Programación Estructurada

Actividad de Aprendizaje 01- ADA 01

Lenguaje de programación: C

1. Palabras Reservadas

Auto: Modificador que indica que una variable local se crea al inicio de la ejecución de la función y se destruye al final.

Break: Finaliza la ejecución de la instrucción do, for, switch o while más próxima que la incluya.

Switch_Case: Esta sentencia es una generalización de las sentencias if_else. En el caso de las sentencias if, la expresión que se evalúa como condición es booleana, lo que quiere decir que sólo hay dos valores posibles, y, por lo tanto, sólo se puede elegir entre dos sentencias a ejecutar.

Char: Tipo de dato carácter alfanumérico.

Const: Define variables cuyo valor debe permanecer constante durante toda la ejecución del programa.

Continue: Provoca que se comience una nueva iteración, evaluándose la expresión de control.

Default: Es el caso por defecto que se ejecuta si dentro del switch no concuerda ninguno de los casos definidos.

Do…while: Variación del while donde primero se ejecuta y después se procede a evaluar la expresión de control.

Double: Las variables de este tipo almacenan números en formato de coma flotante, mantisa y exponente, al igual que float, pero usan una precisión mayor, a costa de usar más memoria.

If…Else: Permite la ejecución condicional de una sentencia. Si la condición es verdadera se ejecutará la sentencia1, si es falsa, (y si existe la parte del else), se ejecutará la sentencia2.

Enum: Permite declarar valores de datos que se ajustan a series ordenadas en las cuales un elemento sigue, o precede, a otro.

Extern: Define que existe una variable global que está definida en otro archivo fuente extern int c.

Float: Las variables de este tipo almacenan números en formato de coma flotante, esto es, contienen un valor de mantisa y otro de exponente**.**

For: Sentencia de control iterativa, que permite inicializar los controles de un ciclo mediante la estructura: for (Inicialización; Condición; Actualización).

Goto: Instrucción de control de salto que permite realizar saltos en el flujo de control de un programa.

Int (short, long): Tipo de dato entero con signo (normalmente 2 o 4Bytes). Las variables enteras almacenan números enteros dentro de los límites de cada uno de sus tamaños. A su vez, esos tamaños dependen de la plataforma, del compilador, y del número de bits que use por palabra de memoria: 8, 16, 32... short int, int y long int.

Register: Aplicable únicamente a variables locales e indica al compilador que esta variable debe ser almacenada permanentemente en un registro del procesador del ordenador.

Return: Sentencia de salida de una función, cuando se ejecuta, se devuelve el control a la rutina que llamó a la función.

Signed: Define que el valor de una variable numérica puede ser positivo o negativo.

Unsigned: Se utiliza cuando la variable sea siempre positiva.

Sizeof: Este operador nos permite obtener el tamaño de un tipo o de una variable.

Static: Modificador que indica que una variable local no se destruye al finalizar la función donde fue declarada.

Struct: Un registro que agrupa distintos tipos de datos en una misma estructura.

Typedef: Se utiliza para renombrar tipos de datos con el fin de que la escritura y lectura del programa se nos haga más sencilla.

Union: La unión es un tipo especial de estructura que permite almacenar elementos de diferentes tipos en las mismas posiciones de memoria.

Void: Es un tipo de dato que puede representar: Nada (para funciones) o cualquier tipo de dato (para punteros).

Volatile: Es un modificador que indica al compilador que el valor de una variable se puede cambiar por medios externos (no especificados) al programa.

While: Sentencia de control iterativa, que evalúa una condición para su control. La sentencia es ejecutada repetidamente mientras la condición sea verdadera.

2. Tipos de variables:

Int: Cantidad entera.

char: Carácter.

float: Almacena valores reales en un punto flotante.

double: Almacena valores reales en doble precisión.

void: Se utiliza para definir una función que no devuelve ningún valor o declarar punteros genéricos.

3. Operadores

Aritméticos:

c	Suma, adición
-	Resta, sustracción
*****	Multiplicación, producto
/	Cociente división entera
%	Resto división entera
/	División

Relacionales:

>	Mayor
>=	Mayor o igual
<	Menor
<=	Menor o igual
==	Igual
!=	Diferente

Lógicos:

&&	And, y, conjunción
||	Or, o, disyunción
!	Not, no, negación

4. Instrucciones de control de flujo:

If: Esta sentencia es equivalente a la que poseen la mayoría de los lenguajes de programación y sirve para bifurcar en un punto de programa. Permite tomar decisiones al programa.

if (expresión)

sentencia1;

o también

if (expresión)

sentencia1;

else

sentencia2;

While: Un bucle es un conjunto de sentencias que se ejecutan repetidamente hasta que se alcanza una condición de fin de bucle, o condición de salida. El bucle while es el tipo de bucle más sencillo.

while (expresión)

sentencia;

do-while: Su funcionamiento es análogo el del bucle while, salvo que la expresión de control se evalúa al final del bucle. Esto nos garantiza que el bucle do-while se ejecuta al menos una vez.

do

sentencia;

while (expresión);

for: Se utiliza para realizar una acción un número determinado de veces. Está compuesto de tres expresiones: la de inicio, la de control y la de incremento, y de una sentencia.

```
for (i =0; i < 10; i++)
        printf("i vale %d\n", i);
```

break: Rompe la ejecución de un bucle o bloque de instrucciones y continúa en la instrucción que sigue al bucle o bloque.

```
int a = 10;
    while (1) {
        if (a-- <= 1) break;
        printf("%d\n", a);
    }
```

continue: Rompe la ejecución habitual del bucle y procede a evaluar de nuevo la expresión del bucle.

```
int a = 0;
    while (a < 10) {
        a++;
        if (a == 7) continue;
```

```
        printf("%d\n", a);
    }
```

Switch: Sirve para agrupar varias sentencias if en una sola, en el caso particular en el que una variable es comparada a diferentes valores, todos ellos constantes, y que realiza acciones si coincide con ellos.

```
switch (control) {
    case expresión1_const:
        sentencia1;
        break;
    case expresión2_const:
        sentencia2;
        break;
    default:
        sentencia0;
        break;
}
```

5. Bibliotecas/Funciones más relevantes:

pow(x,y): Permite calcular el resultado de un número elevado a otro.

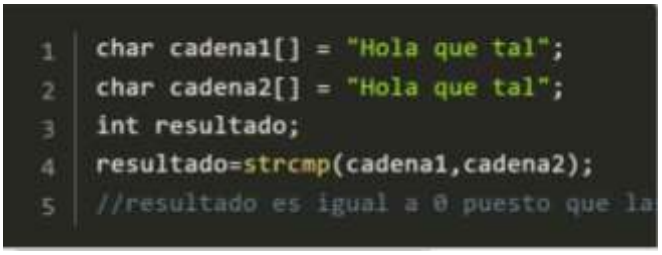
```
int numero =5;
int exponente =3;
int resultado;
resultado=pow(5,3);
//Resultado ahora vale 5^3 =125
```

sqrt: Permite calcular la raíz cuadrada de un número.

```
int numero =16;
int resultado;
resultado=sqrt(16);
//resultado vale 4
```

Gestión de cadenas de caracteres en C: Las cadenas de caracteres, por su forma de constituirse en C, siempre dan más problemas que los tipos numéricos. Por ello es muy recomendable usar la librería string.h para manejarlas correctamente.

strcmp(cadena1,cadena2): Permite comparar si dos cadenas de caracteres son iguales.



```
1  char cadena1[] = "Hola que tal";
2  char cadena2[] = "Hola que tal";
3  int resultado;
4  resultado=strcmp(cadena1,cadena2);
5  //resultado es igual a 0 puesto que la
```

strcpy(cad1,cad2): Copia el contenido de cad2 en cad1. Es muy importante al usar esta función que la cadena de destino tenga un tamaño igual o mayor que la de origen para evitar errores.

Crear números aleatorios “rand()”:

```
1  srand(time(NULL));
2  var=rand();
```

6. Funciones básicas de entrada y salida:

getchar(): Se puede conseguir la entrada de un carácter a través del dispositivo de entrada estándar, usualmente el teclado.

putchar(): Se puede conseguir la salida de un carácter a través del dispositivo de salida estándar, usualmente el monitor.

scanf(): Se puede introducir datos en la computadora a través del dispositivo de entrada estándar. Esta función permite introducir cualquier combinación de valores numéricos, caracteres sueltos y cadenas de caracteres. La función devuelve el número de datos que se han conseguido introducir correctamente.

printf(): Se puede escribir datos en el dispositivo de salida estándar.

gets(): Permite leer una cadena de caracteres desde el dispositivo de entrada estándar.

puts(): Permite visualizar una cadena de caracteres en el dispositivo de salida estándar.

7. Ejemplos

Programa que determina si una letra es vocal o no

switch(letra)

```
{
case 'a':
case 'e':
case 'i':
case 'o':
case 'u':
    EsVocal = true;
    break;
 default:
EsVocal = false;
```

Programa que determina si la contraseña es correcto o incorrecta

```
string password = "";
cout << "Ingrese la contraseña: ";
cin >> password;
if(password == "myClave")
{
          cout << "Contrasenia correcta. Bienvenido";
}
else
{
         cout << "Contrasenia incorrecta.";
}
```

Bucle que se repetirá hasta que el contador llegue a 100

```
for(int i = 0; i < 100; i = i + 1);
```

Programa de números pares

```
int Paridad(int x)
{
        if(x % 2) return 1;
```

```
    return 0;
}
```