	FUNDAMENTOS DE INFORMÁTICA (141211006) INGENIEROS INDUSTRIALES CONVOCATORIA FEBRERO 2008 (29-enero-2008)
	NOMBRE: DNI:

PRIMERA PREGUNTA	1 PUNTO

Vea el siguiente código e indique la salida que ofrecerá por pantalla.

```
#include <stdio.h>
void main(void)
{
      short a = 5, b = 7;   long c = a < b ? b - a : a - b;
      switch(c)
      {
      case 1: printf("uno\t");
      case 2: printf("dos\t");
      case 3: printf("tres\t");
      default: printf("error\t");
      }
}
```

dos tres error

SEGUNDA PREGUNTA	2 PUNTOS

Vea la definición de la función cuyo prototipo es `void f(char*, int);`

```
void f( char *cad, int i)
{
   if( cad[i] != '\0' ){ f(cad,i+1); printf("%c", cad[i] ); }
}
```

Suponga que la invocamos con los siguientes valores de entrada:

`f("EXAMEN", 0);`

Indique entonces qué salida ofrece esta función por pantalla.

NEMAXE

TERCERA PREGUNTA	2 PUNTOS

Lea el siguiente código e indique la salida que generará por pantalla.

```
#include <stdio.h>
void main(void)
{
   short i, j;  long m[2][10];
   for(i = 0 ; i < 2 ; i++)
      for(j = 0 ; j < 10 ; j++)
         if(j < 2) *(*(m + i) + j) = i ? 1 : j + 1;
         else *(*(m+i)+j) = i ? *(*(m+i)+j-2) + *(*(m+i)+j-1) : j+1;
   for(j = 2 ; j < 7 ; j++)
      printf("%3ld --- %3ld\n", *(*(m+0)+j) , *(*(m+1)+j));

}
```

```
 3 ---  2
 4 ---  3
 5 ---  5
 6 ---  8
 7 --- 13
```

CUARTA PREGUNTA	1 PUNTO

Escriba un programa que implemente y muestre la matriz identidad 3 X 3.

```
#include <stdio.h>
void main(void)
{
      long id[3][3];
      short i, j;
      for(i = 0 ; i < 3 ; i++)
            for(j = 0 ; j < 3 ; j++)
                  id[i][j] = i == j ? 1 : 0;
      for(i = 0 ; i < 3 ; i++)
      {
            printf("\n");
            for(j = 0 ; j < 3 ; j++)
                  printf("%5ld", id[i][j]);
      }
}
```

QUINTA PREGUNTA	2 PUNTOS

Llamamos $d_1 = mcd(a_0, a_1)$ y $d_i = mcd(d_{i-1}, a_i)$ para $i = 2,3, \dots n$.

Podemos definir una función para el cálculo de máximo común divisor de varios enteros de la siguiente manera:

$mcd(a_0, a_1, \dots, a_n) = mcd(d_1, a_2, \dots, a_n) = mcd(d_2, a_3, \dots, a_n) = mcd(d_{n-1}, a_n) = d_n$.

Esta función recibirá como parámetros:

1. El valor de n que indica cuántos enteros hay introducidos en el array.
2. Un array (enteros de 4 bytes) con los enteros de los que queremos calcular el máximo común divisor de todos ellos. Ese array tendrá una dimensión igual a 100, que será el valor máximo que podrá tener n.

Y devuelve el valor del máximo común divisor de todos los enteros recibidos.

Escriba el prototipo de la función, que llamaremos *mcd_multiple*:

```
long mcd_multiple(long, long*);
```

Suponga que el siguiente código invoca a la función:

```
long a[100], m, i;
for(i = 0 ; i < 100 ; i++)
{
      printf("entero ... ");      scanf("%ld", a + i);
      if(!*(a + i)) break;
}
m = mcd_multiple(i, a);
```

Escriba el código de la función mcd_multiple que calcula y devuelve ese valor del máximo común divisor de todos los enteros introducidos. Puede suponer que tiene definida una función cuyo prototipo es `long mcd(long, long);` y cuya definición es

```
long mcd(long a, long b) {  return b == 0 ? a : mcd(b, a % b);  }
```

```
long mcd_multiple(long n, long *a)
{
      long i = 1, m = a[0];
      while(i < n)
      {
            m = mcd(m, *(a + i));
            i++;
      }
      return m;
}
```

SEXTA PREGUNTA	2 PUNTOS

El binomio de Newton proporciona la expansión de las potencias de una suma:

$(x+y)^n = \sum_{k=0}^{n} \binom{n}{k} \cdot x^{n-k} \cdot y^k$, donde $\binom{n}{k} = n!/k! \cdot (n-k)!$

Escriba un programa que solicite al usuario los valores de x, y, n y muestre entonces por pantalla el valor $(x+y)^n$ usando la expansión del binomio de Newton. **Sugerencias**: (1) Emplear la función ***double*** *pow(**double**, **double**);* (primer parámetro: base; segundo parámetro: exponente) recogida en *math.h*. (2) Será más cómodo usar la función llamada *factorial*, ya definida en el enunciado.

```
#include <stdio.h>
#include <math.h>
long factorial(short);
long factorial(short n)
{
      long F = 1;
      while(n) F *= n--;
      return F;
}

void main(void)
{
      short n, k;      double x, y, sumando, binomio = 0;

      printf("Valor de n ... "); scanf("%hd", &n);
      printf("Valor de x ... "); scanf("%lf", &x);
      printf("Valor de y ... "); scanf("%lf", &y);

      for(k = 0 ; k <= n ; k++)
      {
            sumando = factorial(n);
            sumando /= ((double)factorial(k) * factorial(n - k));
            sumando *= pow(x, n - k) * pow(y, k);
            binomio += sumando;
      }
      printf("Resultado = %lf.", binomio);
}
```