FUNDAMENTOS DE COMPUTACIÓN
ICM00794

"Impulsando la sociedad del Conocimiento"

Solución a Examen FINAL I Término 2003 02/Septiembre/2003

Colaboración: Ing. Guillermo Gallegos. 2003

TEMA 1:

Inicio

```
/*
 * TEMA 1. Programa que calcula las temperaturas medias de un a¤o
 * y la diferencia de temperaturas entre el mes m s caluroso y el m s fr¡o
 */

#include <stdio.h>
#include <conio.h>

#define NUMERO_MESES 12

/* Prototipos de las funciones */

float promedio(float vector[], int num_elem);
float mayor(float vector[], int num_elem);
float menor(float vector[], int num_elem);

void main(void){
    float temperaturasMedias[NUMERO_MESES];
    float temperaturaMedia, diferencia;
    int j;
    clrscr();
    /* Lectura de las temperaturas medias de los 12 meses */
```

Ingreso

```
   for(j = 0; j < NUMERO_MESES; j++) {
printf("Introduzca la temperatura del mes %d: ", j+1);
scanf("%f", &temperaturasMedias[j]);
  }
```

Procedimiento

```
    /* C lculo de la temperatura media del a¤o */
    temperaturaMedia = promedio(temperaturasMedias,NUMERO_MESES);

    /* C lculo de la diferencia de temperaturas entre el mes m s
       caluroso y el m s fr¡o*/
    diferencia = mayor(temperaturasMedias,NUMERO_MESES) - menor(temperaturasMedias,NUMERO_MESES);
```

Salida

```
    /* Mostrar resultados */
    printf("La temperatura media del a¤o es %f\n", temperaturaMedia);
    printf("La diferencia entre el mes m s caluroso y el m s fr¡o es %f",diferencia);
```

Fin

```
}
```

```
/* Funci¢n promedio:  Esta funci¢n recibe como par metros un vector de elementos
 * reales y el n£mero de elementos del vector y devuelve el promedio de
 * los elementos del vector
 */

float promedio(float vector[], int num_elem){
    float media = 0;
    int j;
    for(j = 0; j < num_elem; j++)
        media = media + vector[j];
```

```
    media = (media / num_elem);
    return (media);
}

/* Funci¢n mayor:  Esta funci¢n recibe como par metros un vector de elementos
 * reales y el n£mero de elementos del vector y devuelve el mayor de
 * los elementos del vector
 */

float mayor(float vector[], int num_elem){
        int indice;
        float may;

        may = vector[0];
        for(indice = 0; indice < num_elem; indice++)
                if(vector[indice] > may)
                        may = vector[indice];
        return(may);
}

/* Funci¢n menor:  Esta funci¢n recibe como par metros un vector de elementos
 * reales y el n£mero de elementos del vector y devuelve el menor de
 * los elementos del vector
 */

float menor(float vector[],int num_elem){
        int indice;
        float min;

        min = vector[0];
        for(indice = 0; indice < num_elem; indice++)
                if(vector[indice] < min)
                        min = vector[indice];
        return(min);
}
```

TEMA 2: (Volver al Examen)

```
# include <stdio.h>
# include <conio.h>

/* Prototipos de las funciones */

float funcion_f(int n);

void main(){
        int i;
        float max;
        clrscr();
     /* C lculo del mayor valor de f para n = 0, 1, 2, 3, 4, 5,
      * mediante sucesivas llamadas a la funci¢n funcion_f
      */
        max = funcion_f(0);
        for (i = 1;i<=5;i++)
                if (funcion_f(i)>max)
                        max = funcion_f(i);
     /* Mostrar el mayor valor */
        printf("El valor m ximo de f (n = 0,1,2,3,4,5) es %f",max);
}

/* Funci¢n recursiva funcion_f, definida tal cual se solicit¢
 * Tenga presente que (0 mod 2) es 0, para el tercer if
 */

float funcion_f(int n){
        float ret;
        if ((n==0)||(n==1))
                ret=(1/2.0);
        if (((n%2)!=0)&&(n>1))
                ret = ((funcion_f(n-1)+funcion_f(n-2))/2.0);
        if (((n%2)==0)&&(n!=0))
                ret = ((funcion_f(n-1)-funcion_f(n-2))/2.0);
        return(ret);
}
```

TEMA 3: (Volver al Examen)

```
#include <stdio.h>
#include <conio.h>

/* Prototipos de las funciones */

void ingresar(char cad[]);
void encriptar(char cad[]);
void mostrar(char cad[]);
int elemento_tres(char cad[],int pos_ini);

void main(){
        char frase[81];
        ingresar(frase);
        encriptar(frase);
        mostrar(frase);
}

/* Funci¢n ingresar.  Esta funci¢n ingresa un m ximo de 80 caracteres
 * en una cadena pasada como par metro
 */

void ingresar(char cad[]){
        char c;
        int cont;
        clrscr();
        printf("Ingrese una frase que contenga un m ximo de 80 caracteres:\n");
        cont = 0;
        c = getchar();
        while ((c!='\n')&&(cont<80)){
                cad[cont] = c;
                cont++;
                c = getchar();
        }
        cad[cont] = '\0';
}

/* Funci¢n encriptar.  Esta funci¢n recibe como par metro una cadena
 * de caracteres y la encripta seg£n las reglas especificadas
 */

void encriptar(char cad[]){
     /* Declaraci¢n de variables locales
      * terminar: bandera de control del bucle while
      * pos_ini: posici¢n inicial del grupo actual de 3 elementos
      * pos_fin: posici¢n final del grupo actual de 3 elementos
      */
        int terminar, pos_fin, pos_ini;
        char aux;
     /* inicializaci¢n de variables */
        terminar = 0;
        pos_fin = -1;
        while (!terminar){
         /* se toma el siguiente grupo de 3 elementos */
            pos_ini = pos_fin +1;
         /* Si el siguiente elemento es '\0' la cadena termino.  Salir */
            if(cad[pos_ini] == '\0')
                        terminar = 1;
         /* Si no, buscar si existe otro grupo de 3 elementos */
            else{
             /* Si el grupo existe, intercambiar el primer
              * elemento con el tercero
              */
               if(elemento_tres(cad,pos_ini)){
                    pos_fin = pos_ini + 2;
                    aux = cad[pos_ini];
                    cad[pos_ini] = cad[pos_fin];
                    cad[pos_fin] = aux;
               }
             /* Si no (el grupo no existe: hay menos de 3 caracteres) salir.*/
               else
                    terminar = 1;
            }
        }
}
```

```
/* Funci¢n elemento_tres.  Esta funci¢n recibe como par metros una cadena
 * de caracteres cad y el ¡ndice de uno de sus elementos pos_ini.
 * Verifica si la cadena contiene un tercer car cter contado a partir
 * de pos_ini (en pos_ini + 2).  Si la cadena contiene dicho elemento
 * se retorna 1, si no se retorna 0
 */

int elemento_tres(char cad[],int pos_ini){
        if ((cad[pos_ini+1]== '\0')||(cad[pos_ini+2]== '\0'))
                return (0);
        else
                return(1);
}

/* Funci¢n mostrar. Esta funci¢n muestra por pantalla una cadena
 * encriptada
 */

void mostrar (char cad[]){
        printf("La cadena encriptada es:\n");
        printf("%s",cad);
}
```

TEMA 4: (Volver al Examen)

```
/*
 * TEMA 4 DEL EXAMEN
 * LIMITACIONES:  no se ha considerado validaci¢n de datos
 */
# include <stdio.h>
# include <conio.h>
# include <string.h>

/* N£mero de carros por ser procesados */
# define NUM_CARS 3

/* Prototipos de las funciones */
void ingresar_datos();
void mostrar_datos();

/* Definici¢n del tipo CARRO */

typedef struct{
     char placa[7];
     int anio;
     char tipo;
     char marca[11];
     char color[11];
     float precio;
}CARRO;

void main(){
        ingresar_datos();
        mostrar_datos();
}

void ingresar_datos(){
        CARRO car;
        FILE * arch;
        int i,cont;
        char c, aux[11];
        clrscr();
        arch = fopen("D:/carros.dat", "wb");
        for (i=1;i<=NUM_CARS;i++){
             /* Ingresar datos del i-,simo carro */
                printf("Carro %d:",i);
             /* Ingresar la placa (de 6 caracteres */
                printf("\n\tPlaca (6 caracteres): ");
                scanf("%s",car.placa);
                while (strlen(car.placa)!=6){
                        printf("\r\tPlaca (6 caracteres):          \b\b\b\b\b\b\b\b\b\b");
                        scanf("%s",car.placa);
                }
             /* Ingresar a¤o.  Ojo. No se verifica que lo que se ingresa es entero*/
```

```
                printf("\n\tA¤o: ");
                scanf("%d",&car.anio);
            /* Ingresar el tipo de carro.  Solamente se acepta 'A' o 'C'*/
                printf("\tTipo (A: auto, C: camioneta):   \b\b");
                c = getche();
                while ((c!='A')&&(c!='C')){
                        printf("\r\tTipo (A: auto, C: camioneta): ");
                        c = getche();
                }
                car.tipo = c;
            /* Ingresar la marca. Ojo.  No se aceptan espacios en blanco*/
                printf("\n\tMarca: ");
                scanf("%s",car.marca);
            /* Ingresar color */
                printf("\tColor: ");
                scanf("%s",car.color);
            /* Ingresar el precio.  Ojo.  No se verifica que lo que se ingresa es float*/
                printf("\tPrecio: ");
                scanf("%f",&car.precio);
                fwrite(&car,sizeof(car),1,arch);
        }
        fclose(arch);
}


void mostrar_datos(){
        CARRO car;
        FILE * arch;
        int i, cont= 0;
        arch= fopen("D:/carros.dat", "rb");
        printf("\n\nLOS CARROS BUSCADOS SON:\n");
        for (i=1;i<=NUM_CARS;i++){
                fread(&car,sizeof(car),1,arch);
            /* Ojo en el color del carro solo se considera la cadena "rojo",  No "Rojo" por ejemplo */
                if ((car.anio>1995)&&(!strcmp(car.color,"rojo"))&&(car.precio<6000)){
                        printf("\nC%d: %s %d %c %s %s %f",i,car.placa,car.anio,car.tipo,car.marca,car.color,car.precio);
                        cont++;
                }
        }
        if (cont == 0)
                printf("\nNo hay carros que cumplan los requisitos solicitados");
        fclose(arch);
}
```


Comentarios: edelros@goliat.espol.edu.ec