Ejercicios del Tema 6

Funciones

1. Áreas de figuras geométricas

Escribe un programa donde el usuario pueda seleccionar entre calcular el área de un círculo, un triángulo o un cuadrado. El programa empleará una función diferente definida para cada caso. Valgan como ejemplo los siguientes prototipos:

```
float aCirc (float r);
float aCuad (float l);
float aTri (float b, float h);
```

```
#include <stdio.h>

#define PI 3.141592

float aCirc (float r);
float aCuad (float l);
float aTri (float b, float h);

int main()
{
  char elige;
  float b, h, r, l;
  float area;

  printf("Elige Triangulo (T), Circulo (C), o Cuadrado (D).\n");
  scanf("%c", &elige);

  switch(elige)
    {
    case 't':
    case 'T':
      printf("Base y Altura?\n");
      scanf("%f %f", &b, &h);
      area = aTri(b, h);
      break;
    case 'c':
    case 'C':
      printf("Radio?\n");
      scanf("%f", &r);
      area = aCirc(r);
      break;
    case 'd':
    case 'D':
      printf("Lado?\n");
      scanf("%f", &l);
      area = aCuad(l);
      break;
    default:
      printf("Opcion desconocida\n");
      area = 0;
    }
```

```
  printf("Area %.2f\n", area);
  return 0;
}

float aCirc (float r)
{
  return PI * r * r;
}

float aCuad (float l)
{
  return l * l;
}

float aTri (float b, float h)
{
  return (b * h) / 2;
}
```

2. Conversión de temperaturas

Escribe un programa para realizar la conversión de temperaturas. El programa pide al usuario que introduzca un valor (número real) y una letra. La letra indica la escala en la que se introduce esa temperatura. Si la letra es 'C', la temperatura se convertirá de grados centígrados a Fahrenheit. Si la letra es 'F' la temperatura se convertirá de grados Fahrenheit a grados Centígrados. El programa se repetirá indefinidamente hasta que el usuario introduzca una letra diferente a 'C' o 'F' como escala. Se usarán 2 funciones `cent2fahr` y `fahr2cent` para convertir de una escala a otra. Estas funciones aceptan un parámetro (la temperatura en una escala) y retornan el valor en la otra escala. La relación entre ambas escalas es $T_F = 9/5 * T_C + 32$

```
#include <stdio.h>

float cent2fahr(float tc);
float fahr2cent(float tf);

int main()
{
  float t0, t1;
  char escala;

  do
    {
      printf("Indique la temperatura y escala (C o F).\n");
      scanf("%f %c", &t0, &escala);

      switch(escala)
        {
        case 'C':
          t1 = cent2fahr(t0);
          printf("El resultado es %.2f.\n", t1);
          break;
        case 'F':
          t1 = fahr2cent(t0);
          printf("El resultado es %.2f.\n", t1);
          break;
        default:
          printf("Opcion desconocida.\n");
          break;
        }
    }
  // El bucle se repetirá mientras la escala elegida sea
```

```
  // uno de los valores válidos
  while (escala == 'C' || escala == 'F');

  return 0;
}


float cent2fahr(float tc)
{
  return (9.0 / 5.0) * tc + 32;
}


float fahr2cent(float tf)
{
  return (5.0 / 9.0) * (tf - 32);
}
```

3. Tablas de multiplicar

Construye un un programa que muestre por pantalla las tablas de multiplicar del 1 al 10 en base a dos funciones específicas. La primera función debe devolver el producto de dos valores numéricos enteros dados como parámetros. La segunda función debe mostrar por pantalla la tabla de multiplicar de un número dado como parámetro.

```
#include <stdio.h>

int producto(int a, int b);
void tabla(int n);

int main()
{
  int i;
  for (i = 1; i <= 10; i++)
    tabla(i);
  return 0;
}

int producto(int a, int b)
{
  return a * b;
}

void tabla(int n)
{
  int i;
  printf("Tabla del %d:\n", n);
  for (i = 1; i <= 10; i++)
    printf("%d x %d = %d\n",
          n, i, producto(n, i));
}
```

4. Distancia entre puntos

Realiza un programa para calcular la distancia entre dos puntos bidimensionales introducidos por el usuario, empleando una función con el siguiente prototipo:

```
float dist (float x1, float y1, float x2, float y2);
```

```c
#include <stdio.h>
#include <math.h>

float dist (float x1, float y1, float x2, float y2);

int main()
{
  float x1, x2, y1, y2;
  float d;

  printf("Primer punto\n");
  scanf("%f %f", &x1, &y1);

  printf("Segundo punto\n");
  scanf("%f %f", &x2, &y2);

  d = dist(x1, y1, x2, y2);

  printf("La distancia entre los dos puntos es %f\n", d);
  return 0;
}

float dist (float x1, float y1, float x2, float y2)
{
  float dx2, dy2, d;
  dx2 = pow(x1 - x2, 2);
  dy2 = pow(y1 - y2, 2);
  d = sqrt(dx2 + dy2);
  return d;
}
```

5. Máximo de 3 números

Escribe un programa que calcule y muestre en pantalla el máximo de tres valores de tipo entero que se introducen por teclado. Este programa usa una función `max3` que admite como parámetros tres números enteros y entrega el valor del mayor. Esta función usará a su vez a otra función `max2` que calcula el máximo entre 2 valores que se pasan como parámetros a la función.

```c
#include <stdio.h>

int max2(int a, int b);
int max3(int a, int b, int c);


void main()
{
  int x, y, z, max;

  printf("Introduzca tres valores:\n");
  scanf("%d %d %d", &x, &y, &z);

  max = max3(x, y, z);

  printf("El mayor es %d.\n", max);
}

int max2(int a, int b)
{
  if (a >= b)
    return a;
```

```
  else
    return b;
}

int max3(int a, int b, int c)
{
  int mx_ab, mx;

  mx_ab = max2(a, b);

  mx = max2(mx_ab, c);

  return mx;
}
```

6. Números combinatorios

Escribe un programa que calcule y muestre en pantalla el número combinatorio a partir de los valores `n` y `k` introducidos por teclado.

$$nk = \frac{n!}{(n-k)! \cdot k!}$$

Este programa debe estar construido en base a dos funciones, una para calcular el factorial de un número, y otra para calcular el número combinatorio.

```
#include <stdio.h>

int factorial(int x);

void main()
{
  int n, k, a;

  printf("Introduzca n y k:\n");
  scanf("%d %d", &n, &k);

  a = factorial(n) / (factorial(n - k) * factorial(k));

  printf("El resultado es %d.\n", a);
}

int factorial(int x)
{
  int res;

  if (x <= 1)
    res = 1;
  else
    res = x * factorial(x -1);
  return res;
}
```

7. Fibonacci

Escribe un programa que genere los `n` primeros términos de la serie de Fibonacci **usando una función recursiva**. El número entero `n` deberá ser leído por teclado. Este valor debe ser positivo, de forma que si el usuario introduce un valor negativo el programa volverá a pedir que lo introduzca.

En la serie de Fibonacci los dos primeros números son 1, y el resto se obtiene sumando los dos anteriores: $1, 1, 2, 3, 5, 8, 13, 21, \ldots$

```c
#include <stdio.h>

int fib(int n);

int main()
{
  int i, num;

  scanf("%d", &num);

  for (i = 1; i <= num; i++)
    printf("%d \t %d\n",
          i, fib(i));
  return 0;
}

int fib(int n)
{
  int res;
  if (n == 1 || n == 2)
    res = 1;
  else
    res = fib(n - 1) + fib(n -2);
  return res;
}
```

8. Serie de Taylor

Escribe un programa que calcule la aproximación de e^{-x} mediante el desarrollo de Taylor:

$$e^{-x} = 1 + \sum_{i=1}^{\infty} \frac{(-x)^n}{n!}$$

El programa estará construido en base a tres funciones: `factorial` calcula el factorial de un número entero `n`; `potencia` calcula la potencia `n` de un número real `x`; `exponencial` calcula la aproximación anterior de un número real `x` usando `n` términos de la serie de Taylor. Sus prototipos son:

```c
float exponencial(float x, int n);
double potencia(float x, int n);
long int factorial(int n);
```

El programa solicitará al usuario en `main` el valor del número real y del número de términos deseados. No se debe utilizar la librería `math.h`.

```c
#include <stdio.h>

float exponencial(float x, int n);
double potencia(float x, int n);
long int factorial(int n);


int main()
{
  float x;
  double y;
  int n;
  printf("Indique el valor de x y el número de términos:\t");
  scanf("%f %i", &x, &n);
  y = exponencial(x, n);
```

```
  printf("La aproximación es %lf.\n", y);
}

float exponencial (float x, int n)
{
  int i;
  float suma = 1;
  for (i = 1; i < n; i++)
    {
      suma += potencia(-x, i) / factorial(i);
    }
  return suma;
}

double potencia (float x, int n)
{
  int i;
  double p = 1.0;
  for (i = 1; i <= n; i++)
     p *= x;
  return p;
}

long int factorial(int n)
{
  int i;
  long int res = 1;
  for (i = 1; i <= n; i++)
    res *= i;
  return res;
}
```