APRENDERAPROGRAMAR.COM

EJERCICIOS RESUELTOS EN C CON IF THEN ELSE. PAR O IMPAR. ECUACIÓN CUADRÁTICA. PARADOJAS. (CU00530F)

Sección: Cursos
Categoría: Curso básico de programación en lenguaje C desde cero

Fecha revisión: 2031

Resumen: Entrega nº30 del curso básico "Programación C desde cero".

Autor: Mario Rodríguez Rancel

EJERCICIO NÚMERO 1: ENUNCIADO

Para practicar el uso de condicional if … else en C vamos a desarrollar una serie de ejercicios resueltos. El primer ejercicio plantea lo siguiente:

Transformar en código el siguiente pseudocódigo, relativo a determinar la naturaleza par o impar de un número. ***Nota:*** El procedimiento a emplear será basado en el uso del operador módulo, %.

Pseudocódigo aludido:

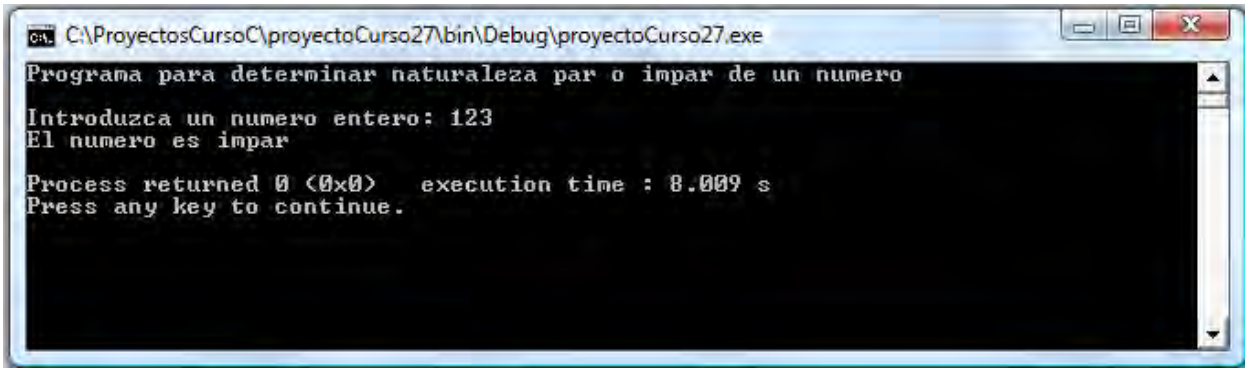
```
1. Inicio [Pseudocódigo aprenderaprogramar.com]
        2. Mostrar "Introduzca un número" : Pedir Num
        3. Res = Num mod 2
        4. Si Res = 0 Entonces
                Mostrar "El número es par"
           SiNo
                Mostrar "El número es impar"
           FinSi
5. Fin
```

SOLUCIÓN

Todas las instrucciones y operadores necesarios las hemos visto en apartados anteriores. Pueden utilizarse los nombres de variable que se estimen oportunos (no tienen por qué coincidir con lo que ponemos aquí).

```
#include <stdio.h>
#include <stdlib.h>
// Ejercicios resueltos aprenderaprogramar.com
int main() {
   int Num; int Res;
   printf("Programa para determinar naturaleza par o impar de un numero\n\n");
   printf ("Introduzca un numero entero: ");
   scanf ("%d", &Num);
   Res = Num%2;
   if (Res==0) {
      printf ("El numero es par\n");
   } else {
      printf ("El numero es impar\n");
   }
   return 0;
}
```

Un resultado de ejecución puede ser similar a este:



Hemos escrito el *if ... – else* como un bloque (usando llaves) en vez de en línea. El motivo para ello es que aporta mayor claridad y facilidad de interpretación.

Recomendamos que los nombres de variables comiencen con una letra minúscula, pero como podemos comprobar no es obligatorio. Normalmente los programadores siguen ciertas pautas o normas a la hora de poner nombres a las variables. Una de ellas es la llamada norma "camelCase" basada en comenzar con una minúscula e introducir una mayúscula con cada cambio de palabra. Por ejemplo: numeroDeCoches ó diasDeFiestaLocal.

EJERCICIO NÚMERO 2: ENUNCIADO

Se quiere crear un programa que resuelva la ecuación cuadrática tipo $ax^2 + bx + c$. Para ello se ha estudiado el problema, que se ha planteado en pseudocódigo. Transformar el pseudocódigo que se muestra a continuación en código C y comprobar que el programa obtiene los resultados esperados para una serie de casos de prueba.

```
1. Inicio  [Pseudocódigo aprenderaprogramar.com]
2. [Resolución de ecuación cuadrática ax² + bx + c = 0]
        3. Mostrar "Introduzca los valores de parámetros"
        4. Pedir a, b, c
        5. d = b ^ 2 – 4 * a * c : e = 2 * a
        6. Si d = 0 Entonces
                Mostrar "x₁ = x₂ =", – b / e
           SiNo
                Si d > 0 Entonces
                    Mostrar "x₁ =", (– b + SQR(d)) / e
                    Mostrar "x₂ =", (– b – SQR(d)) / e
                SiNo
                    Mostrar "x₁ =", – b / e, "+", SQR(– d) / e, "i"
                    Mostrar "x₂ =", – b / e, "–", SQR(– d) / e, "i"
                FinSi
           FinSi
7. Fin
```

EJERCICIO NÚMERO 2: SOLUCIÓN

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
// Ejercicios resueltos aprenderaprogramar.com
int main() {
   //Declaración de variables
   double a, b, c, d, e;
   printf("Programa para calcular ecuacion cuadratica tipo a*x^2 + b*x + c = 0\n\n");
   //Obtención de datos
   printf ("Introduzca valor parametro a: ");    scanf ("%lf", &a);
   printf ("Introduzca valor parametro b: ");    scanf ("%lf", &b);
   printf ("Introduzca valor parametro c: ");    scanf ("%lf", &c);
   //Cálculo y muestra de resultados
   d = pow (b,2) - 4 * a * c;    e = 2 * a;
   if (d == 0) { printf ("El resultado es x1 = x2 =%lf", - b / e);
   } else {
      if (d > 0) {
         printf ("El resultado es:\n");
         printf ("x1 = %lf \n", (-b + sqrt(d)) / e);
         printf ("x2 = %lf \n", (-b - sqrt(d)) / e);
      } else {
         printf ("El resultado es: \n");
         printf ("x1 = %lf + %lf * i \n", -b / e, sqrt(-d)/e);
         printf ("x2 = %lf - %lf * i \n", -b / e, sqrt(-d)/e);
      }
   }
   return 0;
}
```

El programa "funciona", si bien tiene sus limitaciones. Prueba a introducir los siguientes datos: *a = 0*, *b = 3* y *c = 1* . Te aparecerá un error o resultado erróneo similar a: *"El resultado es: x1 = -1.#IND00 x2 = -1.#INF00".* Efectivamente, *a = 0* implica que *e = 0* y al encontrarnos con una operación del tipo *(–b + sqrt(d)) / e* donde el denominador vale cero... ¡error y se acabó! Mejor dicho, "se acabó" mientras no preparemos el programa para detectar este tipo de situaciones, cosa que como veremos más adelante se puede hacer, o diríamos más, sería aconsejable hacer.

Estamos construyendo programas sencillos que en líneas generales responden bien cuando se dan las circunstancias previstas pero que pueden fallar ante situaciones extrañas. Esto no debe preocuparnos por el momento, ya que nos estamos centrando en la lógica y algoritmia básica de la programación. A medida que progresemos como programadores deberíamos ser capaces de mejorar nuestros planteamientos. Igualmente, deberemos complementar nuestra formación en áreas como el tratamiento y gestión de errores (excepciones o circunstancias no controladas).

Un ejemplo de ejecución del programa sería el siguiente:

```
Programa para calcular ecuacion cuadratica tipo a*x^2 + b*x + c = 0
Introduzca valor parametro a: -3
Introduzca valor parametro b: 1
Introduzca valor parametro c: 2
El resultado es: x1 = -0.666667 x2 = 1.000000
```

PARADOJA DEL SI BURLADO POR UN INTERVALO

Prueba los siguientes códigos a los que denominaremos versión 1 y versión 2:

```
#include <stdio.h>
#include <stdlib.h>
int main() {
   int A; A=160;
   if (50<A<75 ) {
      printf ("Arabia\n");
   } else {
      printf ("Eusebio\n");
   }
   return 0;
}
```

```
#include <stdio.h>
#include <stdlib.h>
int main() {
   int A; A=160;
   if (A>50 && A<75 ) {
      printf ("Arabia\n");
   } else {
      printf ("Eusebio\n");
   }
   return 0;
}
```

Si bien podría esperarse "con sentido matemático" obtener el mismo resultado con ambos códigos, al ejecutar la versión 1 obtenemos un error del tipo "*aviso: las comparaciones como 'X<=Y<=Z' no tienen su significado matemático [-Wparentheses]*", mientras que en la versión 2 se muestra *"Eusebio"*. La razón: C no admite comparaciones dobles del tipo $x < y < z$. Ante este tipo de escritura, obtendremos un error. Por tanto debemos recordar que aunque existen similitudes no todas las expresiones matemáticas son directamente trasladables a un lenguaje de programación.

EJERCICIO

Un else puede ir seguido de un if dando lugar a lo que se conoce como una estructura else if. Un if seguido de if else equivale a decir "Si ocurre esto ... haz esto ... y si no comprueba si ocurre esto otro ... Se pueden añadir tantos else if como se desee (y finalmente puede haber un else para contemplar el caso de que no se cumpla ninguna de las condiciones). Estudia este código y responde a las cuestiones indicadas más abajo:

:

```
#include <stdio.h>
 // Ejercicio aprenderaprogramar.com
int main () {
  /* local variable definition */
  int a = 100;
  if( a == 10 ) {
    printf("Value of a is 10\n" );
  }
  else if( a == 20 ) {
    printf("Value of a is 20\n" );
  }
  else if( a == 30 ) {
    printf("Value of a is 30\n" );
  }
  else {
    printf("None of the values is matching\n" );
  }
  printf("Exact value of a is: %d\n", a );
  return 0;
}
```

a) Describe qué es lo que hace este programa detalladamente.

b) ¿Cuál es el resultado si definimos a con valor 20? ¿Cuál es el resultado si definimos a con valor 30? ¿Cuál es el resultado si definimos a con valor 40?

c) Elimina el último else que aparece en el código. ¿Cuál es el resultado ahora si definimos a con valor 20? ¿Cuál es el resultado ahora si definimos a con valor 30? ¿Cuál es el resultado ahora si definimos a con valor 40? ¿Por qué obtenemos ahora estos resultados?

Para comprobar si tus respuestas son correctas puedes consultar en los foros aprenderaprogramar.com.

Próxima entrega: CU00531F

Acceso al curso completo en aprenderaprogramar.com -- > Cursos, o en la dirección siguiente:
http://www.aprenderaprogramar.com/index.php?option=com_content&view=category&id=82&Itemid=210