UNIVERSIDAD DE MAGALLANES
FACULTAD DE INGENIERÍA
DEPTO. DE ING. EN COMPUTACIÓN

Punta Arenas, 09 de Junio del 2003.

Programación de Computadores

Resolución Prueba Nº 2

(Funciones y Arreglos)

Recuerde que muchas formas de resolver un problema, los ejercicios en esta resolución son para usar de referencia, los problemas aquí resueltos no son las alternativas más eficientes, pero si son bastante entendibles y faciles de ver.

1.- Mano de Truco:

(1.5 ptos.)

```
#include <stdio.h>

#define MAX 10

void leematriz(int matriz[MAX][MAX], int tamano);
int esmagica(int matriz[MAX][MAX], int tamano);

int main()
{
   int tamano, matriz[MAX][MAX];
   scanf("%d",&tamano);
   leematriz(matriz, tamano);
   if (esmagica(matriz,tamano)==1)
      printf("\nla matriz ingresada es magica\n");
   else
      printf("\nla matriz ingresada NO es magica\n");
   return 0;
}


void leematriz(int matriz[MAX][MAX], int tamano)
{
   int i,j;
   for(i=0;i<tamano;i++)
      for(j=0;j<tamano;j++)
         scanf("%d",&matriz[i][j]);
   return;
}

int esmagica(int matriz[MAX][MAX], int tamano)
{
   int i, j;
   int fil[MAX], col[MAX];
   int diag1=0, diag2=0;
   for(i=0;i<tamano;i++)
   {
      fil[i]=0;
      col[i]=0;
      /*aprovecho el for para calcular las diagonales*/
      diag1=diag1+matriz[i][i];
      diag2=diag2+matriz[tamano-1-i][tamano-1-i];
```

```
   }
/*   printf("[%d,%d]",diag1,diag2);*/
   if (diag1!=diag2) /*para que seguir*/
      return 0;
   for(i=0;i<tamano;i++)
   {
      for(j=0;j<tamano;j++)
      {
         fil[i]=fil[i]+matriz[i][j];
         col[i]=col[i]+matriz[j][i];
      }
      /* no es necesario continuar si una fila o columna es
      distinta a la diagonal*/
      if (fil[i]!=diag1 || col[i]!=diag1)
         return 0;
   }
   return 1;
}
```

Bastaba que el programa hiciera lo que se pedía, no importando la eficiencia del mismo.

2.- Dígito Verificador:

(1.5 ptos.)

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX 15

int verificar(char rut[MAX]);
int isrut(int rut_trans[MAX],char rut[MAX]);
int char_num(char a);
void transforma(int a[MAX], char rut[MAX]);
int digitos_raros(char a[MAX]);
int cuantas_rayas(char a[MAX]);

main()
{
       char rut[MAX];
       int rut_trans[MAX];
       gets(rut);
       if (verificar(rut))
       {
              transforma(rut_trans,rut);
              if ( isrut(rut_trans,rut)==1 )
                     printf ("\nEl rut esta correcto");
              else
                     printf ("\nEl rut NO esta correcto");
       }
       else
              printf ("\nEl formato de entrada NO es correcto");
       return 0;
}

int verificar(char rut[MAX])
{
       char rut2[MAX];
       int flag=0,i;
       if (digitos_raros(rut))
       {
              printf("digitos_raros = %d",digitos_raros(rut));
              return 0;
       }
       if (cuantas_rayas(rut)!=1)
       {
              printf("cuantas_rayas = %d",cuantas_rayas(rut));
              return 0;
```

```
        }
        for (i=0; rut[i]!= '-'; i++)
                rut2[i] = rut[i];
        rut2[i]='\0';
        if (strlen(rut2)>10)
                return 0;
        else
        {
                for (i=0; i<strlen(rut2); i++)
                        if (rut2[i]<'0' || rut2[i]>'9')
                                flag++;
                if (flag!=2 && flag!=0)
                        return 0;
        }
        return 1;
}

int isrut(int rut_trans[MAX],char rut[MAX])
{
        int i,j=0,suma=0;
        j=0;
        for (i=MAX-1; i>=0; i--)
        {
                printf ("\nsuma= %d + %d * %d",suma,rut_trans[i],((j%6)+2));
                suma = suma + rut_trans[i]*((j%6)+2);
                printf ("\nsuma= %d",suma);
                j++;
        }
        printf ("\nsuma = %d\n", suma);
        suma=suma%11;
        if (suma!=0)
                suma=11-suma;
        printf ("\nsuma = %d\n", suma);
        printf ("\nrut[strlen(rut)-1] = %c",rut[strlen(rut)-1]);
        if (suma==10)
        {       /* rut[strlen(rut)-1] es el ultimo elemento antes del '\0' */
                if (rut[strlen(rut)-1]=='k' || rut[strlen(rut)-1]=='K')
                        return 1;
                else
                        return 0;
        }
        else
        {
                if (char_num(rut[strlen(rut)-1])==suma)
                        return 1;
                else
                        return 0;
        }
}

int char_num(char a)
{
        if (a>='0' && a<='9')
                return (a-'0');
        else
                return -1;
}

void transforma(int a[MAX], char rut[MAX])
{
        int j=MAX-1,i=0;
        for (i=0; i<MAX; i++)
                a[i]=0;
        for (i=strlen(rut)-3; i>=0; i--)
                if (rut[i]>='0' && rut[i]<='9')
                {
                        a[j] = char_num(rut[i]);
                        printf ("\na[%d] = %d",j,a[j]);
                        j--;
                }
        return;
}

int cuantas_rayas(char a[MAX])
```

```
{
	int flag=0,i=0;
	for (i=0; i<MAX; i++)
		if (a[i]=='-')
			flag++;
	return flag;
}

int digitos_raros(char a[MAX])
{
	int flag=0,i=0;
	for (i=0; i<MAX; i++)
		if (a[i]<'0' || a[i]>'9')
			if (a[i]!='.' && a[i]!='-' && a[i]!='k' && a[i]!='K')
				flag++;
	if (flag!=3 && flag!=0)
		return 1;
	else
		return 0;
}
```

Note que este programa está completo, en la corrección de la prueba no se tan exigente.

(Nota: este ejercicio es de prueba anteriores, se agrego la parte de permitir distintas entradas válidas y de detectar entradas no-válidas)

Los demás ejercicios de la prueba son muy parecidos a los resueltos en clases.