INSTITUTO POLITÉCNICO NACIONAL
ESCUELA SUPERIOR DE CÓMPUTO

Redes de Computadoras

"Subnetting Program"

Abstract

A SUBNETTING PROGRAM helps the user to calculate the different addresses which can be generated by only introducing a netmask, a number of hosts, or a number of subnets required.

By:

Eduardo Alberto Pereda Guzmán

Luis

Professor:
MSc. NIDIA ASUNCIÓN CORTEZ DUARTE

October 2018

Index

Contenido

Introduction:

Nowadays, it is unacceptable and furthermore impossible to use an IP address only for a few users, it is necessary to have general knowledge about the usage of an IP address and how to optimize it in the most minimal part in order to be used to the fullest capacity. In this report, a tool is presented to reduce processing time and have subneted networks.

Literature review:

Ip classes

Clase	Rango total	Rango Privadas	Máscara
A	0.0.0.0 a 127.255.255.255	10.0.0.0	255.0.0.0 = /8
B	128.0.0.0 a 191.255.255.255	172.16.0.0 a 172.31.0.0	255.255.0.0 = /16
C	192.0.0.0 a 223.255.255.255	192.168.0.0 a 192.168.255.0	255.255.255.0 = /24
D	224.0.0.0 a 239.255.255.255		
E	240.0.0.0 a 255.255.255.255		

Firstly, every single IP (direction in a network) in a ipv4 configuration belongs to a different class, which are: Class A, Class B, Class C, Class D and Class E. Furthermore, is accounted for 4GB of networks in total, delivering 2GB for Class A, 1GB for Class B, 512MB for Class C, and 256MB for Class D and Class E. Only from Class A to Class c are public and for general use, on the other hand, Class D is used for multicast and Class D for experimental use. Lastly, an IP is conformed by 32 bits, divided into bytes, and only separated with '.'.

An IP has a network part and a host part, that determines how many networks you can find in a class with their respective size and number of hosts. In a computer, this is described by the netmask which every IP has.

A netmask is conformed by '1's and '0's. The '1's are the network part and the '0's are the host part. To add, a netmask is really helpful for computers to know where the network part begins and ends, therefore with the host part can be done either. Specifically, the netmasks cannot have '0's and '1's alternatively, first are the ones and then the ceros.

A network which belongs to any Class is conformed by a network IP, a broadcast IP and a range of hosts. There is a simple way to get the network IP and the broadcast IP, using the logic operator "or" and "and" to get each. In addition, a network IP and a broadcast can not be gotten in a Class D or Class E.

Subnetting

Subnetting is the strategy used to partition a single physical network into more than one smaller logical sub-networks (subnets). An IP address includes a network segment and a host segment. Subnets are designed by accepting bits from the IP address's host part and using these bits to assign a number of smaller sub-networks inside the original network. Subnetting allows an organization to add sub-networks without the need to acquire a new network number via the Internet service provider (ISP). Subnetting helps to reduce the network traffic and conceals network complexity. Subnetting is essential when a single network number has to be allocated over numerous segments of a local area network (LAN).

Subnets were initially designed for solving the shortage of IP addresses over the Internet.

How subnetting works

Each IP address consists of a subnet mask. All the class types, such as Class A, Class B and Class C include the subnet mask known as the default subnet mask. The subnet mask is intended for determining the type and number of IP addresses required for a given local network. The firewall or router is called the default gateway. The default subnet mask is as follows:

- Class A: 255.0.0.0
- Class B: 255.255.0.0
- Class C: 255.255.255.0

The subnetting process allows the administrator to divide a single Class A, Class B, or Class C network number into smaller portions. The subnets can be subnetted again into sub-subnets.

Network address (24 bits)	Subnet number (1 bit)	Extended network	Host address range
11000000 10101000 00000001	0	192.168.1.0	192.168.1.1 - 192.168.1.127
11000000 10101000 00000001	1	192.168.1.128	192.168.1.129 - 192.168.1.255

Figure 1 Subnetting with ip 192.168.1.0

Dividing the network into a number of subnets provides the following benefits:

Reduces the network traffic by reducing the volume of broadcasts

Helps to surpass the constraints in a local area network (LAN), for example, the maximum number of permitted hosts.

Enables users to access a work network from their homes; there is no need to open the complete network.

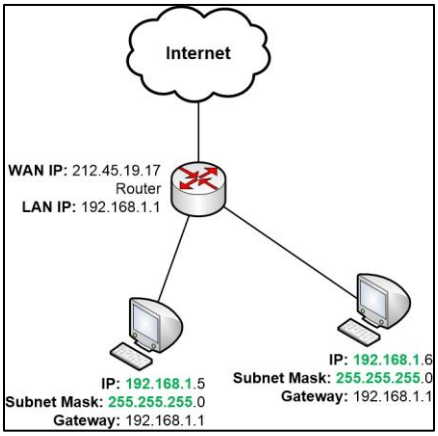
What is a subnet mask?

As with an IP address, a subnet mask comprises four bytes (32 bits) and is written in the same notation as an IP address, typically this is 255.255.255.0. For TCP/IP to work, you need a subnet mask.

The subnet mask complements an IP address and by applying it to the IP address and it determines what subnet an IP address belongs to. An IP address has two components, the network address and the host address. Subnetting further divides the host part of an IP address into a subnet and host address if additional subnetworks are needed. In effect, it masks an IP address and divides the IP address into network address and host address.

What is a default gateway?

When a computer on one network needs to communicate with a computer on another, it uses a router. A router specified on a host, which connects the host's subnet to other networks, is called a default gateway. This passes traffic on one subnet to devices on other subnets. This gateway often connects the local subnet to the internet.



Software (libraries, packages, tools):

*Language C

Libraries: #include<studio.h>

Compilator: GCC Compilator

Software: text-editor and windows cmd.

Procedure:

Procedure:

1. Gets an IP which obtains the class, the type, the Host Range, the Netmask and some other characteristics. (Check Ip Calculator for further information)
2. Depending on the class, chooses a specific function such as: CalculoC, CalculoB & Calculo A.
3. With a loop, calculates the Netmask.
4. Depending on the class, obtains the Net divisions and chooses the way how is going to be calculated each Subnet. Later, enters to a loop and shows on the screen a generated table including the ip Subnet, the Range Host and the Broadcast.
5. Taking in account the amount of Subnets, the program shall ask if the user wants to continue the process.
6. Lastly, asks the user if another calculation will be done.

Results

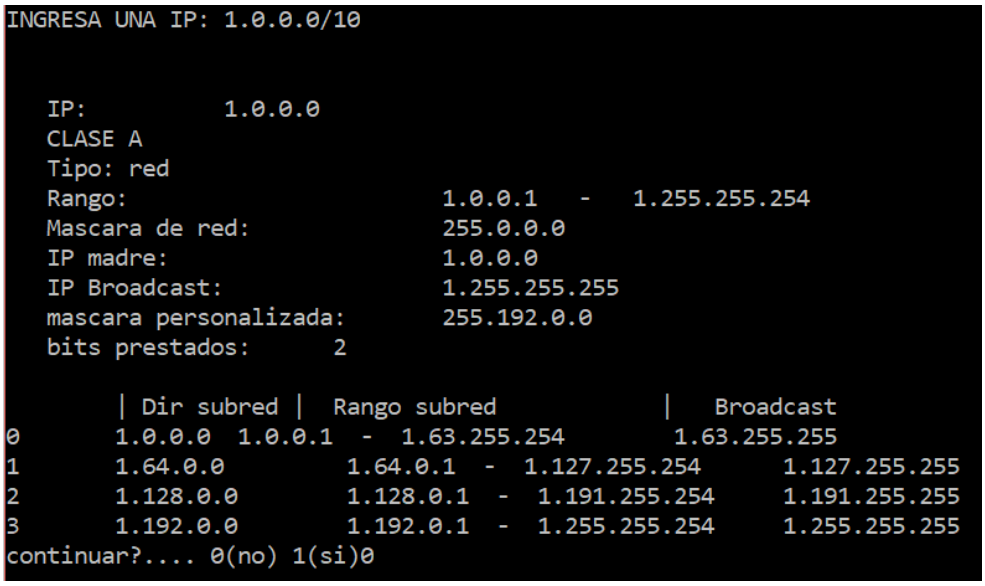


Figure 2 Subnetting IP: 1.0.0.0 Netmask: 255.192.0.0

```
INGRESA UNA IP: 130.0.0.0/17


   IP:          130.0.0.0
   CLASE B
   Tipo: red
   Rango:                       130.0.0.1   -   130.0.255.254
   Mascara de red:              255.255.0.0
   IP madre:                    130.0.0.0
   IP Broadcast:                130.0.255.255
   mascara personalizada:       255.255.128.0
   bits prestados:      1

        | Dir subred |  Rango subred            |   Broadcast
0       130.0.0.0        130.0.0.1  -  130.0.127.254     130.0.127.255
1       130.0.128.0      130.0.128.1  -  130.0.255.254   130.0.255.255
continuar?.... 0(no) 1(si)
```

Figure 3 Subnetting IP: 130.0.0.0 Netmask: 255.255.128.0

```
INGRESA UNA IP: 200.0.0.0/26


   IP:          200.0.0.0
   CLASE C
   Tipo: red
   Rango:                       200.0.0.1   -   200.0.0.254
   Mascara de red:              255.255.255.0
   IP madre:                    200.0.0.0
   IP Broadcast:                200.0.0.255
   mascara personalizada:       255.255.255.192
   bits prestados:      2

        | Dir subred |  Rango subred            |   Broadcast
0       200.0.0.0        200.0.0.1  -  200.0.0.62         200.0.0.63
1       200.0.0.64       200.0.0.65  -  200.0.0.126        200.0.0.127
2       200.0.0.128      200.0.0.129  -  200.0.0.190       200.0.0.191
3       200.0.0.192      200.0.0.193  -  200.0.0.254       200.0.0.255
continuar?.... 0(no) 1(si)
```

Figure 4 Subnetting IP: 200.0.0.0 Netmask: 255.255.255.192

Discussion:

The results show that the any network IP and any broadcast IP can be calculated with a simple logic operation, furthermore, no memory is wasted and unnecessary computer processing is used.

Also, the netmask is fundamental in order to obtain the different ways how an IP is divided, making different subnets within range hosts and Broadcasts IP.

Finally, this program is of main importance because it has the algorithm to divide the networks, nowadays something really important.

Conclusions:

Eduardo Pereda: In this program I learnt how a network can be divided in order to share it with other sections or departments.
In the moment that the program was made, there could be possible errors in the network dividing, so we had to consider different cases related with the netmask; this cases helped to program to take a decision in which algorithm will take to divide the network.

References:

Cortes Duarte, N. (2018). Class Summary. [Text] Instituto Politecnico Nacional ESCOM, Mexico City.

Techopedia.com. (2018). What is Subnetting? - Definition from Techopedia. [online] Available at: https://www.techopedia.com/definition/28328/subnetting [Accessed 2 Oct. 2018].

Millman, R. (2018). What is subnetting?. [online] IT PRO. Available at: http://www.itpro.co.uk/network-internet/31770/what-is-subnetting [Accessed 2 Oct. 2018].

Code:

```c
#include <stdio.h>
#include<math.h>

void RPredef(unsigned char[], unsigned char);
void calculoC(unsigned char[], unsigned char);
void calculoB(unsigned char[], unsigned char);
void calculoA(unsigned char[], unsigned char);

void calculoC(unsigned char ip[], unsigned char mp)
{
    unsigned char i, rd;
    unsigned char mr;
    mr=0;

    mp=mp-24;
    for(i=0; i <= mp; i++)//MP para clase C
        mr=mr + pow(2, 8-i);
    printf("   mascara personalizada:   255.255.255.%d\n", mr);
    printf("   bits prestados:  %d\n\n", mp);

    mp=pow(2, mp);//divisor de red

    rd= 255/mp;//divicion total de la red
    ip[3]=0;
    printf("    | Dir subred |  Rango subred        |   Broadcast\n");
    for (i = 0; i < mp; i++)
    {//192.1.12.25/26
        printf("%d  ", i);
        printf("%d.%d.%d.%d ", ip[0], ip[1], ip[2], ip[3]);
        printf("%d.%d.%d.%d  -  ", ip[0], ip[1], ip[2], (ip[3]+1));
        ip[3]=ip[3]+rd;
        printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]-1);
        printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
        ip[3]++;
    }
}
void calculoB(unsigned char ip[], unsigned char mp)
{
    unsigned char j, rd;
    unsigned char mr;
    int cont, i;
    mr=0;
    cont=0;

    mp=mp-16;
    if(mp>7)//MP para la clase B
    {
        for(i=0; i <= mp-8; i++)
            mr=mr + pow(2, 8-i);
        printf("   mascara personalizada:   255.255.255.%d\n", mr);
        printf("   bits prestados:  %d\n\n", mp);
        printf("    | Dir subred |  Rango subred        |   Broadcast\n");
        mp=pow(2, mp-8);//divisor de red
        rd= 255/mp;//divicion total de la red
        ip[2]=0;
        ip[3]=0;

        for (i=0; i<256; i++)
        {//190.1.12.25/26
```

```
            for (j = 0; j < mp; j++)
            {
                printf("%d  ", cont); cont++;
                printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]);
                printf("%d.%d.%d.%d  -  ", ip[0], ip[1], ip[2], (ip[3]+1));
                ip[3]=ip[3]+rd;
                printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]-1);
                printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
                ip[3]++;
            }
            ip[3]=0;
            ip[2]++;
            mr=getch();
        }/**/
    }
    else//   .../(<23)
    {
        for(i=0; i<=mp; i++)
            mr=mr + pow(2, 8-i);
        printf("   mascara personalizada:   255.255.%d.0\n", mr);
        printf("   bits prestados:  %d\n\n", mp);
        mp=pow(2, mp);//divisor de red
        rd= 255/mp;//divicion total de la red
        ip[2]=0;
        ip[3]=0;
        printf("    | Dir subred |  Rango subred        |   Broadcast\n");
        for (i = 0; i < mp; i++)
        {//192.1.12.25/26
            printf("%d  ", i);
            printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]);
            printf("%d.%d.%d.%d  -  ", ip[0], ip[1], ip[2], ip[3]+1);
            ip[2]=ip[2]+rd;
            printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]+254);
            printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]+255);
            ip[2]++;
        }
    }
}
void calculoA(unsigned char ip[], unsigned char mp)
{
           unsigned char rd;
           unsigned char mr;
           int cont, i, k, j;
           mr=0;
           cont=0;

           mp=mp-8;
           if (mp>15)
           {
               for(i=0; i <= mp-16; i++)
                   mr=mr + pow(2, 8-i);
               printf("   mascara personalizada:   255.255.255.%d\n", mr);
               printf("   bits prestados:  %d\n\n", mp);
               printf("    | Dir subred |  Rango subred        |   Broadcast\n");

               mp=pow(2, mp-16);//divisor de red
               rd= 255/mp;//divicion total de la red
               ip[1]=0;
               ip[2]=0;
               ip[3]=0;

```

```
120.                for (i=0; i<256; i++)
121.                {
122.                    for (j=0; j<256; j++)
123.                    {//12.1.12.25/26
124.                        for (k=0; k<mp; k++)
125.                        {
126.                            printf("%d  ", cont); cont++;
127.                            printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]);
128.                            printf("%d.%d.%d.%d  -  ", ip[0], ip[1], ip[2], ip[3]+
    1);
129.                            ip[3]=ip[3]+rd;
130.                            printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]-
    1);
131.                            printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
132.                            ip[3]++;
133.                        }
134.                        ip[3]=0;
135.                        ip[2]++;
136.                        mr=getch();
137.                    }
138.                    ip[2]=0;
139.                    ip[1]++;
140.                    printf("\n\n\n rd= %d\n\n", rd);
141.                    mr=rd;
142.                    printf("++++++++++++++++++++++++++++++++++++++++++++++++++++
    ++++++++++++++++++\nterminar: no(0) si(1)");
143.                        scanf("%d", &mr);
144.                    printf("\n\n\n rd= %d\n\n", rd);
145.                    if(mr)
146.                        i=255;
147.                }
148.            }/**/
149.            else if(mp>7)//MP para la clase A
150.            {
151.                for(i=0; i <= mp-8; i++)
152.                    mr=mr + pow(2, 8-i);
153.
154.                printf("   mascara personalizada:   255.0.%d.0\n", mr);
155.                printf("   bits prestados:  %d\n\n", mp);
156.                printf("    | Dir subred |  Rango subred        |   Broadcast\n");
157.                mp=pow(2, mp-8);//divisor de red
158.                rd= 255/mp;//divicion total de la red
159.                ip[1]=0;
160.                ip[2]=0;
161.                ip[3]=0;
162.
163.                for (i=0; i<256; i++)
164.                {//12.1.12.25/26
165.                    for (j = 0; j < mp; j++)
166.                    {
167.                        printf("%d  ", cont); cont++;
168.                        printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]);
169.                        printf("%d.%d.%d.%d  -  ", ip[0], ip[1], ip[2], ip[3]+1);
170.                        ip[2]=ip[2]+rd;
171.                        printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]+254);
172.                        printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]+255);
173.                        ip[2]++;
174.                    }
175.                    ip[2]=0;
```

```
                ip[1]++;
                mr=getch();

            }/**/
        }
        else//   .../(<15)
        {
            for(i=0; i<=mp; i++)
                mr=mr + pow(2, 8-i);
            printf("   mascara personalizada:   255.%d.0.0\n", mr);
            printf("   bits prestados:  %d\n\n", mp);

            mp=pow(2, mp);//divisor de red
            rd= 255/mp;//divicion total de la red
            ip[1]=0;
            ip[2]=0;
            ip[3]=0;
            printf("    | Dir subred |  Rango subred        |   Broadcast\n");

            for (i = 0; i < mp; i++)
            {//192.1.12.25/26
                printf("%d  ", i);
                printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2], ip[3]);
                printf("%d.%d.%d.%d  -  ", ip[0], ip[1], ip[2], ip[3]+1);
                ip[1]=ip[1]+rd;
                printf("%d.%d.%d.%d  ", ip[0], ip[1], ip[2]+255, ip[3]+254);
                printf("%d.%d.%d.%d\n", ip[0], ip[1], ip[2]+255, ip[3]+255);
                ip[1]++;
            }
        }
    }

    void RPredef(unsigned char ip[], unsigned char mp)
    {
        unsigned char mr[4], tipo;

        mr[0]=255;
        mr[1]=255;
        mr[2]=255;
        mr[3]=255;

        printf("\n\n   IP:      %d.%d.%d.%d\n", ip[0], ip[1], ip[2], ip[3]);
        if(ip[0] & 128)
        {
            if(ip[0] & 64)
            {
                if(ip[0] & 32)
                {
                    if(ip[0] & 16)
                    {
                        printf("   CLASE E\n");
                        mr[0]=0;
                        tipo=3;
                    }
                    else
                    {
                        printf("   CLASE D\n");
                        mr[0]=0;
                        tipo=3;
                    }
                }
```

```
236.                    else
237.                    {
238.                        printf("   CLASE C\n");
239.                        mr[3]=0;
240.                        printf("   Tipo: ");
241.                            if(ip[3]==0)//
242.                                printf("red\n");
243.                            else if(ip[3]==255)//
244.                                printf("Broadcast\n");
245.                            else//
246.                                printf("host\n");
247.                        printf("   Rango:           %d.%d.%d.%d   -   ", (ip[0]),
    (ip[1]), (ip[2]), ((ip[3] & mr[3]) + 1));
248.                        printf("%d.%d.%d.%d \n", (ip[0]), (ip[1]), (ip[2]), ((ip[3
    ] & mr[3]) + 254));
249.                        tipo=0;
250.                    }
251.                }
252.                else
253.                {
254.                    printf("   CLASE B\n");
255.                    mr[2]=0;
256.                    mr[3]=0;
257.                    printf("   Tipo: ");
258.                        if(ip[2]==0)//
259.                        {
260.                            if (ip[3]==0)
261.                                printf("red\n");
262.                        }
263.                        else if(ip[2]==255)//
264.                        {
265.                            if(ip[3]==255)
266.                                printf("Broadcast\n");
267.                        }
268.                        else//
269.                        {
270.                            printf("host\n");
271.                        }
272.                    printf("   Rango:           %d.%d.%d.%d   -   ", (ip[0]), (ip[
    1]), ((ip[2] & mr[2])), ((ip[3] & mr[3]) + 1));
273.                    printf("%d.%d.%d.%d \n", (ip[0]), (ip[1]), ((ip[2] & mr[2]) +
    255), ((ip[3] & mr[3]) + 254));
274.                    tipo=1;
275.                }
276.            }
277.            else
278.            {
279.                printf("   CLASE A\n");
280.                mr[1]=0;
281.                mr[2]=0;
282.                mr[3]=0;
283.                printf("   Tipo: ");
284.                    if(ip[1]==0)//
285.                    {
286.                        if (ip[2]==0)
287.                        {
288.                            if (ip[3]==0)
289.                                printf("red\n");
290.                        }
291.                    }
292.                    else if(ip[1]==255)//
```

```
293.                     {
294.                         if(ip[2]==255)
295.                         {
296.                             if(ip[3]==255)
297.                                 printf("Broadcast\n");
298.                         }
299.                     }
300.                     else//
301.                     {
302.                         printf("host\n");
303.                     }
304.                 printf("   Rango:           %d.%d.%d.%d   -   ", (ip[0]), ((ip[1]
    & mr[1])), ((ip[2] & mr[2])), ((ip[3] & mr[3]) + 1));
305.                 printf("%d.%d.%d.%d \n", (ip[0]), ((ip[1] & mr[1]) + 255), ((ip[2]
     & mr[2]) + 255), ((ip[3] & mr[3]) + 254));
306.                 tipo=2;
307.             }
308.             if(mr[0])
309.             {
310.                 printf("   Mascara de red:      %d.%d.%d.%d \n", mr[0], mr[1], mr[
    2], mr[3]);
311.                 printf("   IP madre:            %d.%d.%d.%d \n", (ip[0] & mr[0]),
    (ip[1] & mr[1]), (ip[2] & mr[2]), (ip[3] & mr[3]));
312.                 mr[0]=~mr[0];
313.                 mr[1]=~mr[1];
314.                 mr[2]=~mr[2];
315.                 mr[3]=~mr[3];
316.                 printf("   IP Broadcast:        %d.%d.%d.%d \n", (ip[0] | mr[0]),
    (ip[1] | mr[1]), (ip[2] | mr[2]), (ip[3] | mr[3]));
317.             }
318.             else
319.                 printf("   Es una IP reservada\n");
320.             switch(tipo)
321.             {
322.                 case 0:
323.                     if(!mp)
324.                         mp=24;
325.                     calculoC(ip, mp);
326.                     break;
327.                 case 1:
328.                     if(!mp)
329.                         mp=16;
330.                     calculoB(ip, mp);
331.                     break;
332.                 case 2:
333.                     if(!mp)
334.                         mp=8;
335.                     calculoA(ip, mp);
336.                     break;
337.                 default:
338.                     break;
339.             }/**/
340.         }
341.
342.         int main()
343.         {
344.             unsigned char ip[5];
345.             ip[0]=0;
346.             ip[1]=0;
347.             ip[2]=0;
348.             ip[3]=0;
```

```
349.            ip[4]=0;
350.            //printf("\nhola Nidia\nmi ip: 8.12.0.174\nmi mac: D8-CB-8A-D5-7D-
    38\n\nINGRESA UNA IP: ");
351.            printf("\nINGRESA UNA IP: ");
352.            scanf("%d.%d.%d.%d/%d", &ip[0], &ip[1], &ip[2], &ip[3], &ip[4]);
353.            if(!ip[0])
354.                ip[0]=1;
355.            if(ip[4]>30)
356.                ip[4]=0;
357.
358.            RPredef(ip, ip[4]);
359.
360.            printf("continuar?.... 0(no) 1(si)");
361.            scanf("%d", &ip[0]);
362.            if(ip[0])
363.                main();
364.            return 0;
365.        }
```