APRENDERAPROGRAMAR.COM

AMBITO DE VARIABLES EN C. GLOBALES Y LOCALES. UNDECLARED (FIRST USE IN THIS FUNCTION). EJEMPLOS (CU00548F)

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

Fecha revisión: 2031

Resumen: Entrega nº48 del curso básico "Programación C desde cero".

Autor: Mario Rodríguez Rancel

VARIABLES GLOBALES Y VARIABLES LOCALES

Nos interesamos ahora por el concepto de variable global y local (explicado en el curso Bases de la programación nivel I de aprenderaprogramar.com) y su traslación al lenguaje C.

En primer lugar, usaremos el término *ámbito* para referirnos a aquel conjunto de partes del programa en el que una variable es conocida. Así distinguiremos:

Variables con ámbito una función

Son declaradas dentro de la función y sólo son conocidas dentro de ella, de ahí que se denominen variables locales a la función. Las declararemos normalmente al principio de la función (cabecera de la función), aunque podrían introducirse en un punto intermedio. No puede invocarse la variable sin que antes se haya declarado. En general, para una mejor ordenación del programa siempre será preferible que las declaraciones sean en cabecera.

Variables con ámbito el programa

Son declaradas en la cabecera del programa, antes de la declaración de funciones y antes del método *main*. Decimos que son variables globales porque estas variables son conocidas por todas las funciones en el programa, incluida la función *main*.

Prueba el siguiente código:

```
#include <stdio.h>
#include <stdlib.h>
int varGlob = 9;
void ejemplo() {
   int varLoc = 5;
   printf("varLoc es una variable local y vale %d\n", varLoc);
   printf("varGlob*varLoc vale %d\n", varGlob*varLoc);
}

int main() {
   ejemplo(); // Ejemplo aprenderaprogramar.com
   printf("varGlob es una variable global y vale %d\n", varGlob);
   //Genera error printf("varLoc es una variable local y vale %d\n", i);
   return 0;
}
```

En este ejemplo comprobamos que la variable *varLoc* es una variable local a la función ejemplo, por lo que no es conocida en el resto de funciones del programa. La variable *varGlob* es una variable global y tenemos acceso a ella en todas las funciones del programa.

EJERCICIO RESUELTO Nº1: ENUNCIADO

Transformar en código el siguiente pseudocódigo y razonar para comprender su lógica. Se trata de un ejemplo de programación modular que debemos desarrollar en C mediante el uso de funciones.

```
PROGRAMA SUC 01
   Variables
         Reales: a, Suma
         1. Inicio
               2. Llamar EntraDatos
               3. Llamar Calculo
               4. Mostrar "El valor del sumatorio es", Suma
         5. Fin

         Módulo EntraDatos
               1. Mientras a <= 0 ó a > 100 Hacer
                     1.1 Mostrar "Por favor introduzca un número entero comprendido
                           entre 1 y 100"
                     1.2 Pedir a
                     1.3 a = Redondear(a)
                  Repetir
               2. Mostrar "El dato base es", a
         FinMódulo

         Módulo Calculo
               Variables
                     Enteras: i
               1. Hacer
                     1.1 Suma = Suma + 1 / a
                     1.2 a = a – 1
                     1.3 i = i + 1
                  Repetir Mientras a <> 0
               2. Mostrar "Contabilizados", i, "términos"
         FinMódulo [Ejemplo aprenderaprogramar.com]
```

EJERCICIO RESUELTO Nº1: SOLUCIÓN

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

double a=0.0; double suma=0.0;

void entraDatos(); void calculo();

int main() {
   entraDatos();
   calculo();
   printf("El valor del sumatorio es: %lf\n\n ", suma);
   return 0;
   //Ejemplo aprenderaprogramar.com
}
```

```
void entraDatos() {
   while (a <= 0 || a > 100) {
      printf("Por favor introduzca un numero entero comprendido entre 1 y 100: ");
      scanf("%lf", &a);
      a = floor(a);
   }
   printf("El dato base es: %.0lf\n\n", a); return;
}

void calculo() {
   int i=0; // Ejemplo aprenderaprogramar.com
   do {
      suma = suma + 1/a; a = a-1; i = i+1;
   } while ( a != 0);
   printf("Contabilizados %d terminos \n\n", i); return;
}
```

Este podría ser un resultado de ejecución:

```
Por favor introduzca un numero entero comprendido entre 1 y 100: 9
El dato base es: 9.000000 - Contabilizados 9 terminos - El valor del sumatorio es: 2.828968
```

El programa trabaja con tres variables: *a*, *suma* y *i*. Las variables *a* y *suma* son de tipo *double*, y por estar declaradas en la cabecera del programa son conocidas en todas las funciones (decimos que son variables globales). La variable *i* es de tipo entero y por estar declarada en la cabecera de una función es una variable local de dicha función. Si pruebas a escribir en el main lo siguiente:

printf("El valor del sumatorio es: %lf con %d terminos\n\n ", suma, i);

Obtendrás un error del tipo <<*error: 'i' undeclared (first use in this function)*>>. En realidad la variable *i* sí está definida, pero tiene un ámbito restringido y no podemos usarla en el sitio donde hemos tratado de hacerlo.

Sugerencia: reescribe el código para que la función *entraDatos* devuelva el dato a emplear y la función *calculo* devuelva el resultado del cálculo de modo que no sea necesario el uso de variables globales.

EJERCICIO

Estudia el código que se muestra más abajo y responde a las preguntas:

a) Sin ejecutar el código (sólo pensando) responde: ¿para qué sirve la función check_prime? ¿Qué resultado devolverá si le pasamos el número 4? ¿Por qué?

b) En dicha función se usa la siguiente definición de bucle: for(j=2;j<=num/2;++j) ¿Por qué crees que el bucle tiene un valor inicial 2? ¿Por qué crees que el bucle tiene un valor final num/2?

c) ¿La variable flag que se usa en el main es la misma que se usa en la función check_prime? ¿Es posible declarar y usar dos variables con el mismo nombre en la función main? ¿Y declarar y usar dos variables con el mismo nombre en la función main y otra función? ¿Por qué?.

d) Crea una tabla de variables del programa que conste de las siguientes columnas: nombre de variable, ámbito, utilidad. Por ejemplo la variable i tiene nombre de variable i, ámbito la función main, y utilidad servir como índice en el bucle for del main.

e) Si se introducen como números 11 y 30 ¿Son evaluados 11 y 30? ¿Qué resultado se obtiene? ¿Por qué?

```
#include <stdio.h>
#include <stdlib.h>

int check_prime(int num);

int main(){
  int n1,n2,i,flag;
  printf("Enter two numbers(intervals): ");
  scanf("%d %d",&n1, &n2);
  printf("Prime numbers between %d and %d are: ", n1, n2);
  for(i=n1+1;i<n2;++i)
  {
    flag=check_prime(i);
    if(flag==0)
      printf("%d ",i);
  }
  return 0;
}

int check_prime(int num) {
  int j,flag=0;
  for(j=2;j<=num/2;++j){
     if(num%j==0){
        flag=1;
        break;
     }
  }
  return flag; // Ejercicios aprenderaprogramar.com
}
```

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

Próxima entrega: CU00549F

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