Programación

Ejercicios Tema 4
Ejercicios Avanzados

Autores:

M. Paz Sesmero Lorente
Paula de Toledo Heras
Fco. Javier Ordoñez Morales
Juan Gómez Romero
Jose A. Iglesias Martínez
Jose Luis Mira

SOLUCIONES

1. **Escribir un programa en C que escriba los números comprendidos entre 1 y 1000. El programa escribirá en la pantalla los números en grupos de 20, solicitando al usuario si quiere o no continuar visualizando el siguiente grupo de números. Generalizar el programa para que escriba los números comprendidos entre dos valores que introduzca el usuario, y sea éste también quien decida el tamaño del grupo a visualizar por pantalla.**

```
#include <stdio.h>

int main(void)
{
  int continuar=1; //variable lógica que indica si se mostrará el
                   //siguiente bloque
  int i, j;

  j=1; //La variable j determina el número a imprimir por pantalla
  while ((continuar==1)&&(j<1000)){
     for (i=j; i<j+20; i++){
         printf ("%d\n", i);
     }
     j=i;   //Se actualiza el valor de j y se pregunta si se desea
            //mostrar el siguiente bloque
     printf ("Desea continuar 1:SI; 0:NO ");
     scanf ("%d", &continuar);

  }

  return 0;
}
```

Segunda parte:

```
#include <stdio.h>

int main(void)
{
  int continuar=1; //variable lógica que indica si se mostrará el
siguiente bloque
  int i,j;
  int l1, l2;    //l1 y l2 son los límites establecidos
  int tam;       //Tamaño del bloque

  //Se solicitan los valores de l1, l2 y tam al usuario.
  printf("Introduzca el primer y el ultimo numero a visualizar ");
  scanf ("%d %d", &l1, &l2);

  printf ("Introduzca la dimensión del bloque ");
  scanf ("%d", &tam);

  j=l1;  //El primer número a mostrar coincide con l1
  while ((continuar==1)&&(j<l2)){
      //En este caso es necesario controlar que el número a
      //mostrar es inferior a l2. Para ello se usará una condición
      //doble.
     for (i=j; ((i<j+tam)&&(i<=l2)); i++){
```

```
            printf ("%d\n", i);
        }
        j=i;
        if (j<12){
            printf ("Desea continuar 1:SI; 0:NO ");
            scanf ("%d", &continuar);
        }

    }

    return 0;
}
```

2. Escriba un programa que lea números por teclado hasta que se introduzca el cero. En ese momento deberá representar el número de introducciones efectuadas, y la mayor secuencia de números consecutivos iguales, indicando cuál fue el número que se repitió y cuántas veces seguidas apareció.
Ejemplo: Si se introduce 8 8 8 4 5 6 6 6 7 7 7 7 2 0, el resultado a mostrar será. 'El número más repetido es el 7 y se ha escrito 4 veces'.

```
#include<stdio.h>

int main(void){

  int num ,ant,cont,repe,veces ,maxVeces ;
  // num - último numero leído (entero)
  // ant - anterior número leído
  // cont - contador de numeros leídos
  // repe - número que más se repitió
  // veces - veces que se repite un numero
  //maxVeces - maximo numero de veces que se repitió

  //Se inicializan las distintas variables

  cont = 0;
  repe =  0;
  veces = 0;
  maxVeces =0;
  ant = 0;

  //Se lee el primer número
  printf ("dame numeros, termina con 0\n");
  scanf("%i", &num);

  //Si el número es distinto de cero:
  while(num != 0){
    //a)   Se actualiza el contador de números
    cont = cont + 1;
    //b)   Se comprueba si coincide o no con el anterior
    if (num == ant){
      //En caso afirmativo:
      //   a) Aumentamos "veces" en una unidad
      veces=veces + 1;
      //   b) Se compruba si el tamaño de la secuencia es superior
      //      a la máxima secuencia introducida. En ese caso se
      //      actualiza el valor de repe y maxVeces.
      if (veces >maxVeces){
        repe = num;
        maxVeces=veces;
```

```
        }
      }
      else{
        // El caso contrario indica que ha comenzado una nueva
        // secuencia. Por tanto, el número de repeticiones se
        // inicializa a 1
          veces = 1;
      }
      //Se procede a leer un nuevo número. Previamente se actualiza
      // el valor de ant:
      ant = num;
      scanf("%i", &num);

    }

    // Se muestran los resultados:
    if (cont!=0){
       printf ("El numero mas repetido es el %i \n", repe);
       printf ("y ha aparecido %i veces\n", maxVeces);
    }


    return 0;
}
```

3. Escriba un programa en C que solicite un número y calcule su factorial. El factorial de un número n (representado por n!) es el producto de todos los números naturales desde 1 hasta n.
n! = 1 * 2 * 3 * … * (n-1) * n.

```
#include <stdio.h>

int main(void)
{
  int n, factorial;
  int i;
  printf ("Introduzca el numero del que quiere conocer el factorial
");
  scanf ("%d", &n);

  //Se inicializa factorial a 1;
  factorial=1;

  //Dado que el factorial tanto de 0 como de 1 es 1, el caso
genérico
  //se da para n>=2. En este caso:
  //n!=1*2*3*....*n

  for (i=2; i<=n; i++)
    factorial=factorial*i;

  printf ("El factorial de %d es %d\n", n, factorial);


  return 0;
}
```

4. Escriba un programa que calcule la raíz cuadrada de un número con n decimales, por el método de aproximaciones sucesivas.

Nota: Método de las aproximaciones sucesivas
Se basa en 'ensayo y error'. A este tipo de algoritmos se les llama de aproximaciones sucesivas: se empieza por un valor inicial y se va modificando para acercarse al resultado.
En este caso se comienza suponiendo que la raíz es 1. Se comprueba si raiz*raiz=num y si no es así se suma un incremento, y se vuelve a comprobar. Se repiten estos pasos hasta que raiz*raiz>num.
Para computar el valor con un número determinado de decimales se repiten esos pasos con un incremento cada vez menor (la primera vuelta 1, la segunda 0,1, la tercera 0,01
En cada paso la precisión es un decimal más, por lo que se repite hasta que el número de decimales sea el introducido por el usuario, o hasta que se encuentre el valor exacto.

```
#include <stdio.h>
int main (void){
   int n;                 //Num. del que queremos calcular la raíz
   double raiz;           //Num. con el que probaremos si es la raiz
   float incremento;
   int decimales, dec; //Num. de decimales a los que aproximaremos
                          //y núm. de decimales con los que estamos
                          //realizando el cálculo

   //Solicitamos un número hasta que Éste sea positivo
   do{
      printf("Introduzca el núm. del que quiere calcular la
raiz\n");
      scanf ("%d", &n);
      if (n<0)
      printf("ERROR: No se puede calcular la raiz de un num
negativo\n");
   }while (n<0);

   //Solicitamos el número de decimales que tendrá la aproximación.
  // Este número debe ser positivo:
   do{
      printf ("Introduza el número. de decimales que tendrá la
aproximación\n");
      scanf ("%d", &decimales);
   }while (decimales<0);

   raiz=0;      //Inicializamos raiz al menor entero posible (0)
   incremento=1.0;//Inicialmente buscamos el entero más próximo a
                  // la raiz; Es decir, probaremos con números que
                  // difieren entre sí en 1 unidad.
   dec=0;       //Por tanto, el número de decimales para la primera
                  //aproximación es 0

   while ((n>(raiz*raiz) )&&(decimales>=dec)){
     while(raiz*raiz<n) {
         raiz=raiz+incremento;
     }
    //Al salir del bucle, el valor almacenado en raiz será mayor
```

```
        //o igual que el número buscado.
        //Si es mayor (nos hemos pasado), actualizamos el valor de raiz
            if (raiz*raiz!=n){
                raiz=raiz-incremento;
                   //Ajustamos a un nuevo decimal
                incremento=incremento/10.0;
                  dec++;
            }
       }
       printf ("La raiz cuadrada de %d es %f\n", n, raiz);
       return 0;
}
```

5. **Se quiere averiguar el número mágico de una persona. Para calcularlo se suman todos los números de su fecha de nacimiento y a continuación se reducen a un solo dígito.**
   Ejemplo:
   Fecha de nacimiento: 05/02/1973
   5 + 2 + 1973 = 1980 => 1 + 9 + 8 + 0 = 18 =>9

   Realice un programa que:
   - **a) Solicite al usuario el año, el mes y el día que componen su fecha de nacimiento y compruebe que la fecha introducida es una fecha válida. La fecha se considerará válida si: año >0; 1<=mes<=12; 1<=dia<=DIAS_MES. Se considerará que, con independencia del año, el número de días del mes de Febrero es 28.**
   - **b) Calcule el número mágico asociado a esa fecha.**
   - **c) Muestre el número mágico por pantalla**

```
#include <stdio.h>
int main(void){
  //Declaración de variables:
  int year, mes, dia; //Datos referidos a la fecha de nacimiento.
  int diasMes;       //Días del mes de nacimeinto. Se usa para
                      //comprobar la validez de la fecha
  int magico;         //Almacena el número mágico
  int aux;

  //Se solicita la fecha de nacimiento de una persona comprobando
  //que es válida:
  do{
    printf ("Introduza su año de nacimiento: ");
    scanf ("%i", &year);
  }while (year<0);

  do{
    printf ("Introduza su mes de nacimiento: ");
    scanf ("%i", &mes);
  }while ((mes<1)||(mes>12));

  //SE CALCULA EL NÚMERO DE DÍAS QUE TIENE EL MES INTRODUCIDO
  switch (mes){
    case 1: case 3: case 5: case 7: case8: case 10: case 12:{
       diasMes=31;
       break;
    }
```

```
      case 4: case 6: case 9: case 11:{
        diasMes=30;
        break;
      }
      default: //Se considera que febrero tiene 28 dias
       diasMes=28;
    }
    do{
      printf ("Introduza su dia de nacimiento: ");
      scanf ("%i", &dia);
    }while ((dia<1)||(dia>diasMes));

    //Se inicializa mágico a la suma de todos los números de la fecha
    //de nacimiento:
    magico=dia+mes+year;
    //Se procede a reducir el número obtenido a un número de una sola
    //cifra
    while (magico> 9) {
       //Se suman los dígitos que componen el número mágico
      aux = magico;
      magico=0;
      while (aux> 0) {
        magico = magico + (aux % 10);
        aux = aux - (aux %10);
        aux = aux / 10;
      }
      //Si magico tiene dos o más cifras, se repite el bucle.
    }
    printf ("Su numero magico es = %d \n", magico);

     return 0;
}
```