Preparador Informática

www.preparadorinformatica.com

PRÁCTICA 8 bis

PROGRAMACIÓN EN C y REDES

OPOSICIONES INFORMÁTICA 2018

ARAGÓN

PROGRAMACIÓN EN C y REDES.

(OPOSICIONES INFORMÁTICA ARAGÓN 2018)

3.- En una empresa, se han capturado datos de información de ip's en una red. El administrador de red, ha generado un fichero con los datos de las IP de los equipos que envían paquetes.

La información almacenada está en un fichero de texto que contiene por cada línea una ip de un equipo junto con su máscara.

La máscara se especifica en formato CIDR (Classless Interdomain Routing), como se muestra en el párrafo siguiente. Un posible contenido del fichero sería:

72.18.20.2/16
172.19.15.2/20
80.19.15.2/8
192.168.10.15/32
192.168.10.15/8
192.168.10.15/16
192.168.10.15/24
192.168.10.15/32
192.168.10.15/0

El fichero que contiene dicha información se llama **direcciones.txt** y está ubicado en el mismo directorio donde se ejecutará la aplicación. Se pide implementar un programa en C que genere la siguiente información:

Por cada línea del fichero *direciones.txt* se ha de obtener:

- ✓ Mostrar la línea leída
- ✓ Dirección IP del equipo
- ✓ Máscara del equipo en formato de 4 octetos visualizados en decimal separados por un punto
- ✓ Dirección de red a partir de la ip y de la máscara especificada

Posibles aclaraciones sobre la corrección

- ✓ En ningún momento se ha de validar la información del fichero, la cual se supone correcta; Es decir todas las ip's son correctas y la máscara especificada también.
- ✓ Se ha de implementar el código en **C**
- ✓ El código ha de seguir una lógica bien estructurada y correctamente comentada, siguiendo las buenas prácticas de programación
- ✓ La sintaxis ha de ser correcta.
- ✓ Como pide el enunciado, el objetivo es obtener la información de ip de red y máscara a partir de cada línea del fichero.

Como ejemplo, para el fichero aportado, se espera la siguiente salida:

```
Línea leída 172.18.20.2/16
ip: 172.18.20.2 - mascara : 255.255.0.0 - red: 172.18.0.0 -
-------------------------
Línea leída 172.19.15.2/20
ip: 172.19.15.2 - mascara : 255.255.240.0 - red: 172.19.0.0 -
-------------------------
Línea leída 80.19.15.2/8
ip: 80.19.15.2 - mascara : 255.0.0.0 - red: 80.0.0.0 -
-------------------------
Línea leída 192.168.10.15/32
ip: 192.168.10.15 - mascara : 255.255.255.255 - red: 192.168.10.15 -
-------------------------
Línea leída 192.168.10.15/8
ip: 192.168.10.15 - mascara : 255.0.0.0 - red: 192.0.0.0 -
-------------------------
Línea leída 192.168.10.15/16
ip: 192.168.10.15 - mascara : 255.255.0.0 - red: 192.168.0.0 -
-------------------------
Línea leída 192.168.10.15/24
ip: 192.168.10.15 - mascara : 255.255.255.0 - red: 192.168.10.0 -
-------------------------
Línea leída 192.168.10.15/32
ip: 192.168.10.15 - mascara : 255.255.255.255 - red: 192.168.10.15 -
-------------------------
Línea leída 192.168.10.15/0
ip: 192.168.10.15 - mascara : 0.0.0.0 - red: 0.0.0.0 -
-------------------------
BUILD SUCCESSFUL (total time: 0 seconds)
```

EJERCICIO. PROGRAMACIÓN EN C y REDES. SOLUCIÓN PROPUESTA

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>


#define ERR -1
#define OK 1
#define MAX_DIM 80


char *fgets(char *linea, int max, FILE * fp);
int separar_linea();
int leer_archivo();
int binarioADecimal(char *cadenaBinaria, int longitud);


FILE * fp;
int max = MAX_DIM;


int main(int argc, char *argv[])
{
    if (leer_archivo()== ERR)
          return ERR;

    separar_linea();

    fclose(fp);
    return OK;
}


int leer_archivo()
{
    if ((fp = fopen("direcciones.txt", "r")) == NULL)
    {
          printf("Error al intentar abrir el archivo\n");
          return ERR;
    }
    return OK;
}
```

```
int separar_linea()
{
    char cad[MAX_DIM];
    char delimitador[]=". / \n";

    while (!feof(fp))
    {
        fgets(cad, MAX_DIM, fp);
        printf("Línea leída %s \n",cad);
        char *token = strtok(cad,delimitador);

        int numeros[60];
        int contador=0;

        if(token!=NULL)
        {
            do
            {
                int aux= atoi(token);
                numeros[contador]=aux;

                token = strtok(NULL,delimitador);
                contador++;

            }while (token!=NULL);

            char mascaraAux[8];
            int masc[4];
            int unos;

            //mascara de 32 unos
            if(numeros[4]==32)
            {
                masc[0]=255;
                masc[1]=255;
                masc[2]=255;
                masc[3]=255;
            }
            else
            if(numeros[4]>24)
            {
                masc[0]=255;
                masc[1]=255;
                masc[2]=255;
                unos = numeros[4]-24;

                for(int i=0;i<unos;i++)
                    mascaraAux[i]='1';

                for(int j=unos;j<8;j++)
                    mascaraAux[j]='0';

                masc[3]=binarioADecimal(mascaraAux,8);
            }
            else
            if(numeros[4]==24)
            {
                masc[0]=255;
                masc[1]=255;
```

```
		masc[2]=255;
		masc[3]=0;
	}
	else
	if(numeros[4]>16)
	{
		masc[0]=255;
		masc[1]=255;
		masc[3]=0;
		unos = numeros[4]-16;

		for(int i=0;i<unos;i++)
			mascaraAux[i]='1';

		for(int j=unos;j<8;j++)
			mascaraAux[j]='0';

		masc[2]=binarioADecimal(mascaraAux,8);
	}
	else
	if(numeros[4]==16)
	{
		masc[0]=255;
		masc[1]=255;
		masc[2]=0;
		masc[3]=0;
	}
	else
	if(numeros[4]>8)
	{
		masc[0]=255;
		masc[2]=0;
		masc[3]=0;
		unos = numeros[4]-8;

		for(int i=0;i<unos;i++)
			mascaraAux[i]='1';

		for(int j=unos;j<8;j++)
			mascaraAux[j]='0';

		masc[1]=binarioADecimal(mascaraAux,8);
	}
	else
	if(numeros[4]==8)
	{
		masc[0]=255;
		masc[1]=0;
		masc[2]=0;
		masc[3]=0;
	}
	else //menos de 8 unos
	{
		masc[1]=0;
		masc[2]=0;
		masc[3]=0;
		unos = numeros[4];
```

```
                        for(int i=0;i<unos;i++)
                                mascaraAux[i]='1';

                        for(int j=unos;j<8;j++)
                                mascaraAux[j]='0';

                        masc[0]=binarioADecimal(mascaraAux,8);
                }

                //calcular la red con IP AND MASCARA
                int red[4];
                for(int i=0;i<4;i++)
                {
                        red[i]=numeros[i] & masc[i];
                }

                //mostrar datos
                printf("ip: %d.%d.%d.%d -",numeros[0],numeros[1],numeros[2],numeros[3]);
                printf("mascara: %d.%d.%d.%d - ",masc[0],masc[1],masc[2],masc[3]);
                printf("red: %d.%d.%d.%d - \n",red[0],red[1],red[2],red[3]);
                printf("------------------------------ \n");
        }
    }
    return 1;
}

int binarioADecimal(char *cadenaBinaria, int longitud)
{
      int decimal = 0;
      int multiplicador = 1;
      char caracterActual;

      for (int i = longitud - 1; i >= 0; i--)
      {
           caracterActual = cadenaBinaria[i];
           if (caracterActual == '1')
           {
             decimal += multiplicador;
           }
           multiplicador = multiplicador * 2;
      }
      return decimal;
}
```