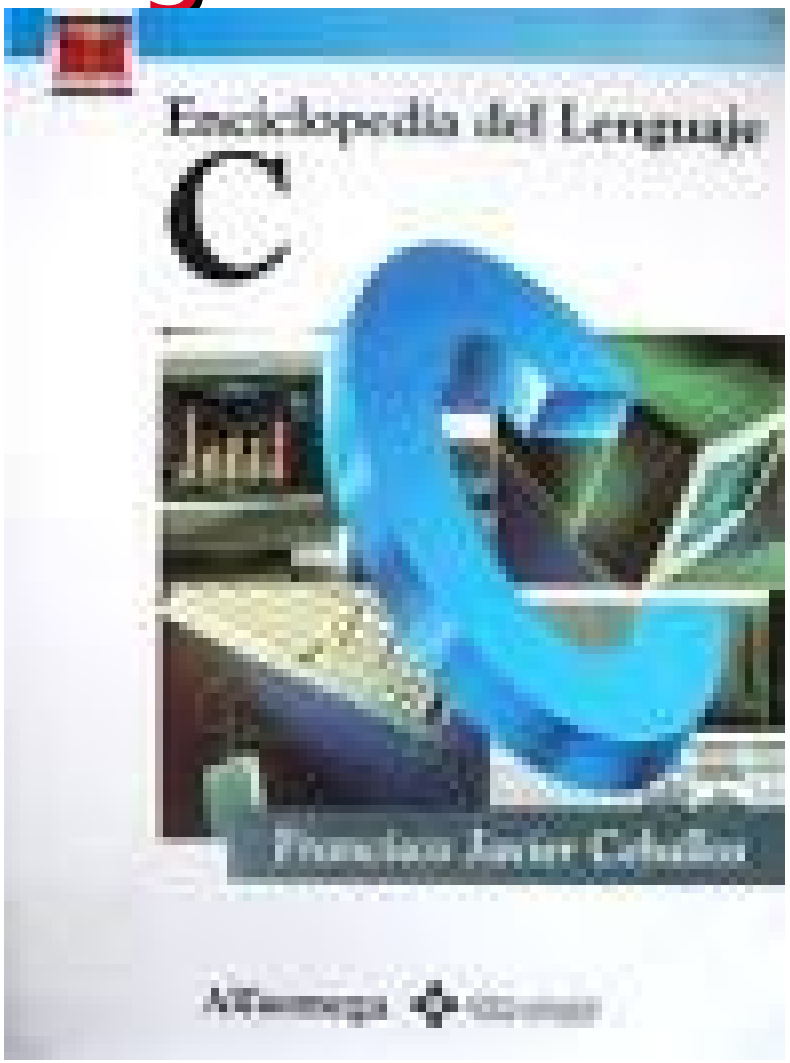
Lenguaje de programación “C”



Introducción

Elementos del lenguaje “C”

- Colección de funciones

- Estas funciones contienen **declaraciones, sentencias, expresiones** y otros elementos que en conjunto indican a la computadora que realizará

Identificadores

- Son nombres dados a constantes, variables, tipos, funciones de un programa.
- Ejemplos:
  - Cuenta
  - Calcula_primos
  - ab123
  - pl
  - i

Palabras reservadas

- Son identificadores predefinidos que tienen un significado especial para el compilador C.
- Ejemplos:
  - **auto**
  - **continue**
  - **else**
  - **for**

Estructura de un programa

- Un programa fuente es una colección:
  - Directrices (inclusión de archivos)
  - Declaraciones
  - Definiciones
  - Expresiones
  - Sentencias
  - Funciones.

Estructura de un programa

- /* Este es un comentario *
- muy largo ya que ocupa * mas de un renglón */
- Ejemplo de la estructura general de un programa en C:
- #include <stdio.h> /* Directrices del preprocesador */
- #include <stdlib.h>
- #define SU 100 /* Definición de constantes */
- int x,y; /* Variables globales */
- main() /* Programa principal */
- { /* Inicia el programa principal */
- float real; /* Variables locales */
- /* Acciones */
- printf (“\n Dame dos numero entero: ”);
- scanf(“%d%d”, &x,&y) ;
- real=x/y;
- printf(“\n Resultado: %f” ,real);

Tipos de datos estándar

- **Carácter:** Se declara con la palabra reservada **char** y ocupa un byte (se pueden representar 256 símbolos posibles).
- **Real:** Se declara con la palabra reservada **double**(8 bytes) o **float**(4 bytes),
- **Entero:** Se declara con la palabra reservada **int** y ocupa 2 bytes de memoria.

Tarea 3

- Investigar todos los tipos de datos en el lenguaje C
- Reglas para dar nombre a los identificadores en lenguaje c

Añadir la bibliografía consultada

Variables

- Es un identificador que tiene asociado un valor que puede cambiar a lo largo de la ejecución del programa.

- Piden al compilador que separe la cantidad de memoria necesaria.

Declaración

- **tipo identificador [, identificador , ..., identificador] ;**

donde,

- **tipo** :Tipo de la variable (char, int, ...).
- **identificador:** Nombre de la variable.

Ejemplo:

- int cuenta, suma_prom;
- float pí, radio;

Constantes

- Se refieren a valores fijos que no pueden ser alterados por un programa y pueden ser de cualquier tipo.

Declaración

- **#define identificador valor**

donde,

  - **identificador** es el nombre de la constante
  - **valor** es el valor asociado a la constante

- Ejemplo:
  - #define entero 2
  - #define cad “Soy constante “
  - #define car ‘a’

Expresiones

- Son combinaciones de constantes, variables, operadores y llamados a funciones.

Ejemplos:

- tan(1.8)+calcula(4)
- a+b*3.0*x-9.3242
- 3.77+sen(3.14*98.7)

Proposiciones de asignación

- **variable = expresión;**

Ejemplo:

- i=7;
- x=3.1+sin(10.8);
- Suma=prom(4);

Operadores aritméticos

OPERADOR	OPERACIÓN
+	Suma. Operandos enteros o reales
-	Resta. Operandos enteros o reales
*	Multiplicación. Operandos enteros o reales
/	División. Operandos enteros o reales. Si ambos operandos son enteros el resultado es entero. En el resto de los casos el resultado es real.
%	Módulo o resto de la división entera. Operandos tienen que ser enteros.
-	Menos unario. Operadores enteros o reales.

Ejemplos

```
int a=10, b=3, c;
float x=2.0, y;
```

- y = x +a; /* 12.0 */
- c = a / b; /* 3 */
- c = a % b; /* 1 */
- a = -b; /* –3 */

Operadores relacionales

OPERADOR	OPERACIÓN
<	Primer operando menor que el segundo
>	Primer operando mayor que el segundo
<=	Primer operando menor o igual que el segundo
>=	Primer operando mayor o igual que el segundo
==	Primer operando igual que el segundo
!=	Primer operando distinto que el segundo

Operadores Lógicos

OPERADOR	OPERACIÓN
&&	AND. 1. Si ambos operandos son distintos de 0. 0. Cualquier otro caso
||	OR. 0. Si ambos operandos son 0. 1. Cualquier otro caso.
!	NOT. 0 Si el operando tiene un valor distinto de cero. 1 en caso contrario. Resultado int. Operandos: entero, real o apuntador.

Ejemplos

- Expresión Valor

  !5 0

  !a Depende del valor de a

  !'z' 0

  !(x+7.7) Depende del valor de x

  !!5 1

Prioridad

OPERADOR	ASOCIATIVIDAD
()	Izquierda-derecha
-,++, --	Derecha-izquierda
*, /, %	Izquierda-derecha
+,-	Izquierda-derecha
<, <=,> >=	Izquierda-derecha
==, !=	Izquierda-derecha
&&, ||	Izquierda-derecha
=	Derecha-izquierda

Práctica 1. equipo de 4

Encuentra la expresión equivalente y el valor de la expresión

- int a, b, c, d;
- a=2; b=-3; c=7; d=-19;
- **Expresión Equivalente Valor**
- a / b
- b / b / a (b/b)/a 0

- c%a
- a%b
- d/b%a
- -a*d

Práctica 1

- a%-b*c
- 9/c+-20/d
- a%-b*c
- 9/c+-20/d
- -d%c-b/a*5+5
- 7-a%(3+b)
- - - - a
- A= b= c= -33

Práctica 1

- char c; int i,j,k; double x,y;
- c=’w’; i=j=k=3; x=0.0; y=2.3;

- i && j && k
- X && i || j-3
- X || i && j-3
- i<j && x<y
- i<j || x<y

Práctica 1

- i= =j && x<=y
- i= = 2 || j= = 4 ||k= =6

Otros operadores

- **Incremento (++)**

  Añade 1 a una variable

- **Decremento: (- -)**

  Resta 1 a una variable

Ejemplos

- A+=3 es equivalente a A=A+3
- k*=3+x es equivalente a k=k*(3+x)
- x = x+1; equivalente a ++x;
- x = x-1; equivalente a - -x;
- Estos operadores pueden ir antes o después de la variable.
- x = x+1; ++x; x = x-1; - -x;
- x++; x - -;

Funciones de Entrada salida

- **stdio.h (**standard input-output header: Es la biblioteca estándar del lenguaje de programación C.

- El archivo de cabecera que contiene las definiciones de macros, las constantes, las declaraciones de funciones y la definición de tipos usados por varias operaciones

Las siguientes funciones son algunas de las más utilizadas para entrada y salida de datos:

printf, scanf, getch, getchar, puts, gets,

Función Printf

Sintaxis:

printf(cadena de control, lista de argumentos)

Donde,

cadena de control

Es una cadena delimitada por comillas (“ “), formada por caracteres ordinarios, secuencias de escape y especificaciones de formato bajo el cual se requiere la salida de la información.

lista de argumentos

representa el valor o valores a escribir en la pantalla.

Ejemplos

- ***Secuencias de escape:\n,\t***

- ***Códigos de Formato:%d,%f, etc.***

Ejemplo

Tiempo=8

printf(“bienvenido a puebla son las %d \n", tiempo);

Función Scanf

Sintaxis:

scanf (cadena de control, lista de argumentos);

Donde,

cadena de control

Esta formada por códigos de formato de entrada, que están precedidas por un signo % y encerrados entre comillas “ “ dobles

Función Scanf

lista de argumentos

Representa el valor o valores a escribir en la pantalla.

Ejemplos:

```
int n;
printf(“Dame un numero : ”);
scanf(“%d ”, &n);
printf(“El numero capturado es %d”, n)
```

Ejemplo

```
main()
 {
  int a,r;
  float b;
  char c;
  printf("Introducir un valor entero, un real y
          un carácter \n => ");
   r=scanf(" %d %f %c \n",a,b,c);
   printf("Numero de datos leidos: %d \n",r);
   printf(" Datos leídos: %d %f %c \n",a,b,c);
  }
```

Práctica 2

En equipo de 4 integrantes diseña un programa que lea 10 datos de tipo entero e imprima cada uno de estos con la leyenda “primer, segundo, etc numero”.

Tarea 4

1. Investigar en algún libro las siguientes funciones(sintaxis, para que sirven, librería que las incluye)

- ***getch, getchar, putchar***
- ***continue***
- ***exit***
- ***Break***

Estructuras de Control

- En el lenguaje C se incluyen:
- if
- if – else
- switch

Decisión

- For
- While
- Do - while.

Repetición

Estructura if (sintaxis)

```
if (expresión)
   {
    propocisiones; /* bloque */
   }
proposición_siguiente;
```

Sintaxis

- **Expresión:** Numérica, relacional o lógica.
- **Proposiciones:** Si solo es una proposición entonces no es necesario poner { }

- Si son varias, estas se separan por **,**

Ejemplo

- if (grado>=90)
  printf(“\n FELICIDADES”):
- printf(“\nSu grado es %d”, grado);

¿Cuándo se imprime grado y cuándo se imprime FELICIDADES?

If – else (sintaxis)

```
if (expresión)
   {
    proposición_1 /* bloque 1*/
   }
else
    {
     proposición_2 /* bloque 2*/
    }
proposición_siguiente;
```

Ejemplo

```
Int x,y,min;
x=2; y=3;
if (x<=y)
    min=x
else
   min=y;
```

Anidamiento de if

```
if (expresion1)
    Sentencias1;
else if(expresión2)
        Sentencias2;
    else if(expresión3)
            sentencia3;
        ....
        else
            sentenciasN;
```

Ejemplo

```
if (a>b)
  printf("%d es mayor que %d",a,b);
else if (a<b)
        printf("%d es menor que %d",a,b);
        else
           printf("%d es igual a %d",a,b);
```

Proposición switch (sintaxis)

```
switch (expresión-test)
 {
   case constante1: sentencia;break;
   case constante2: sentencia;break;
   case constante3: sentencia;break;
    ...
   default : sentencia;
 }
```

switch

- **expresión-test:** Es una constante entera, una constante de caracteres o una expresión constante.

- **Sentencia:** Es una sentencia simple o compuesta (bloque)

Ejemplo

```
int calif;
switch (calif)
    {
     case 5: printf(“REPROBADO”); break;
     case 7: printf(“BIEN”); break;
     case 8: printf(“MUY BIEN”); break;
     case 10: printf(“EXELENTE”); break;
     default: printf(“NADA ”); break;
}
```

Practica 3

- En equipo de 4 integrantes diseña
1. Un programa que lea los números enteros a, b, c y d, e imprima luego dichos números y; a) Si son positivos,
2. Un programa en lenguaje C que: Acepte como datos los coeficientes a, b y c, de una ecuación de segundo grado, e imprima sus raíces.

Tarea 5

Hacer en lenguaje c

1. Un programa que lea los números enteros a, b, c y d e imprima luego dichos números y

a) Si son positivos, un mensaje que indique, para cada uno de ellos, si es o no múltiplo de 5;

2. Un programa lea los números enteros a, b, c y d e imprima luego dichos números y;

a) Si son positivos, un mensaje que indique, si están o no ordenados en secuencia ascendente,

b) Un mensaje de error en caso contrario.

Proposición for (sintaxis)

- **for (inicialización ; condición ; incremento)**

```
{
 sentencia1;
 sentencia2;
 ...
 sentencian;
}
```

for

- **Inicialización:** Proposición(es) de asignación que establece(n) la(s) variable(s) de control.
- **Condición:** Es una expresión relacional o lógica que determina cuando terminara el ciclo.
- **Incremento:** Define como cambiara la variable(s) de control.

Ejemplos

1. Imprimir los múltiplos de 7 entre 7 y 112

```
for (k = 7 ; k <= 112 ; k = k+7)
  printf("%d",k)
```

2. Codificación de la suma de 10 al 1.

```
  suma=0;
  for(i=10; i>=1; i--)
     suma+=i;
```

Ejemplo

- 3. Ejemplo: Se imprimen los números del 9 al 1.

```
for (a=9 ; a >= 1 ; a--)
Printf("%d",a);
```

Proposición While

- Ejecuta una sentencia(s), 0 o + veces

```
while (expresión)
    {
      sentencia1;
      sentencia2;
      sentencia3;
      ...
     sentencian;
    }
```

While

- **Expresión:** Es cualquier expresión numérica, relacional o lógica.
- **sentencia_i:** Es una sentencia(s), si es una sentencia (no es necesario {}).

Ejemplo: Imprime los primeros 10 #s

```
Int num;
num=1;
while (num<=10)
{
  printf("el número es\n%d",num);
  num=num+1;
}
```

Do - while

- Ejecuta una sentencia(s) 1 o + veces.

```
do
  {
   sentencia1;
   sentencia2;
   sentencia3;
   ...
   sentencian;
  }while (expresión);
```

Do - while

- **Expresión:** Es cualquier expresión numérica, relacional o lógica.
- **Sentencia:** Es una sentencia(s), si es simple (sin usar {})

Ejemplo

```
int n;
do
  {
   printf("\n Da un número : ");
   scanf("%d",&n);
  }while ( n>100);
```

Práctica 4. equipo de 4

- Hacer programas en c para

1. Realizar la suma de dos números con incrementos de 1.
2. Realizar la resta de dos números con decrementos de 1
3. Leer N números y obtener el promedio solo de los números pares de la lista.
4. Introducir un conjunto de M números, determinar la cantidad de números positivos y negativos del conjunto.
5. Calcular la sumatoria $\sum_{1}^{n} x$

Tarea 6

1. Investigar las funciones en lenguaje c para calcular un numero aleatorio.
2. Hacer un programa que permita determinar si un número entero dado n es primo.
3. Un programa que acepte como datos los coeficientes a, b y c, de una ecuación de segundo grado, e imprima sus raíces.