OBJETIVOS DE LA TABLA DE SÍMBOLOS

La tabla se símbolos permite guardar información de las variables (globales, locales y parámetros de funciones) y de las funciones.

La tabla de símbolos es un **diccionario** que permite almacenar claves e información asociada a cada clave. Las claves son los identificadores de las variables y las funciones y la información ya lo veremos.

La tabla de símbolos debe implementar **las reglas de visibilidad** del lenguaje concreto y **la regla de colisiones** (no repetición de identificadores en el mismo ámbito).

REGLAS DE VISIBILIDAD EN ALFA

```
main
{
// Declaraciones de variables globales
   int g1, g2;

// Funciones
        function int fun1 (int p1)
        {
           int l1;    // declaración variable local
           ……
        }

        function  int fun2  (int p2)
        {
           int l2;    // declaración variable local
           ……
        }

// Sentencias

        …….
}
```

ÁMBITO (ESTRUCTURA DE UN PROGRAMA ALFA)	QUÉ SE DECLARA EN CADA ÁMBITO	QUÉ SE VE EN CADA ÁMBITO
Global	g1, g2, fun1, fun2	g1, g2, fun1, fun2
Función fun1	l1, p1	g1, g2, fun1, l1, p1
Función fun2	l2, p2	g1, g2, fun1, fun2, l2, p2

REGLAS DE COLISIONES EN ALFA

En un mismo ámbito no se pueden repetir identificadores.

FUNCIONES DE LA TABLA DE SÍMBOLOS

- Como hemos visto en las reglas de visibilidad, en ALFA hay como mucho dos ámbitos. La tabla de símbolos del compilador se puede implementar con dos tablas, una global y una local. La global se crea al principio y se destruye al final. La local se crea y se destruye con cada función.

- Cada tabla individual tiene que tener funciones: init, destroy, set(id, info), get (id) -> info.

- **La tabla de símbolos debe tener las siguientes funciones**
  - DeclararGlobal(id, desc_id) devuelve éxito/error
  - UsoGlobal(id) devuelve éxito/error + desc_id
  - DeclararLocal(id, desc_id) devuelve éxito/error
  - UsoLocal(id) devuelve éxito/error + desc_id
  - DeclararFuncion(id, desc_id) devuelve éxito/error

```
main
{
// Declaraciones
   int x, resultado;              // declaración variable global

// Funciones
        function int suma (int x; int y)   // declaración función y declaración parámetro (local)
        {
           int aux;    // declaración variable local


                aux = x+y;   // uso local (parámetro y variable)
                return aux;
        }

// Sentencias

        scanf x;                          // uso global
        scanf y;
        resultado = suma(x,y);
        printf resultado;
}
```

Pseudocódigo de funciones de tabla de símbolos y reparto de programas por recorrido

DeclararGlobal(id, desc_id)

```
if ( TablaSimbolosGlobal -> get(id) == null)
{
        TablaSimbolosGlobal->set(id, desc_id);
        return ok;   // 9
}
else  return error; // 1, 2, 3
```

DeclararLocal(id, desc_id)

```
if ( TablaSimbolosLocal -> get(id) == null)
{
        TablaSimbolosLocal->set(id, desc_id);
        return ok;   // 10 11
}
else  return error; // 4, 5, 6, 7, 8
```

UsoGlobal(id)

```
dato = TablaSimbolosGlobal -> get(id);
if ( dato == null)
{
        return err;   // 12, 17 (funciones)
}
else  return dato; // 9, 16 (funciones)
```

UsoLocal(id)

```
dato = TablaSimbolosLocal -> get(id);
if ( dato == null)
{
        dato = TablaSimbolosGlobal -> get(id);
        if (dato == null) return err;   // 13
        else return dato                // 14
}
else  return dato; // 10, 11, 15
```

DeclararFuncion(id, desc_id)

```
if ( TablaSimbolosGlobal-> get(id) != null )
        return error;  // 3
else {
        TablaSimbolosGlobal -> set(id, desc_id);
        TablaSimbolosLocal->init();
        TablaSimbolosLocal->set(id, desc_id);
        return ok;     // 11  }
```

Ejemplo 1

```
main
{

      int x, x;


      printf x;

}
```

Ejemplo 2

```
main
{

      boolean i;
      int j;


      function int i (int arg)
      {
            return arg*2;
      }

      printf i(j);

}
```

Ejemplo 3

```
main
{

      boolean i;
      int j;


      function int doble (int arg)
      {
            return arg*2;
      }

      function boolean doble (int arg; int dosPorArg)
      {
            return (dosPorArg == 2*arg);
      }

      j=20;

      printf doble(j);

}
```

Ejemplo 4

```
main
{

      int i, j;


      function int suma (int arg1; int arg2)
      {
            boolean aux;
            int aux;

            return arg1 + arg2;
      }

      j=20;
      i=10;

      printf suma(j,i);

}
```

Ejemplo 5

```
main
{

      int i, j;


      function boolean mayorque (int num; int num)
      {
            return (num > num2);
      }

      j=20;
      i=10;

      printf mayorque(j,i);

}
```

Ejemplo 6

```
main
{
      int num, num2;

      function int potencia (int x; int exp)
      {
            int x;
            int i;



         x=1;
      i = 0;

      while ((i<exp))
      {
         x = x * x;
                  i = i + 1;
      }

            return x;
      }


      num=10;
      scanf num2;

      printf potencia(num,num2);

}
```

Ejemplo 7

```
main
{
// Declaraciones

    int x, y, resultado;

// Funciones
      function int suma (int num1; int num2)
      {
            int suma;

            suma = num1 + num2;

            return suma;
      }

// Sentencias

      scanf x;
      scanf y;

      resultado = suma (x,y);

      printf resultado;
}
```

Ejemplo 8

```
main
{
// Declaraciones

    int x, y, resultado;

// Funciones
      function int suma (int suma; int suma2)
      {
            return suma + suma2;
      }

// Sentencias

      scanf x;
      scanf y;

      resultado = suma (x,y);

      printf resultado;
}
```

Ejemplo 9

```
main
{

      int num1, num2;

      scanf num1;
      scanf num2;

      printf num1 + num2;

}
```

Ejemplo 10

```
main {

// Declaraciones
      int x;

// Funciones

      function int doble (int num)
      {
            int dos;


            dos = 2;

            return dos * num;
      }

      scanf x;

      printf doble(x);

}
```

Ejemplo 11

```
main
{
// Declaraciones

    int x, resultado;

// Funciones
      function int factorial (int n)
      {
            if ( (n == 0) )
            {
                  return 1;
            }
            else
            {
                  return n * factorial ( n - 1);
            }
      }

// Sentencias

      scanf x;

      resultado = factorial (x);

      printf resultado;
}
```

Ejemplo 12

```
main
{
      int num, num2;

      scanf num;
      scanf num2;

      printf num1 + num2;
}
```

Ejemplo 13

```
main {

// Declaraciones
      int x;

// Funciones
      function int negar ( int numero )
      {
            int w;

            w = -arg;
            return  w;
      }

      scanf x;
      printf negar(x);
}
```

Ejemplo 14

```
main {

// Declaraciones
      int x;


// Funciones
      function int negar ( int numero )
      {
            int w;

            w = -x;

            return  w;

      }

      scanf x;

      printf negar(x);

}
```

Ejemplo 15

```
main
{

      int num, num2;


      function int potencia (int num; int exp)
      {
            int num2;
            int i;



         num2=1;
         i = 0;

         while ((i<exp))
         {
                num2 = num2 * num;
                i = i + 1;
         }

            return num2;
      }


      num=10;
      scanf num2;

      printf potencia(num,num2);

}
```

Ejemplo 16

```
main
{

      boolean i;
      int j,k;


      function int doble (int arg)
      {
            return arg*2;
      }

      function boolean esdoble (int arg; int dosPorArg)
      {
            return (dosPorArg == doble(arg));
      }

      scanf j;
      scanf k;

      printf esdoble(j,k);

}
```

Ejemplo 17

```
main
{

      boolean i;
      int j,k;



      function boolean esdoble (int arg; int dosPorArg)
      {
            return (dosPorArg == doble(arg));
      }

      function int doble (int arg)
      {
            return arg*2;
      }

      scanf j;
      scanf k;

      printf esdoble(j,k);

}
```

TRABAJO DE LA PRÁCTICA

Material proporcionado

- Enunciado
- Tabla hash programada en C

Trabajo a desarrollar

- Tabla de símbolos que implemente la gestión de ámbitos
- Programa de prueba de la tabla de símbolos

En la siguiente tabla se muestran los ficheros de entrada y salida del programa de prueba correspondientes a algunos de los ejemplos anteriores.

Ejemplo	Fichero de entrada	Fichero de salida
1	x 1 x 1 x	x -1 x x 1
4	i 1 j 2 suma -1 arg1 3 arg2 4 aux 5 aux 5 arg1 arg2 cierre -999 j i suma j i	i j suma arg1 arg2 aux -1 aux arg1 3 arg2 4 cierre j 2 i 1 suma -10 j 2 i 1