	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: Karina García Morales

Asignatura: Fundamentos de programación

Grupo: 20

No. de práctica(s): 11

Integrante(s): Suzán Herrera Álvaro

No. de lista o brigada: 49

Semestre: 1

Fecha de entrega: 13 de diciembre 2022

Observaciones:

CALIFICACIÓN: _____

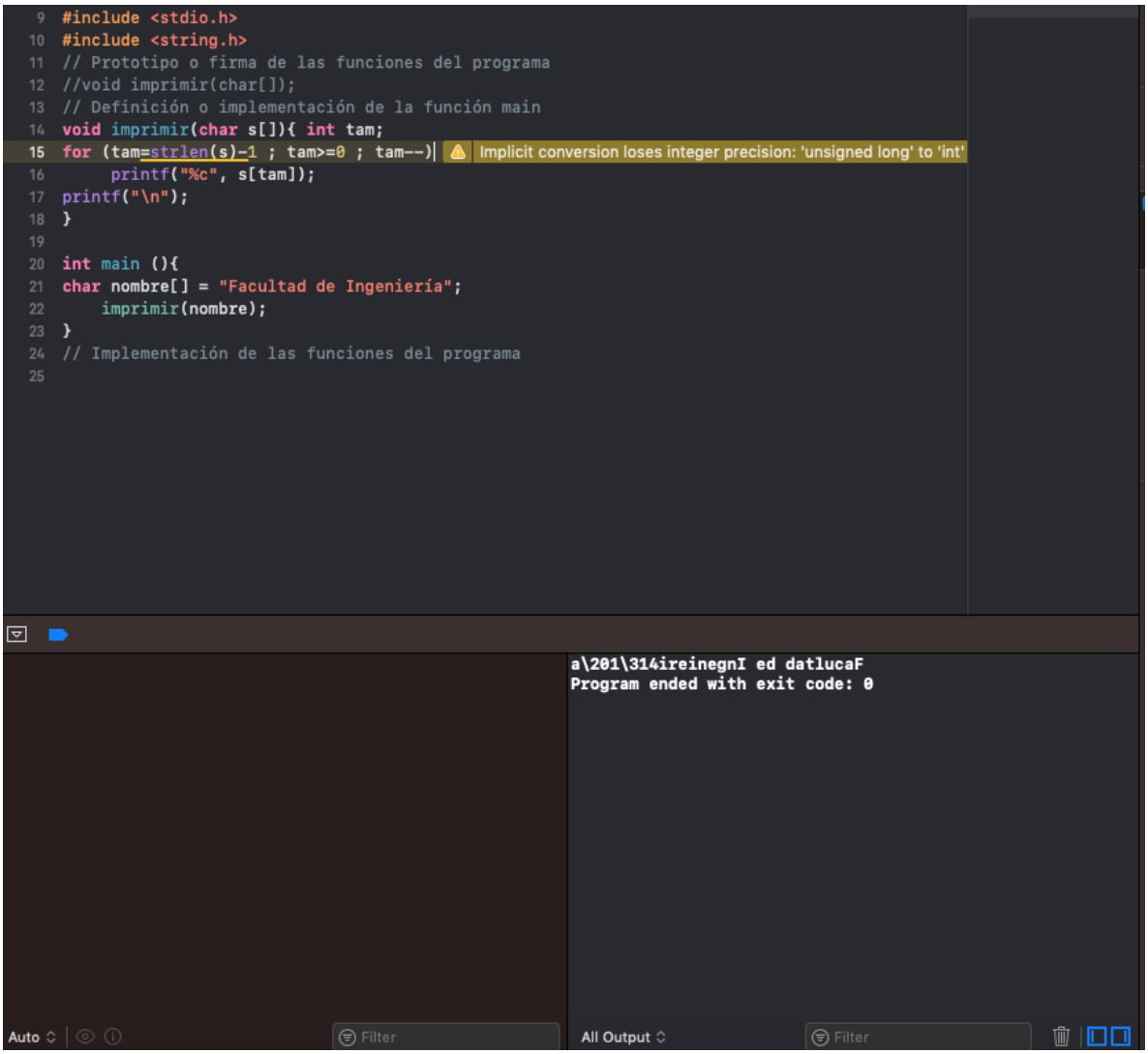
Objetivo: El alumno elaborará programas en C donde la solución del problema se divida en funciones. Distinguir lo que es el prototipo o firma de una función y la implementación de ella, así como manipular parámetros tanto en la función principal como en otras.

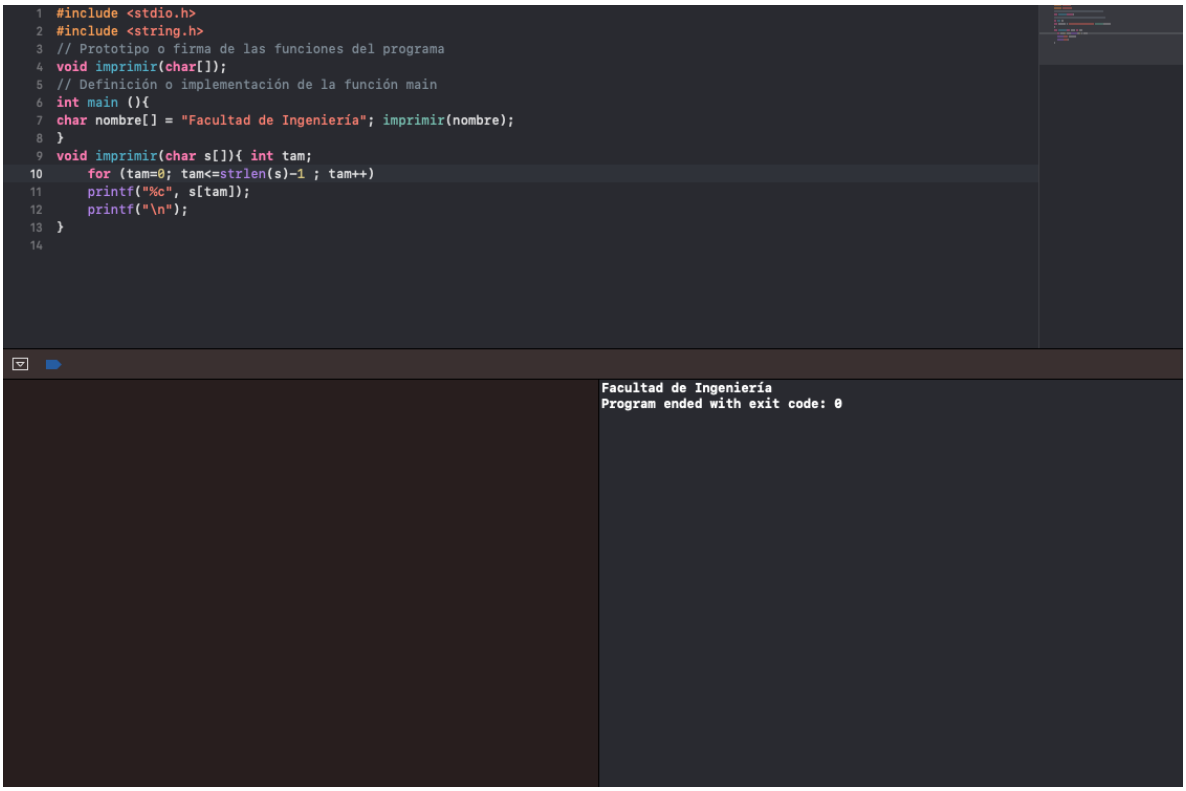
Conceptos

Funciones: Elemento agrupador de un bloque de acciones que puede utilizarse para realizar operaciones y con ello ahorrar tiempo en código

- TipoDato fnNombre (entradas){• • •}

Código (funciones)
El siguiente programa contiene dos funciones: la función main y la función imprimir. La función main manda llamar a la función imprimir. La función imprimir recibe como parámetro un arreglo de caracteres y lo recorre de fin a inicio imprimiendo cada carácter del arreglo.

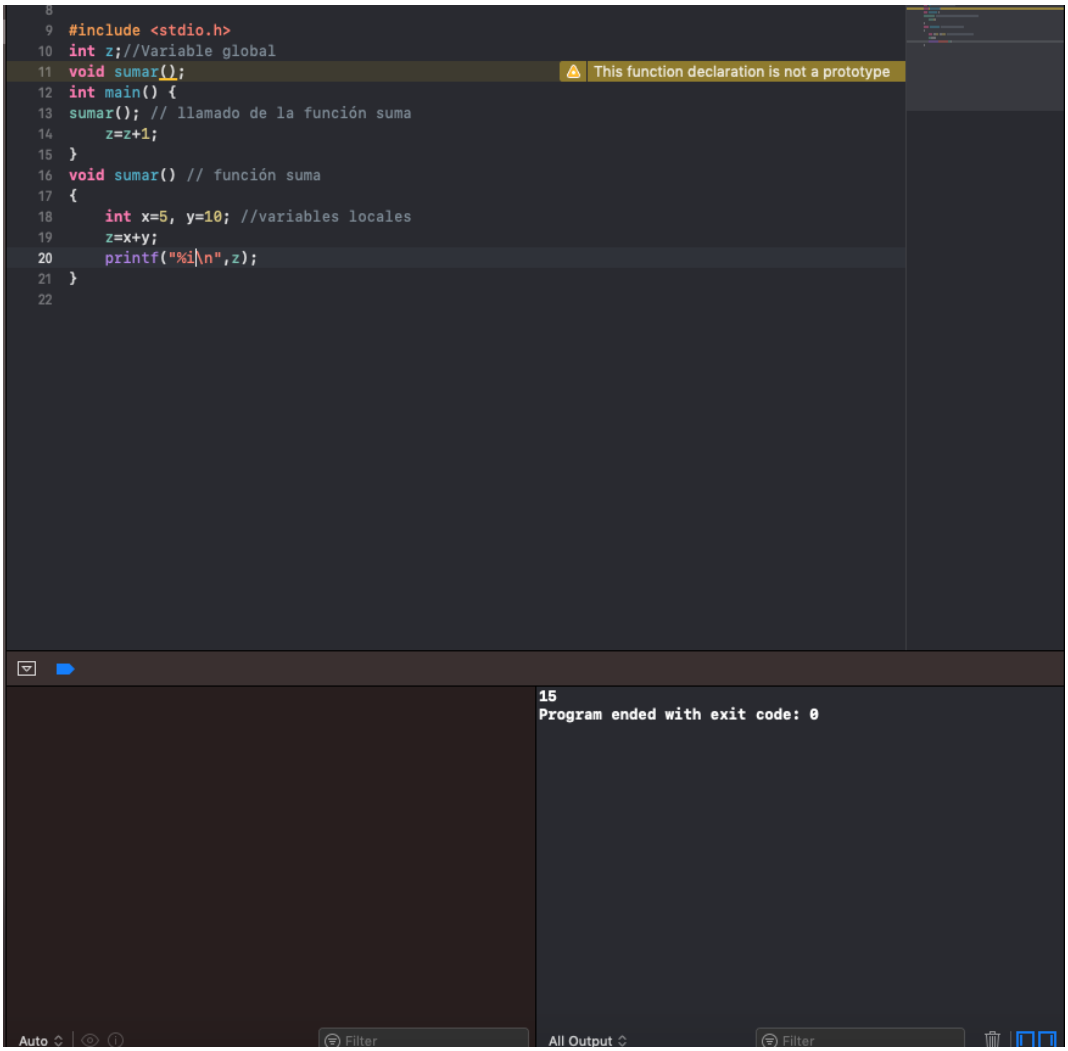

#include <stdio.h>
#include <string.h>
// Prototipo o firma de las funciones del programa
//void imprimir(char[]);
// Definición o implementación de la función main
void imprimir(char s[]){ int tam;
for (tam=strlen(s)-1 ; tam>=0 ; tam--)
Implicit conversion loses integer precision: 'unsigned long' to 'int'
printf("%c", s[tam]);
printf("\n");
}
int main (){
char nombre[] = "Facultad de Ingeniería";
imprimir(nombre);
}
// Implementación de las funciones del programa
a\201\314ireinegnI ed datlucaF
Program ended with exit code: 0
Auto
Filter
All Output
Filter



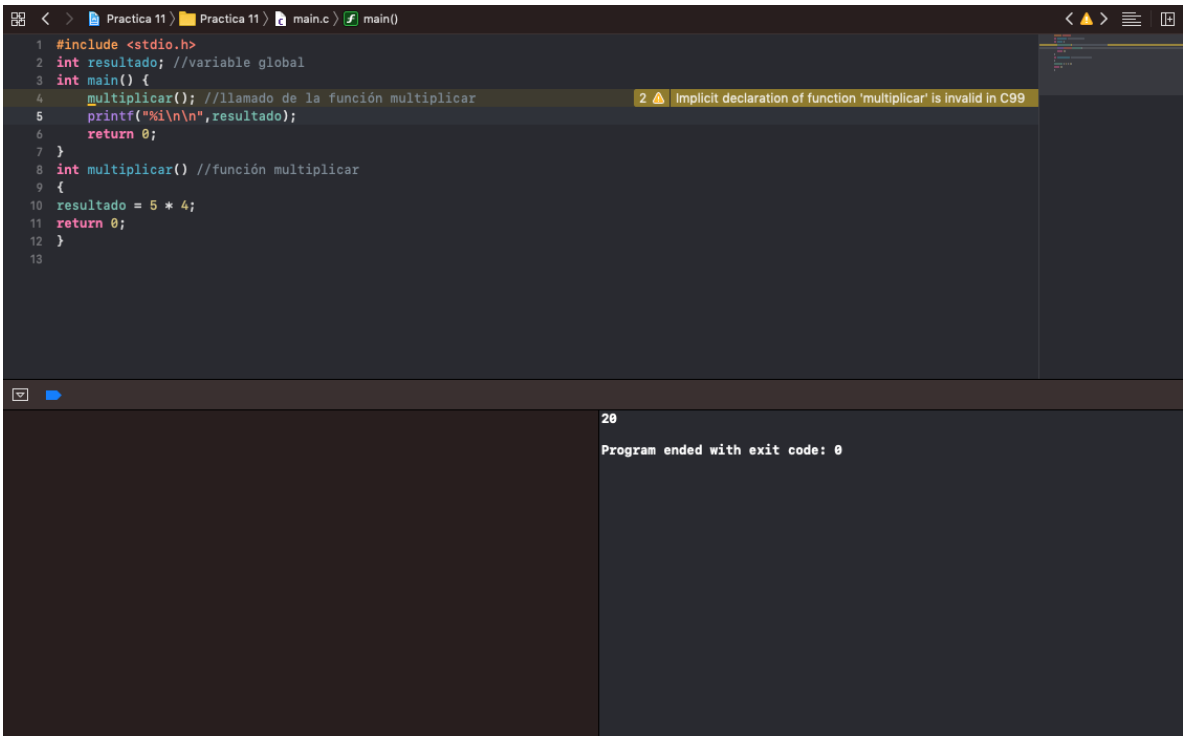
```c
#include <stdio.h>
#include <string.h>
// Prototipo o firma de las funciones del programa
void imprimir(char[]);
// Definición o implementación de la función main
int main (){
char nombre[] = "Facultad de Ingeniería"; imprimir(nombre);
}
void imprimir(char s[]){ int tam;
    for (tam=0; tam<=strlen(s)-1 ; tam++)
    printf("%c", s[tam]);
    printf("\n");
}

```

```
Facultad de Ingeniería
Program ended with exit code: 0
```

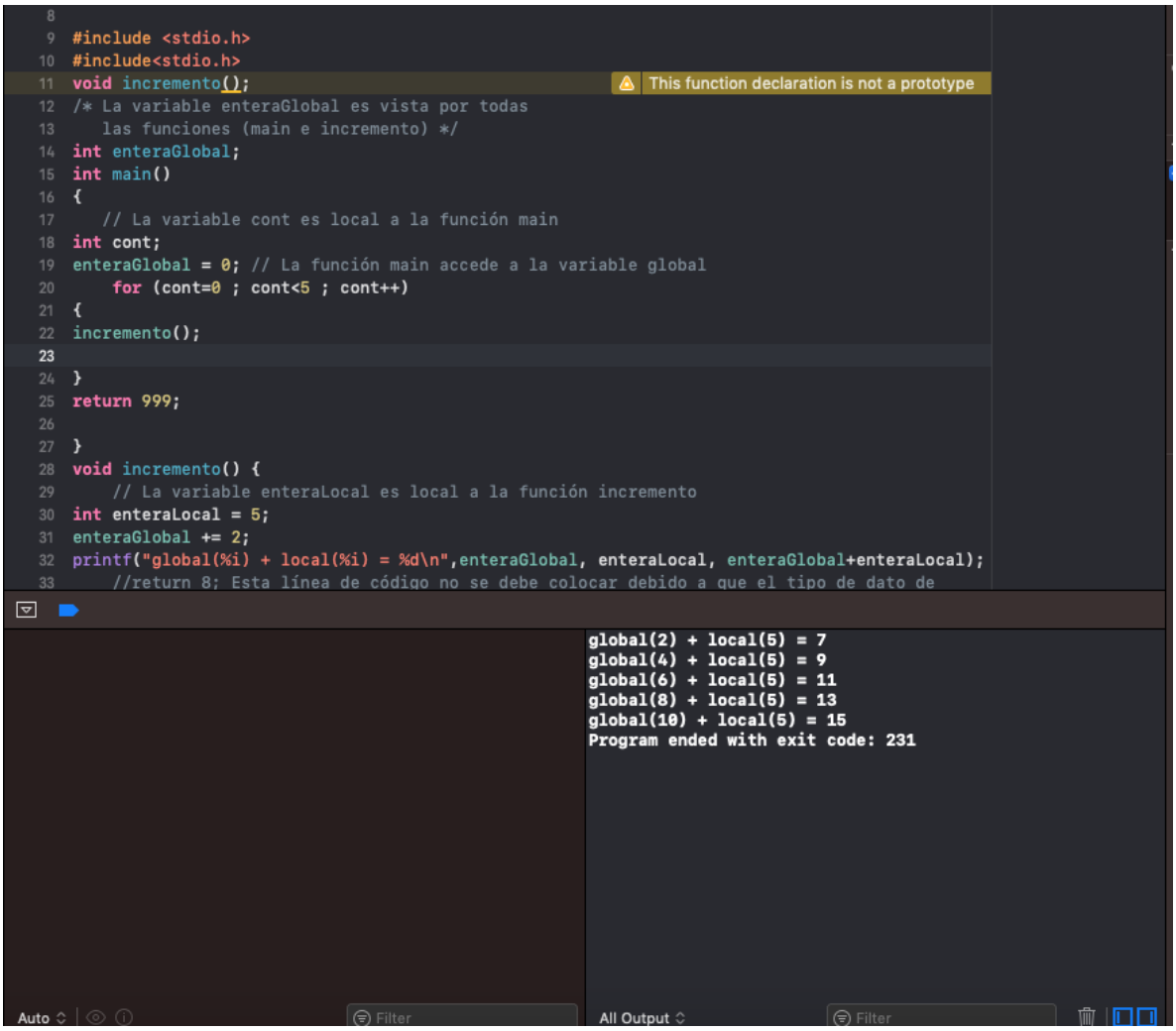
Código (variables locales) El siguiente programa muestra la declaración y uso de variables locales en la función de suma.



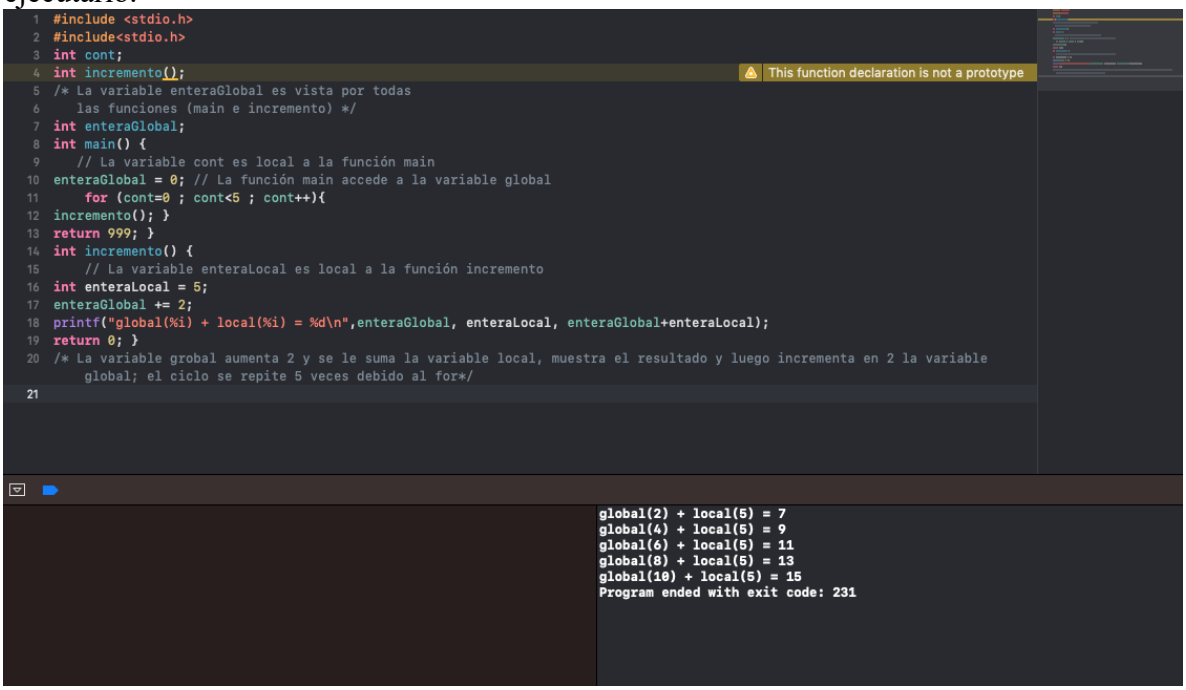
Código (variables globales) El siguiente programa muestra la declaración de la variable global resultado, la cual es utilizada en ambas funciones.


Practica 11
Practica 11
main.c
main()
#include <stdio.h>
int resultado; //variable global
int main() {
multiplicar(); //llamado de la función multiplicar
2 Implicit declaration of function 'multiplicar' is invalid in C99
printf("%i\n\n",resultado);
return 0;
}
int multiplicar() //función multiplicar
{
resultado = 5 * 4;
return 0;
}
20
Program ended with exit code: 0

Código (Ámbito de las variables) Este programa contiene dos funciones: la función main y la función incremento. La función main manda llamar a la función incremento dentro de un ciclo for. La función incremento aumenta el valor de la variable enteraGlobal cada vez que es invocada.



Código (argumentos función main) Este programa muestra los argumentos enviados al ejecutarlo.

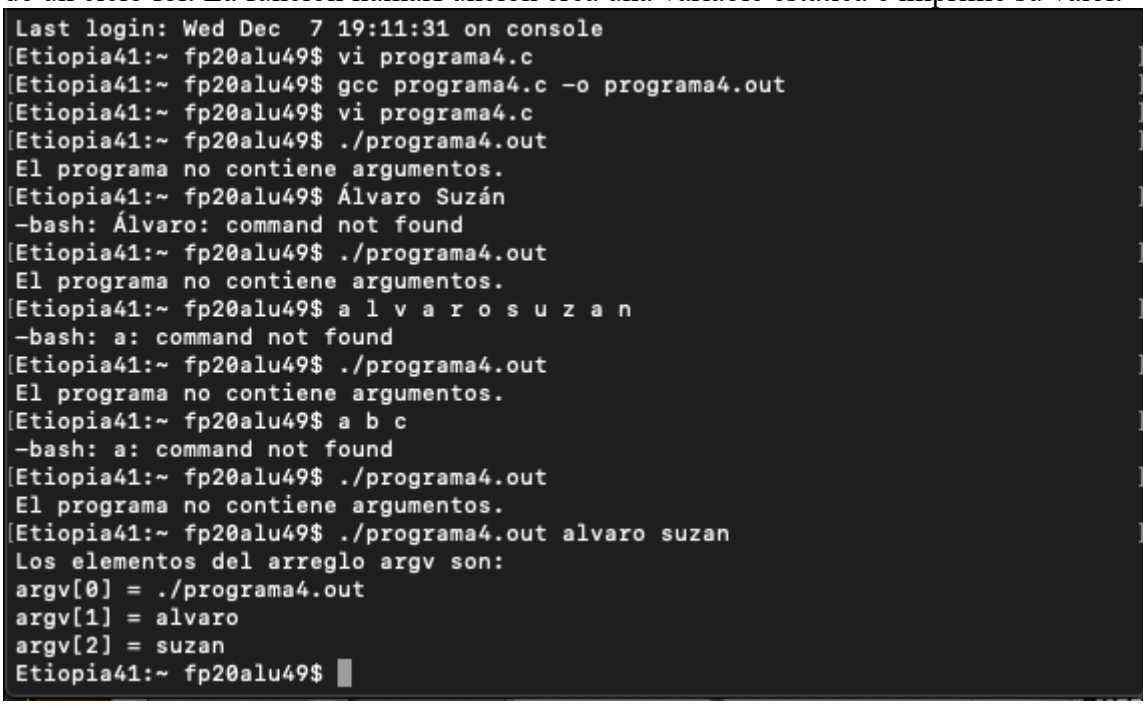


```c
#include <stdio.h>
#include<stdio.h>
int cont;
int incremento();
/* La variable enteraGlobal es vista por todas
   las funciones (main e incremento) */
int enteraGlobal;
int main() {
   // La variable cont es local a la función main
enteraGlobal = 0; // La función main accede a la variable global
    for (cont=0 ; cont<5 ; cont++){
incremento(); }
return 999; }
int incremento() {
    // La variable enteraLocal es local a la función incremento
int enteraLocal = 5;
enteraGlobal += 2;
printf("global(%i) + local(%i) = %d\n",enteraGlobal, enteraLocal, enteraGlobal+enteraLocal);
return 0; }
/* La variable grobal aumenta 2 y se le suma la variable local, muestra el resultado y luego incrementa en 2 la variable
    global; el ciclo se repite 5 veces debido al for*/
```

This function declaration is not a prototype

```
global(2) + local(5) = 7
global(4) + local(5) = 9
global(6) + local(5) = 11
global(8) + local(5) = 13
global(10) + local(5) = 15
Program ended with exit code: 231
```

Código (variable estática) Este programa contiene dos funciones: la función main y la función llamarFuncion. La función main manda llamar a la función llamarFuncion dentro de un ciclo for. La función llamarFuncion crea una variable estática e imprime su valor.



```
Last login: Wed Dec  7 19:11:31 on console
[Etiopia41:~ fp20alu49$ vi programa4.c
[Etiopia41:~ fp20alu49$ gcc programa4.c -o programa4.out
[Etiopia41:~ fp20alu49$ vi programa4.c
[Etiopia41:~ fp20alu49$ ./programa4.out
 El programa no contiene argumentos.
[Etiopia41:~ fp20alu49$ Álvaro Suzán
 -bash: Álvaro: command not found
[Etiopia41:~ fp20alu49$ ./programa4.out
 El programa no contiene argumentos.
[Etiopia41:~ fp20alu49$ a l v a r o s u z a n
 -bash: a: command not found
[Etiopia41:~ fp20alu49$ ./programa4.out
 El programa no contiene argumentos.
[Etiopia41:~ fp20alu49$ a b c
 -bash: a: command not found
[Etiopia41:~ fp20alu49$ ./programa4.out
 El programa no contiene argumentos.
[Etiopia41:~ fp20alu49$ ./programa4.out alvaro suzan
 Los elementos del arreglo argv son:
 argv[0] = ./programa4.out
 argv[1] = alvaro
 argv[2] = suzan
 Etiopia41:~ fp20alu49$
```

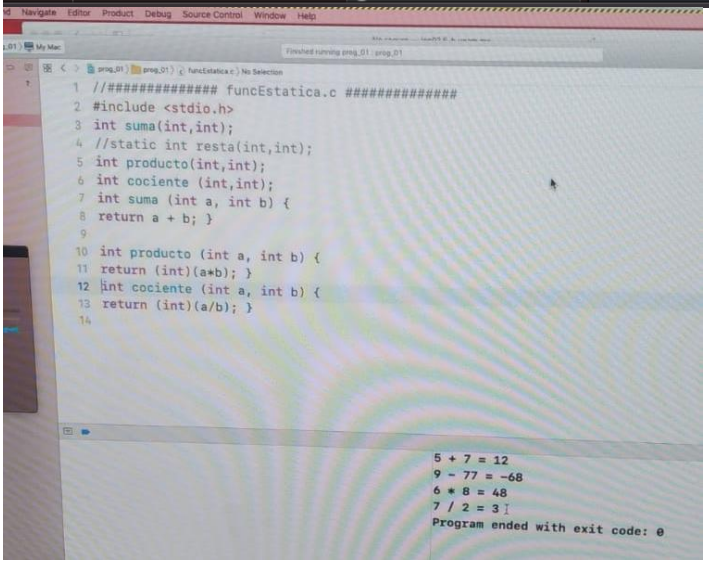
Código (función estática). Este ejemplo consta de dos archivos: funcEstatica.c y calculadora.c. El programa funcEstatica.c contiene las funciones de una calculadora básica: suma, resta, producto y cociente

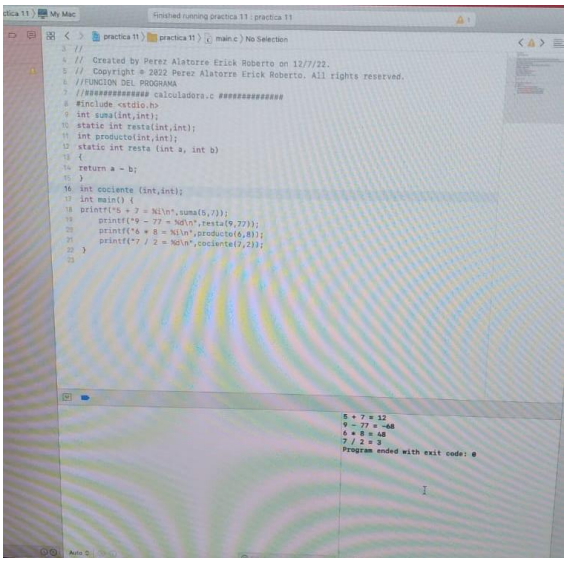
```c
//
#include <stdio.h>
void llamarFuncion();
int main () {
for (int j=0 ; j < 5 ; j++) {
llamarFuncion(); }
}
void llamarFuncion() {
/* Solo la primera vez que se llame a esta función se creará y se le asignará el valor de 0
    a la variable estática numVeces */
static int numVeces = 0;
printf("Esta función se ha llamado %d veces.\n",++numVeces);
}

```

This function declaration is not a prototype

```
Esta función se ha llamado 1 veces.
Esta función se ha llamado 2 veces.
Esta función se ha llamado 3 veces.
Esta función se ha llamado 4 veces.
Esta función se ha llamado 5 veces.
Program ended with exit code: 0
```

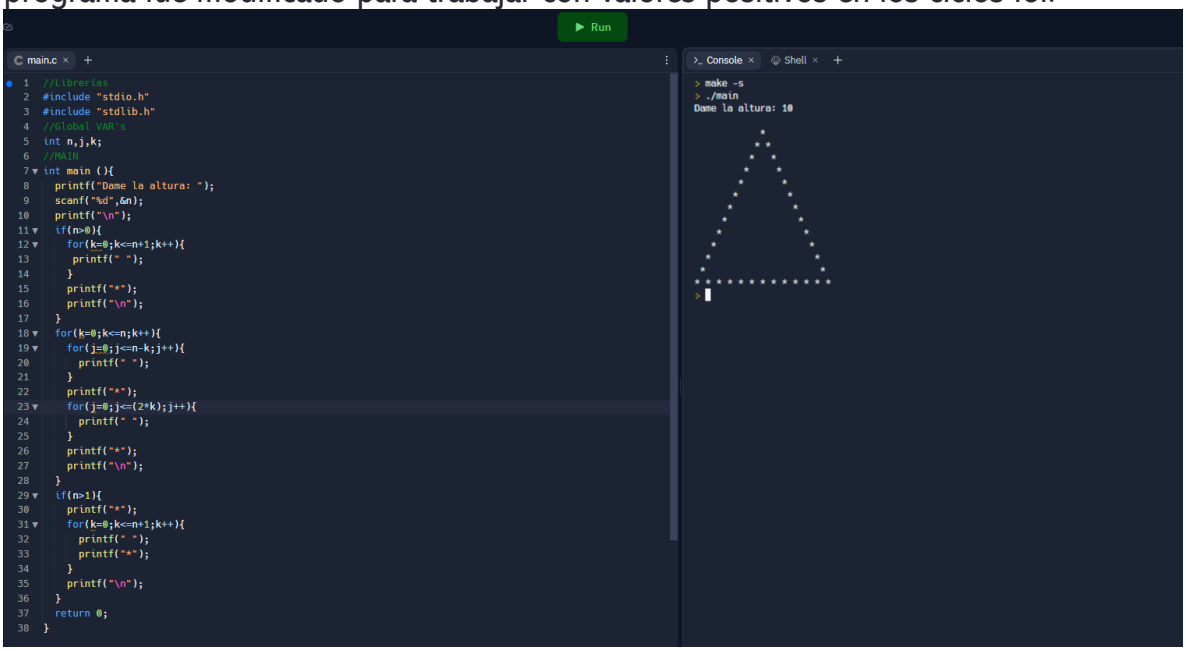
Auto | Filter | All Output | Filter





Tarea

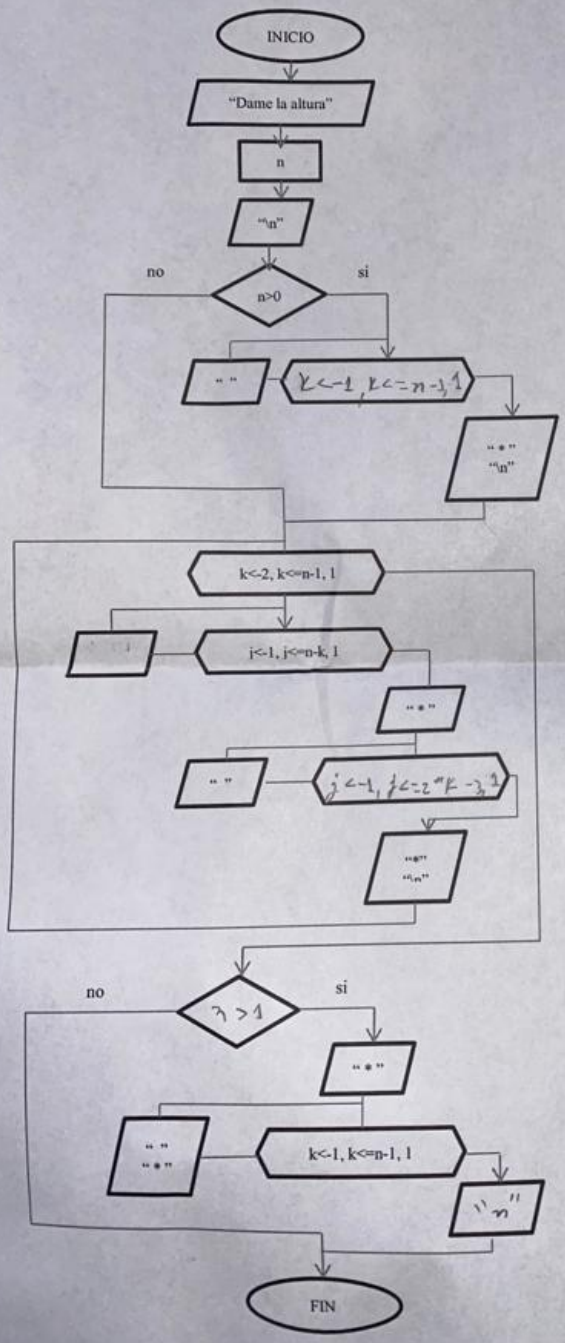
Pruebas de escritorio; programa que muestra un triángulo en ASCII recopilando solamente la altura, el diagrama y el pseudocódigo se completan y se analizan; el programa fue modificado para trabajar con valores positivos en los ciclos for.




```c
//Librerias
#include "stdio.h"
#include "stdlib.h"
//Global VAR's
int n,j,k;
//MAIN
int main (){
  printf("Dame la altura: ");
  scanf("%d",&n);
  printf("\n");
  if(n>0){
   for(k=0;k<=n+1;k++){
    printf(" ");
   }
   printf("*");
   printf("\n");
  }
  for(k=0;k<=n;k++){
   for(j=0;j<=n-k;j++){
    printf(" ");
   }
   printf("*");
   for(j=0;j<=(2*k);j++){
    printf(" ");
   }
   printf("*");
   printf("\n");
  }
  if(n>1){
   printf("*");
   for(k=0;k<=n+1;k++){
    printf(" ");
    printf("*");
   }
   printf("\n");
  }
  return 0;
}
```

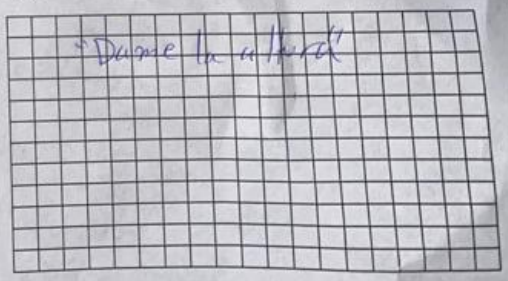
Diagrama de flujo y pseudocódigo

Alvaro Suzain



Compara y completa las instrucciones en el diagrama de flujo y el pseudocódigo, escribe en la cuadricula cual es la salida de los algoritmos para un valor de n=10 **(Valor 5 puntos)**

```
INICIO
ENTERO k,j,n;
  ESCRIBIR "Dame la altura"
  LEER n;
  ESCRIBIR "\n";

  SI n>0 ENTONCES
    PARA k<-1 HASTA k<=n-1 CON PASO 1 HACER
                          ESCRIBIR " ";
    FINPARA
                    ESCRIBIR "*"
                    ESCRIBIR "\n";
  FINSI

        PARA k<-2 HASTA k<=n-1 CON PASO 1 HACER
              PARAj<-1HASTA j<=n-k
              CON PASO 1 HACER
                          ESCRIBIR " ";
              FINPARA

              ESCRIBIR "*";

              PARA j<-1 HASTA j<=2*k-3 CON PASO 1 HACER
                          ESCRIBIR " ";
              FINPARA

              ESCRIBIR "*";
              ESCRIBIR "\n";

        FINPARA

        SI n > 1 ENTONCES
                    ESCRIBIR "*";
                    PARAk<-1HASTA k<=n-1
                    CON PASO 1 HACER
                                ESCRIBIR " ";
                                ESCRIBIR "*";
                     FINPARA
                     ESCRIBIR "\n";
              FINSI
FIN
```



Conclusión:

Las Funciones tienen diversos usos en muchos aspectos de ingeniería, en este caso de programación sirven para recopilar los datos dentro de un programa que requiere varias veces la misma estructura y así ahorrar tiempo al momento de ejecutar los bloques y declararlos, se puede realizar por medio de varios archivos .C dentro de uno solo declarándolos antes de usarlos.

Esta práctica será muy útil en futuros programas que tengamos que hacer.

https://github.com/3201050964/Pr-ctica-11