Ejercicios de programación en C

1.

Escribe un programa que solicite un nombre y haga lo siguiente con él:

1. Lo escribe entre comillas dobles.
2. Lo escribe en un campo de 20 caracteres, con el campo encapsulado entre comillas y el nombre en el extremo derecho del campo.
3. Lo escribe en un campo de 20 caracteres, con el campo encapsulado entre comillas y el nombre en el extremo izquierdo del campo.
4. Lo escribe en un campo de 3 caracteres mayor que el nombre.

#include <stdio.h>

#include <string.h>

int main(void)

{

char nom[15];

printf("Ingresa tu nombre: ");

scanf("%s", &nom);

    printf("\"%s\"", nom); //inciso a

    printf("\n'%20s'", nom); //inciso b

    printf("\n'%-20s'", nom); //inciso c

    printf("\n'%-\*s'", strlen(nom) + 3, nom); //inciso d

return 0;

}

2.

Escribe un programa que lea un numero de punto flotante y lo escriba primero en notación decimal y luego en notación exponencial. Que los datos de salida tengan los siguientes formatos (el número de dígitos mostrados en el exponente pude variar dependiendo del sistema):

1. La entrada es 21.3 o 2.1e+1.
2. La entrada es +21.290 o 2.129E+1.

#include <stdio.h>

int main(void)

{

float num;

printf("Ingresa tu numero de punto flotante: ");

scanf("%f", &num);

printf("La entrada es %.1f o %.1e", num, num); //inciso a

printf("\nLa entrada es %+.3f o %.3E", num, num); //inciso b

return 0;

}

3.

Escribe un programa que solicite la velocidad de descarga en megabits por segundo (Mbs) de un archivo y el tamaño de un archivo en megabytes (MB). El programa debe calcular el tiempo de descarga del archivo. En este contexto un byte es igual a ocho bits. Usa el tipo flotante, y usa / para la división. El programa debe reportar los tres valores (velocidad de descarga, tamaño de dato y tiempo de descarga) mostrando dos digitos a la derecha del punto decimal como en el siguiente ejemplo:

A 18.12 megabits por segundo, un archivo de 2.20 megabytes

se descarga en 0.97 segundos.

#include <stdio.h>

int main(void)

{

float Mbs, MB;

printf("Ingresa la velocidad de descarga en megabits por segundo (Mbs): ");

scanf("%f", &Mbs);

printf("Ingresa el tamano del archivo en megabytes (MB): ");

scanf("%f", &MB);

printf("A %.2f megabits por segundo, un archivo de %.2f megabytes", Mbs, MB);

printf("\nse descarga en %.2f segundos", MB/(Mbs/8));

return 0;

}

4.

Escribe un programa que solicite el nombre y apellidos del usuario. El prograba debe imprimir los nombres ingresados en una linea y el numero de letras de cada nombre en la siguiente, y alinear la cantidad de letras con la primer letra del nombre correspondiente. Ejemplo:

Carla Perez Gomez

5 5 5

#include <stdio.h>

#include <string.h>

int main(void)

{

char nom[15], ap1[15], ap2[15];

printf("Ingresa tu nombre y apellidos: ");

scanf("%s %s %s", &nom, &ap1, &ap2);

printf("%s %s %s", nom, ap1, ap2);

printf("\n%-\*d %-\*d %-\*d",

strlen(nom), strlen(nom), strlen(ap1), strlen(ap1) , strlen(ap2), strlen(ap2));

return 0;

}

5.

Escribe un programa que asigne a una variable de tipo flotante (float) el valor para la razón 1.0/3.0 y a una variable de tipo doble (double) el mismo valor. El programa debe mostrar cada resultado en tres formatos diferentes: uno con cuatro dígitos a la derecha del punto decimal, otro con 12 dígitos, y otro con 16 dígitos. Además, incluye la biblioteca <float.h> para que el programa muestre los valores de FLT\_DIG y DBL\_DIG. ¿Son los valores mostrados de 1.0/3.0 consistentes con estos límites?

#include <stdio.h>

#include <float.h>

int main(void)

{

float fx = 2.0 / 3.0;

double dx = 2.0 / 3.0;

printf("1.0 / 3.0 = %.4f", fx);

printf("\n1.0 / 3.0 = %.12f", fx);

printf("\n1.0 / 3.0 = %.16f", fx);

printf("\n\n1.0 / 3.0 = %.4f", dx);

printf("\n1.0 / 3.0 = %.12f", dx);

printf("\n1.0 / 3.0 = %.16f", dx);

printf("\n\nPrecision de float: %d", FLT\_DIG); //número de dígitos significativos para un float

printf("\nPrecision de double: %d", DBL\_DIG); //número de dígitos significativos para un double

return 0;

}

Datos de salida:

1.0 / 3.0 = 0.6667

1.0 / 3.0 = 0.666666686535

1.0 / 3.0 = 0.6666666865348816

1.0 / 3.0 = 0.6667

1.0 / 3.0 = 0.666666666667

1.0 / 3.0 = 0.6666666666666666

Precision de float: 6

Precision de double: 15

Los valores FLT\_DIG y DBL\_DIG indican cuántos dígitos decimales significativos se pueden representar con certeza en los tipos float y double, respectivamente. Más allá de estos límites, los valores pueden contener errores de redondeo o ser incorrectos. Los formatos de punto flotante como float y double generalmente incluyen un dígito extra que no es confiable. Este dígito adicional puede ser correcto, pero no está garantizado debido a limitaciones en la representación binaria.

6.

Escribe un programa que pida al usuario ingresar el número de millas recorridas y la cantidad de galones de gasolina consumidos. El programa debe calcular y mostrar el valor de millas por galón, mostrando un decimal. Después, utilizando el hecho de que un galón equivale aproximadamente a 3.785 litros y una milla equivale aproximadamente a 1.609 kilómetros, el programa debe convertir el valor de millas por galón a litros por cada 100 kilómetros, que es la forma habitual en Europa de expresar el consumo de combustible, y mostrar el resultado, también con un decimal. Ten en cuenta que el sistema de los EE. UU. mide la distancia recorrida por cantidad de combustible (mayor es mejor), mientras que el sistema europeo mide la cantidad de combustible por distancia (menor es mejor). Usa constantes simbólicas (utilizando const o #define) para los dos factores de conversión.

#include <stdio.h>

int main(void)

{

float mi, ga;

const float GAL = 3.785, MAK = 1.609; //Factores de conversión de galones a litro (GAL) y millas a kilómetros (MAK).

printf("Ingresa el numero de millas recorridas: ");

scanf("%f", &mi);

printf("Ingresa el numero de galones de gasolina consumidos: ");

scanf("%f", &ga);

printf("Consumo en millas por galon: %.1f", mi / ga);

printf("\nConversion a litros por 100 kilometros: %.1f",

((ga \* GAL) \* 100) / (mi \* MAK) );

return 0;

}

7.

Escribe un programa que convierta tiempo en minutos a tiempo en horas y minutos. Usa #define o const y crea una constante simbólica para 60. Usa un bucle while para permitir al usuario ingresar valores repetidamente y terminar el bucle si se ingresa un valor igual o menor que cero.

#include <stdio.h>

int main(void)

{

const int MINH = 60;

int min = 1;

printf("Ingresa la cantidad de minutos (ingresa 0 para salir): ");

scanf("%d", &min);

while (m > 0)

{

printf("\n%d minuto(s) equivale a %d hora(s) y %d minuto(s)\n", min, min / MINH, min % MINH);

printf("\nIngresa la cantidad de minutos (ingresa 0 para salir): ");

scanf("%d", &min);

}

return 0;

}

8.

Escribe un programa que pide un número entero y luego muestra todos los enteros desde ese valor (incluyéndolo) hasta ese valor aumentado en 10 (incluyéndolo); es decir, si el valor ingresado es 5 el programa escribe todos los números enteros desde 5 hasta 15.

#include <stdio.h>

int main(void)

{

int n, m;

printf("Escribe un numero entero: ");

scanf("%d", &n);

m = n+10;

while (n <= m)

printf("%d ", n++);

return 0;

}

9.

Escribe un programa que pida una altura en centímetros y la convierta en pies y pulgadas. Debe permitir la entrada de valores fraccionarios, y debe permitir al usuario ingresar valores hasta que un número igual o menor que cero sea ingresdado. Ejemplo:

Ingresa una altura en centímetros: **182**

182.0 cm = 5 pies, 11 pulgadas

Ingresa una altura en centímetros (<=0 para salir): **168.7**

168.0 cm = 5 pies, 6.4 pulgadas

Ingresa una altura en centímetros (<=0 para salir): **0**

Adios, puerco.

#include <stdio.h>

int main(void)

{

const float CM\_A\_IN = 0.393701;

const float FT\_A\_IN = 12.0;

float cm, ft, in;

printf("Ingresa una altura en cm: ");

scanf("%f", &cm);

while(cm > 0)

{

ft = (cm \* CM\_A\_IN) / FT\_A\_IN;

in = (cm \* CM\_A\_IN) - ((int) ft \* FT\_A\_IN);

printf("%.1f cm = %d ft, %.1f in", cm, (int)ft, in);

printf("\nIngresa una altura en cm (<=0 para salir): ");

scanf("%f", &cm);

}

printf("Adios, puerco\n");

return 0;

}

10.

El siguiente programa calcula la suma de los primeros 20 numeros enteros (Se puede pensar como un programa que calcula cuánto dinero se obtiene si el primer dia se obtiene $1, el segundo $2, $3 el tercero y así sucesivamente). Modifica el programa para que se le pueda indicar de manera interactiva hasta donde debe continuar la suma de números, es decir, reemplaza el 20 por una variable que sea leida.

#include <stdio.h>

int main(void)

{

int count, sum;

count = 0;

sum = 0;

while (count++ < 20)

sum = sum + count;

printf("sum = %d\n", sum);

return 0;

}

#include <stdio.h>

int main(void)

{

int count, sum, n;

count = 0;

sum = 0;

printf("ingresa un numero para encontrar la suma de los numeros que lo conforman: ");

scanf("%d", &n);

while (count++ < n)

sum = sum + count;

printf("suma = %d\n", sum);

return 0;

}

11.

Ahora modifica el programa para que calcule la suma de los cuadrados de cada entero que conforma el número ingresado (imagina que el primer día recibes $1, el segundo $4, $9 el tercero, y así sucesivamente. ¡Pero que buen trato!). Con n \* n se calcula el cuadrado de un número n.

#include <stdio.h>

int main(void)

{

int count, sum, n;

count = 0;

sum = 0;

printf("Ingresa un numero entero para obtener la suma de los cuadrados de los numeros que lo conforman: ");

scanf("%d", &n);

while (count++ < n)

sum = sum + count \* count;

printf("suma de los cuadrados = %d\n", sum);

return 0;

}

12.

Escribe un programa que pida un número de tipo double y regrese el cubo del número ingresado. Usa una función tuya para elevar al cubo el número e imprimirlo. El programa main() debe pasarle a tu función el valor ingresado.

#include <stdio.h>

void cubo(double m);

int main(void)

{

double n;

printf("Ingresa un numero para obtener su cubo: ");

scanf("%lf", &n);

cubo(n);

return 0;

}

void cubo(double m)

{

m = m \* m \* m;

printf("El cubo del numero ingresado es %lf\n", m);

}

13.

Escribe un programa que demuestre la aplicación de la operación módulo. El usuario debe ingresar primero un número entero que será utilizado como el segundo operando, luego, el usuario ingresará otro número para calcular el módulo (primer operando); y cuando se ingrese 0 o un número menor, el programa finalizará. Ejemplo de salida:

Este programa calcula modulos.

Ingresa el numero entero que servira como segundo operando: **256**

Ahora ingresa el primer operando: **438**

438 % 256 es igual a 182

Ingresa el siguiente operando primero (<=0 para salir): **1234567**

1234567 % 256 es igual a 135

Ingresa el siguiente operando primero (<=0 para salir): **0**

Bababooey

#include <stdio.h>

int main(void)

{

int segundo, primero;

printf("Este programa calcula modulos.\n");

printf("Ingresa el numero entero que servira como segundo operando: ");

scanf("%d", &segundo);

printf("Ahora ingresa el primer operando: ");

scanf("%d", &primero);

while(primero > 0)

{

printf("%d %% %d es igual a %d\n", primero, segundo, primero % segundo);

printf("ingresa el siguiente operando primero (<=0 para salir): ");

scanf("%d", &primero);

}

printf("Bababooey");

return 0;

}

14.

Escribe un programa que le solicite al usuario ingresar una temperatura en Fahrenheit. Debe leer la temperatura como un numero de tipo double y pasarlo como argumento a una función llamada Temperaturas(). Esta función debe calcular las conversiones a Celsius y a Kelvin y escribir las tres temperaturas con una precisión de dos dígitos decimales. Debe identificar cada valor de temperatura con la letra (símbolo) correspondiente: F para ahrenheit, C para Celsius y K para Kelvin. Fórmulas de conversión:

Celsius = 5.0 / 9.0 \* (Fahrenheit – 32.0)

Kelvin = Celsius + 273.16

La función Temperaturas() debe usar const para crear representaciones simbólicas de las tres constantes que aparecen en las formulas de conversión. La función main() debe usar un bucle que permita al usuario ingresar temperaturas de manera repetida, hasta que se ingrese una q u otro valor no numérico; para lograrlo use el hecho de que scanf regresa el número de elementos leidos, tal que si lee un número va a regresar 1, pero no regresará 1 si se ingresa q. El operador == comprueba la equivalencia, asi que puede usarse para comparar el valor de retorno de scanf con 1.

#include <stdio.h>

void Temperaturas(double n);

int main(void)

{

double fahrenheit;

printf("Ingresa una temperatura en Fahrenheit: ");

while (scanf("%lf", &fahrenheit) == 1)

{

Temperaturas(fahrenheit);

printf("Ingresa una temperatura en fahrenheit (q para salir): ");

}

return 0;

}

void Temperaturas(double n)

{

const double CELSIUS = (5.0 / 9.0) \* ( n -32);

const double KELVIN = CELSIUS + 273.16;

const double FAHRENHEIT = 9.0 / 5.0 \* CELSIUS + 32;

printf("%.1lf grados Fahrenheit es igual a %lf grados Celsius, y a %lf Kelvin\n",

FAHRENHEIT, CELSIUS, KELVIN);

}

asdf