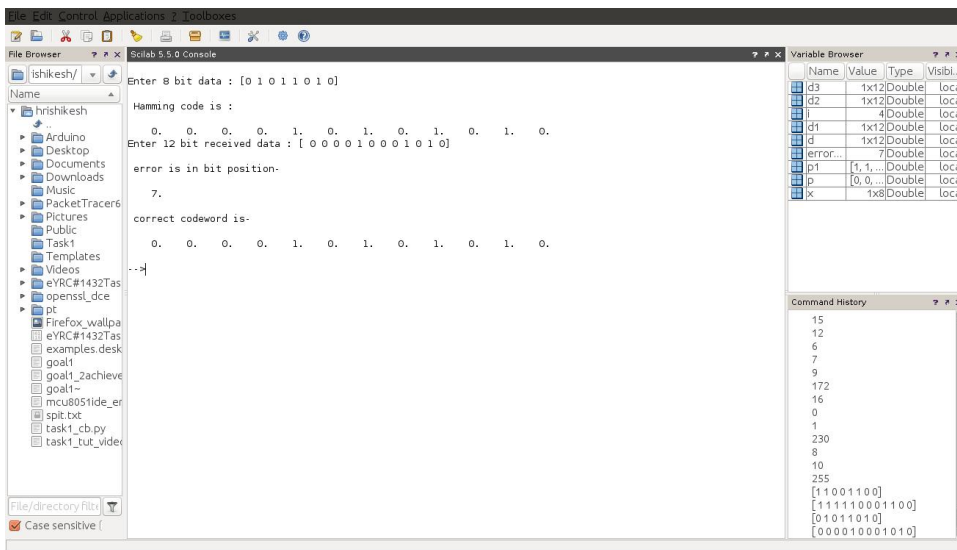


Figure 1.2: Hamming code for error detection and correction

Experiment: 2

IP CHECKSUM

Scilab code Solution 2.2 Calculation of IP Header Checksum

```
1 //Note: Details of scilab software version and OS
   version used:
2 //Tested on OS: Ubuntu 14.04 LTS, 64 bit
3 //Scilab version: 5.5.0 (Tested on 64 bit version)
4 //Program Title: HEADER CHECKSUM OF INTERNET
   PROTOCOL(IP)
5 clear;
6 clc;
7 ve = input('Enter the version number ')
8 hlen = input('Enter header length ')
9 ds = input('Enter DS ')
10 t1 = input('Enter the total length ')
11 inum = input('Enter the identification number ')
12 flag = input('Enter Flag bits ')
13 frag = input('Enter Fragmentation offset bits ')
14 ttl =input('Enter Time to Live bits')
15 prot = input('Enter Protocol Bits')
16 hech = input('Enter Header checksum bits ')
17 sip1 =input('Enter the 1st part of Source IP : ')
```

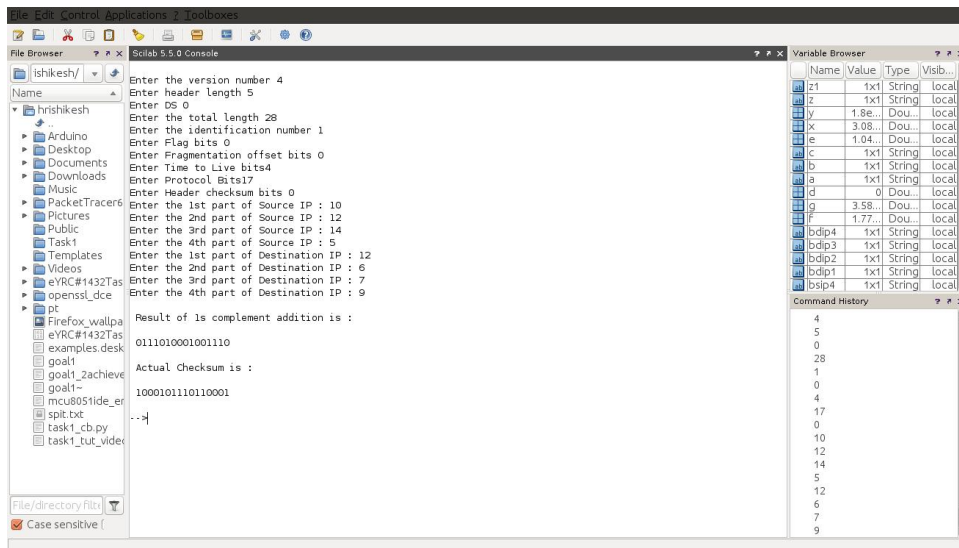


Figure 2.1: Calculation of IP Header Checksum

```

18 sip2 =input('Enter the 2nd part of Source IP : ')
19 sip3 =input('Enter the 3rd part of Source IP : ')
20 sip4 =input('Enter the 4th part of Source IP : ')
21 dip1 =input('Enter the 1st part of Destination IP :
    ')
22 dip2 =input('Enter the 2nd part of Destination IP :
    ')
23 dip3 =input('Enter the 3rd part of Destination IP :
    ')
24 dip4 =input('Enter the 4th part of Destination IP :
    ')
25
26 bver=dec2bin(ve,4);
27 bhlen=dec2bin(hlen,4);
28 bds=dec2bin(ds,8);
29 btl=dec2bin(tl,16);
30 binum=dec2bin(inum,16);
31 bflag=dec2bin(flag,3);
32 bfrag=dec2bin(frag,13);
33 btatl=dec2bin(ttl,8);

```

```

34 bprot=dec2bin(prot,8);
35 bhech=dec2bin(hech,16);
36 bsip1=dec2bin(sip1,8);
37 bsip2=dec2bin(sip2,8);
38 bsip3=dec2bin(sip3,8);
39 bsip4=dec2bin(sip4,8);
40 bdip1=dec2bin(dip1,8);
41 bdip2=dec2bin(dip2,8);
42 bdip3=dec2bin(dip3,8);
43 bdip4=dec2bin(dip4,8);
44
45 a=string(bver);
46 b=string(bhlen);
47 c=string(bds);
48 d=strcat([a,b,c]); //line 1
49
50 f=bin2dec(d); //line 1 dec
51 g=f+tl+inum+hech;
52 a=string(bflag);
53 b=string(bfrag);
54 c=strcat(a,b);
55
56 d=bin2dec(c);
57 a=string(bt1);
58 b=string(bprot);
59 c=strcat([a,b]);
60
61 e=bin2dec(c);
62 g=g+d+e;
63
64 a=string(bsip1);
65 b=string(bsip2);
66 c=strcat([a,b]);
67 x=bin2dec(c);
68 a=string(bsip3);
69 b=string(bsip4);
70 c=strcat([a,b]);
71 y=bin2dec(c);

```

```

72 g=g+x+y;
73
74 a=string(bdip1);
75 b=string(bdip2);
76 c=strcat([a,b]);
77 x=bin2dec(c);
78 a=string(bdip3);
79 b=string(bdip4);
80 c=strcat([a,b]);
81 y=bin2dec(c);
82
83 g=g+x+y;
84
85 z=dec2bin(g,16);
86 disp("Result of 1s complement addition is : ");
87 disp(z);
88 g=bitxor(g,65535);
89 z1=dec2bin(g,16);
90 disp("Actual Checksum is : ");
91 disp(z1);
92
93 //INPUT
94 //Enter the version number 4
95 //Enter header length 5
96 //Enter DS 0
97 //Enter the total length 28
98 //Enter the identification number 1
99 //Enter Flag bits 0
100 //Enter Fragmentation offset bits 0
101 //Enter Time to Live bits4
102 //Enter Protocol Bits17
103 //Enter Header checksum bits 0
104 //Enter the 1st part of sip10
105 //Enter the 2nd part of sip12
106 //Enter the 3rd part of sip14
107 //Enter the 4th part of sip5
108 //Enter the 1st part of dip12
109 //Enter the 2nd part of dip6

```

```
110 //Enter the 3rd part of dip7
111 //Enter the 4th part of dip9
112 //
113 // OUTPUT
114 //Result of 1s complement addition is :
      0111010001001110
115 //
116 // Actual checksum -1000101110110001
```

Experiment: 3

SHORTEST PATH DISTANCE ALGORITHM

Scilab code Solution 3.3 Study and calculate shortest path distance using Dijkstras Algorithm

```
1 //Note: Details of scilab software version and OS
  version used:
2 //Tested on OS: Ubuntu 14.04 LTS, 64 bit
3 //Scilab version: 5.5.0 (Tested on 64 bit version)
4 //Program Title: Study and calculate shortest path
  distance using Dijkstra's Algorithm
5 clear;
6 clc;
7 printf("B . .\n. . . .\n. . . .\n. . . .\n
  .\n. . . . A.....D\n. . . .\n. . . .\n
  . . . .\n. . . .\n. . . .\n. . . .\n
  .\n. . . .\n. . . .\n. . . .\n
  C.....E")
8 //Topology
9 // B . .
10 // . . . .
11 // . . . .
12 // . . . .
13 // . . . . A.....D
```

```

14 //      .   .   ..
15 //      .   .   .
16 //      .   .   .
17 //      .   .   .
18 //      ..  .   .
19 //      C.....E
20 // Here Node A is the source node
21 disp('Here Node A is the source node');
22 ds=input('select the destination node from b c d e :
          ','s'); //a is the source node
23 d1=input('enter the distance from a-b: '); //metric
    from respective node
24 d2=input('enter the distance from a-c: ');
25 d3=input('enter the distance from a-d: ');
26 d5=input('enter the distance from b-c: ');
27 d6=input('enter the distance from b-d: ');
28 d7=input('enter the distance from c-e: ');
29 d8=input('enter the distance from d-e: ');
30 d9=input('enter the distance from c-d: ');
31 if(ds=='b')
32     disp('There are 6 possible path and the shortest
          path is '); //paths available and the
          shortest path alongwith intermediate node for
          b
33     b1=d1;
34     b2=d2+d5;
35     b3=d3+d6;
36     b4=d3+d8+d7+d5;
37     b5=d3+d9+d5;
38     b6=d2+d9+d6;
39     if b1<b2& b1<b3& b1<b4& b1<b5& b1<b6
40         disp('Path from a to b');
41     else
42         if b2<b3& b2<b4& b2<b5& b2<b6
43             disp('Path from a to b via c');
44         else
45             if b3<b4& b3<b5& b3<b6
46                 disp('Path from a to b via d');

```



```

47         else
48             if b4<b5& b4<b6
49                 disp('Path from a to b via d - e - c
                    ');
50             else
51                 if b5<b6
52                     disp('Path from a to b via d - c');
53                 else
54                     disp('Path from a to b via c - d');
55                 end
56             end
57         end
58     end
59 end
60 end
61 if(ds=='c')
62     disp('There are 4 possible path and the shortest
        path is');//paths available and the shortest
        path alongwith intermediate node for c
63     c1=d2;
64     c2=d1+d5;
65     c3=d3+d9;
66     c4=d3+d8+d7;
67     if c1<c2& c1<c3& c1<c4
68         disp('Path from a to c');
69     else
70         if c2<c3& c2<c4
71             disp('Path from a to c via b');
72         else
73             if c3<c4
74                 disp('Path from a to c via d');
75             else
76                 disp('Path from a to c via d - e');
77             end
78         end
79     end
80 end
81 if(ds=='e')

```

```

82     disp('There are 5 possible path and the shortest
        path is ');//paths available and the shortest
        path alongwith intermediate node for e
83     e1=d3+d8;
84     e2=d2+d7;
85     e3=d1+d5+d7;
86     e4=d2+d9+d8;
87     e5=d3+d7+d9;
88     if e1<e2& e1<e3 & e1<e4& e1<e5
89         disp('Path from a to e via d');
90     else
91         if e2<e3 & e2<e4& e2<e5
92             disp('Path from a to e via c ');
93         else
94             if e3<e4& e3<e5
95                 disp('Path from a to e via b - c');
96             else
97                 if e4<e5
98                     disp('Path from a to e via c - d');
99                 else
100                     disp('Path from a to e via d - c');
101                 end
102             end
103         end
104     end
105 end
106 if(ds=='d')
107     disp('There are 6 possible path and the shortest
        path is ');//paths available and the
        shortest path alongwith intermediate node for
        b
108     x1=d3;
109     x2=d1+d6;
110     x3=d2+d9;
111     x4=d2+d7+d8;
112     x6=d1+d7+d5+d8;
113     x5=d1+d5+d9;
114     if x1<x2& x1<x3& x1<x4& x1<x5& x1<x6

```

```

115         disp('Path from a to d');
116     else
117         if x2<x3& x2<x4& x2<x5& x2<x6
118             disp('Path from a to d via b');
119         else
120             if x3<x4& x3<x5& x3<x6
121                 disp('Path from a to d via c');
122             else
123                 if x4<x5& x4<x6
124                     disp('Path from a to d via c - e ');
125                 else
126                     if x5<x6
127                         disp('Path from a to d via b - c');
128                     else
129                         disp('Path from a to d via b - c - e
130                             ');
131                     end
132                 end
133             end
134         end
135     end
136
137
138
139 //INPUT
140 //select the destination node from b c d e : e
141 //enter the distance from a-b: 1
142 //enter the distance from a-c: 2
143 //enter the distance from a-d: 1
144 //enter the distance from b-c: 1
145 //enter the distance from b-d: 2
146 //enter the distance from c-e: 1
147 //enter the distance from d-e: 2
148 //enter the distance from c-d: 1
149 //
150 //OUTPUT
151 // There are 5 possible path and the shortest path

```

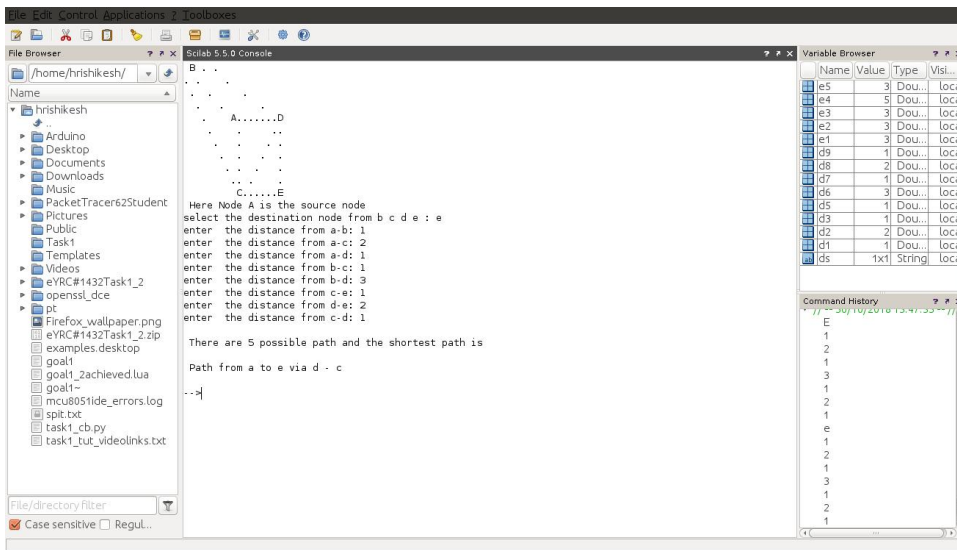


Figure 3.1: Study and calculate shortest path distance using Dijkstras Algorithm

```

152 // is
153 // Path from a to e via d - c

```

Experiment: 4

CALCULATE FIRST AND LAST ADDRESS OF GIVEN NETWORK

Scilab code Solution 4.4 To determine First address and Last address and Number of addresses in the block from any given classless address

```
1 //Note: Details of scilab software version and OS
   version used:
2 //Tested on OS: Ubuntu 14.04 LTS, 64 bit
3 //Scilab version: 5.5.0 (Tested on 64 bit version)
4 //Program Title: Classless Addressing: To determine
   First address, Last address and Number of
   addresses in the block from any given classless
   address.
5
6 clear ;
7 clc ;
8 ip1 =input('Enter the 1st part of ip : ')
9 ip2 =input('Enter the 2nd part of ip : ')
10 ip3 =input('Enter the 3rd part of ip : ')
11 ip4 =input('Enter the 4th part of ip : ')
12 n =8;
```

```

13 a3 = dec2bin (ip1 , n ) ;
14 a2 = dec2bin (ip2 , n ) ;
15 a1 = dec2bin (ip3 , n ) ;
16 a0 = dec2bin (ip4 , n ) ;
17
18 mask =input('Enter the prefix length/ CIDR :');
19
20 num_of_zeros =32 - mask ;
21 a = a3 + a2 + a1 + a0 ;
22 p1 = strsplit (a , mask ) ;
23 p = p1 (1) ;
24 for i = 1: num_of_zeros
25     p=p+ ' 0 ' ;
26 end
27 b = strsplit (p ,[8 16 24 ]);
28 b3 = bin2dec ( b (1) ) ;
29 b2 = bin2dec ( b (2) ) ;
30 b1 = bin2dec ( b (3) ) ;
31 b0 = bin2dec ( b (4) ) ;
32
33 printf ( "\n a)Dotted Decimal notation of first
        address :      %d . %d . %d . %d / %d" ,b3 , b2 ,
        b1 , b0 ,mask) ;
34
35 num_of_ones =32 - mask ;
36 a = a3 + a2 + a1 + a0 ;
37 p1 = strsplit (a , mask ) ;
38 p = p1 (1) ;
39 for i = 1: num_of_ones
40     p=p+ ' 1 ' ;
41 end
42 b = strsplit (p ,[8 16 24 ]);
43 b3 = bin2dec ( b (1) ) ;
44 b2 = bin2dec ( b (2) ) ;
45 b1 = bin2dec ( b (3) ) ;
46 b0 = bin2dec ( b (4) ) ;
47
48 printf ( "\n b)Dotted Decimal notation of last

```

```

        address : %d . %d . %d . %d / %d" ,b3 , b2 ,
        b1 , b0 ,mask) ;
49
50 num_of_addresses = 2^(32-mask);
51 printf ( "\n c) The number addresses is %d . " ,
        num_of_addresses ) ;
52
53
54 ///INPUT
55 //Enter the 1st part of ip : 167
56 //Enter the 2nd part of ip : 199
57 //Enter the 3rd part of ip : 170
58 //Enter the 4th part of ip : 82
59 //Enter the prefix length/ CIDR :27
60 //
61 //OUTPUT
62 //
63 //a)Dotted Decimal notation of first address :
        167 . 199 . 170 . 64 / 27
64 //b)Dotted Decimal notation of last address :      167
        . 199 . 170 . 95 / 27
65 //c) The number addresses is 32 .

```

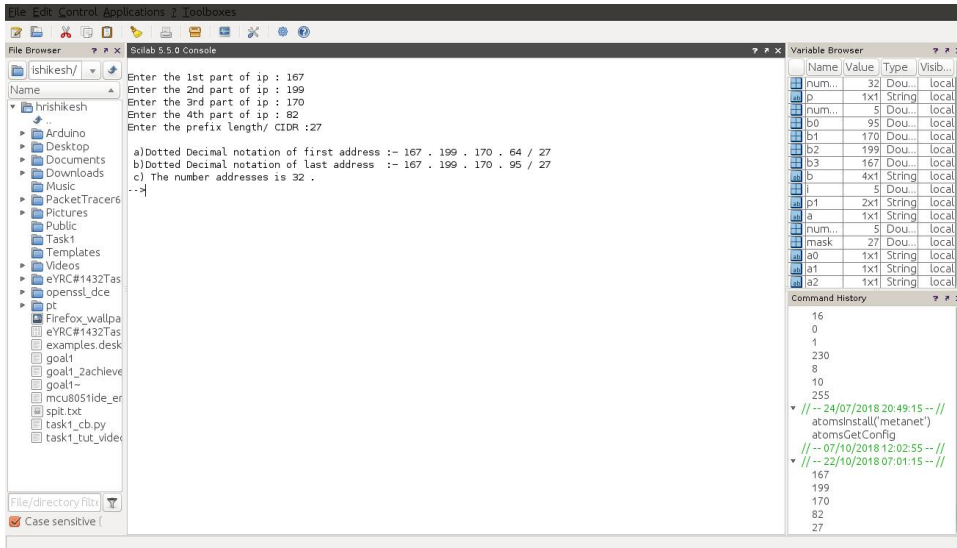


Figure 4.1: To determine First address and Last address and Number of addresses in the block from any given classless address

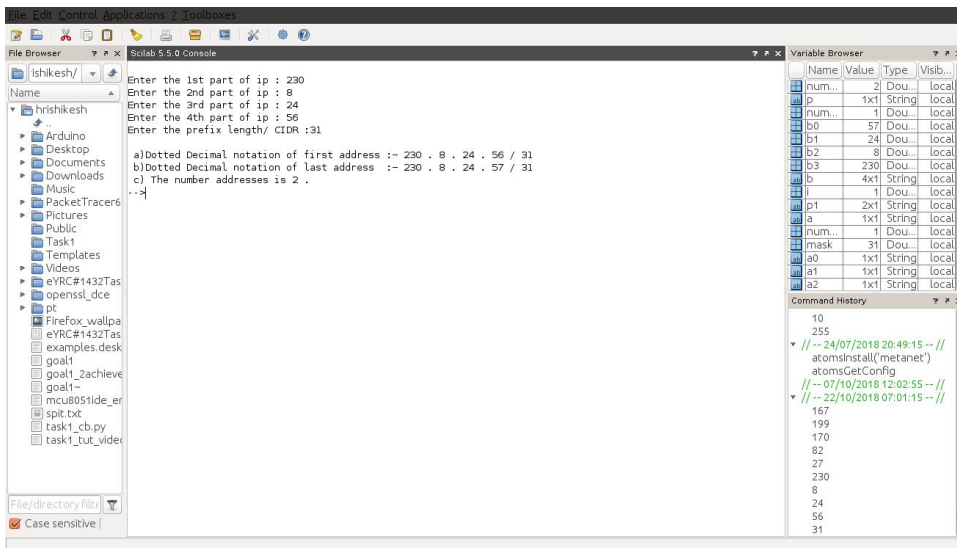


Figure 4.2: To determine First address and Last address and Number of addresses in the block from any given classless address

Experiment: 5

DESIGN OF SUBNET IN CLASSFUL IP ADDRESS

Scilab code Solution 5.5 To determine the Class and first and last address of the class Number of networks and Number of hosts for a given classful address

```
1 //Note: Details of scilab software version and OS
   version used:
2 //Tested on OS: Ubuntu 14.04 LTS, 64 bit
3 //Scilab version: 5.5.0 (Tested on 64 bit version)
4 //Program Title: Classful addressing: To determine
   the Class,1st & last address of the class , Number
   of networks and Number of hosts in the networks
   from any given classful address.
5 clear ;
6 clc ;
7 ip1 =input('Enter the 1st part of ip : ')
8 ip2 =input('Enter the 2nd part of ip : ')
9 ip3 =input('Enter the 3rd part of ip : ')
10 ip4 =input('Enter the 4th part of ip : ')
11 printf ( "Dotted Decimal notation of the IP address
   is :      %d . %d . %d . %d\n" ,ip1,ip2,ip3,ip4) ;
12
```

```

13 q=ip1;
14 n=0;
15 i=0;
16 if (q >=0 & q < 127) then
17     n=8;
18     i=1;
19     disp("The first byte is between 0 and 127.
           Therefore this is a Class A address.");
20     printf ("Network id: %d " ,ip1) ;
21     printf ("\nHost id: %d.%d.%d" ,ip2,ip3,ip4) ;
22     printf ("\nStart address : 0.0.0.0");
23     printf ("\nEnd address : 127.255.255.255");
24 elseif q==127 then
25     n=8;
26     i=1;
27     printf ("Network id: %d " ,ip1) ;
28     printf ("\nHost id: %d.%d.%d" ,ip2,ip3,ip4) ;
29     disp("The first byte is 127. Therefore it is a
           Class A address. This is used for Loopback
           addresses.");
30     printf ("\nStart address : 0.0.0.0");
31     printf ("\nEnd address : 127.255.255.255");
32 elseif (q >=128 & q <=191) then
33     n=16;
34     i=2;
35     disp("The first byte is between 128 and 191.
           Therefore this is a Class B address.");
36     printf ("Network id: %d.%d" ,ip1,ip2) ;
37     printf ("\nHost id: %d.%d" ,ip3,ip4) ;
38     printf ("\nStart address : 128.0.0.0");
39     printf ("\nEnd address : 191.255.255.255");
40 elseif ( q >=192 & q <=223) then
41     n=24;
42     i=3;
43     disp("The first byte is between 192 and 223.
           Therefore this is a Class C address.");
44     printf ("Network id: %d.%d.%d " ,ip1,ip2,ip3) ;
45     printf ("\nHost id: %d" ,ip4) ;

```

```

46     printf ("\nStart address : 192.0.0.0");
47     printf ("\nEnd address : 223.255.255.255");
48 elseif ( q >=224 & q <=239) then
49     disp("The first byte is between 224 and 239.
         Therefore this is a Class D address.");
50     printf ("\nStart address : 224.0.0.0");
51     printf ("\nEnd address : 239.255.255.255");
52 elseif (q >=240 & q <=255) then
53     disp("The first byte is between 240 and 255.
         Therefore this is a Class E address.");
54     printf ("\nStart address : 240.0.0.0");
55     printf ("\nEnd address : 255.255.255.255");
56 end
57
58
59 if n~=0 then
60     printf ("\nNumber of Networks %d" ,(2^(n-i))) ;
61     printf ("\nNumber of Hosts %d" ,(2^(32-n))-2) ;
62 end
63
64
65 //INPUT
66 //
67 //Enter the 1st part of ip : 172
68 //Enter the 2nd part of ip : 16
69 //Enter the 3rd part of ip : 0
70 //Enter the 4th part of ip : 1
71 //
72 //OUTPUT
73 //
74 //Dotted Decimal notation of the IP address is :
       172 . 16 . 0 . 1
75 //The first byte is between 128 and 191. Therefore
       this is a Class B address.
76 //Network id: 172.16
77 //Host id: 0.1
78 //Start address : 128.0.0.0
79 //End address : 191.255.255.255

```

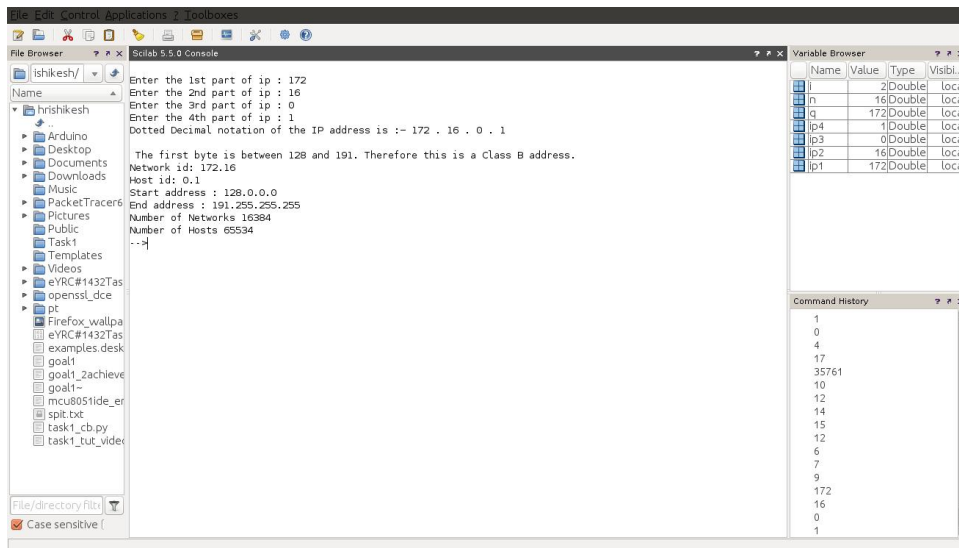


Figure 5.1: To determine the Class and first and last address of the class Number of networks and Number of hosts for a given classful address

```

80 //Number of Networks 16384
81 //Number of Hosts 65534
82 //

```

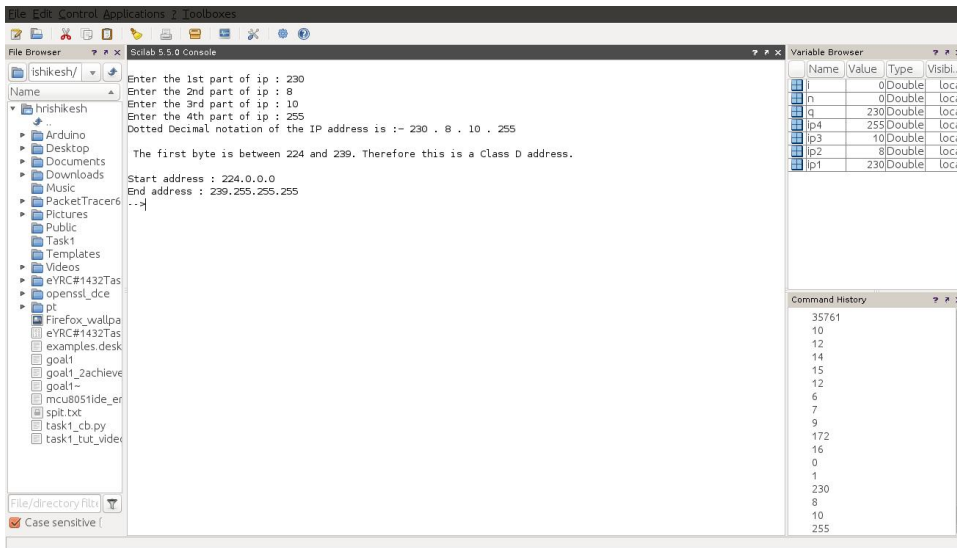


Figure 5.2: To determine the Class and first and last address of the class Number of networks and Number of hosts for a given classful address