TIPOS ARRAYS

TIPO ARRAY

Un ***Array*** es una estructura de datos que almacena bajo el mismo nombre (variable) a una colección de datos del mismo tipo.

Declaración de un array

```
<tipo_de_dato> <nombre_de_variable> [numero_de_elementos];
```

```
char nombre [25];  //variable nombre de 25 elementos tipo char

int  edades [25];  /variable edades de 25 elementos de números enteros
```

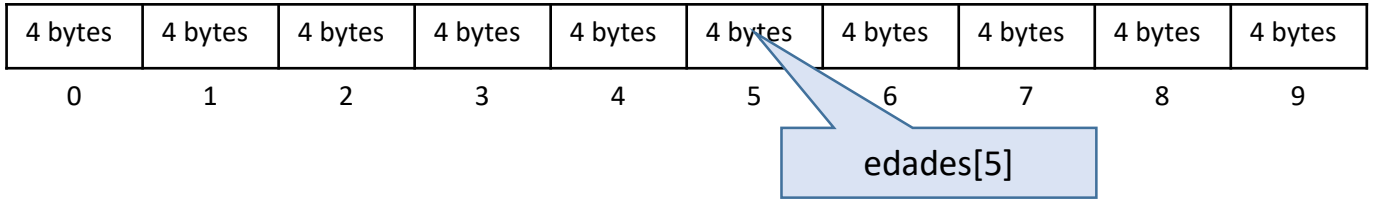
Cuando declaramos un array, el compilador reserva el espacio necesarios en memoria en posiciones contiguas. Para conocer los bytes ocupados se puede usar la función sizeof(var).

```
cout << sizeof(edades);  //devolverá el valor 100
```

Acceso a los elementos de un array

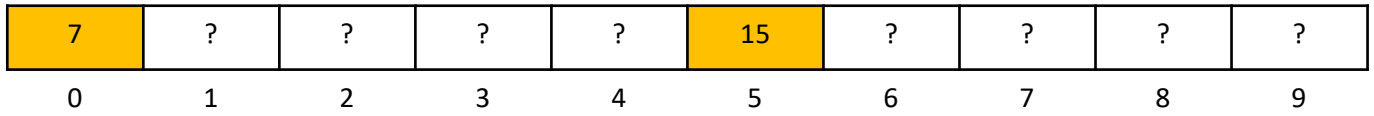
Para acceder a los elementos de un array hay que poner el nombre y el **índice** del elemento al que queremos acceder ó referenciar

int edades[10];




```
//arrays01.cpp
#include <iostream>
using namespace std;
int main ()
{
   int edades[10];
   int i;
   cout <<"Tamaño en bytes:"<< sizeof(edades)<<endl; //Tamaño en bytes reservados
   edades[0]=7; //el primer elemento vale 7
   edades[5]=15; //el sexto elemento 15.
   for ( i=0; i<10; i++)
   {
                cout<<edades[i]<<endl; //el resto de elementos toman el valor que hubiese en memoria
   }

   return 0;
}
```



ACCESO A TODOS LOS ELEMENTOS DE UN ARRAY

```
for ( int i = 0; i < num_elementos ; i++ )
{
        procesar_elemento [i];
}
```

```
//arrays02.cpp
#include <iostream>
#include <conio.h> //función getch() y getche()
using namespace std;
int main ()
{
   char nombre[5];
   for (int i=0;i<5;i++)
   {
        cout<<endl<<"Dame el nombre "<<i;
            nombre[i]=getche();
   }
   for (int i=0;i<5;i++)
        cout<<endl<<nombre[i];

   return 0;
}
```

Captura un carácter y lo muestra en pantalla, a diferencia de getch(). Biblioteca conio.h

INICIALIZACIÓN DE UN ARRAY

Antes de usar una variable de tipo array hay que asignar valores a cada uno de sus elementos.

1. Inicialización en la declaración.
2. Inicialización elemento a elemento.
3. Inicialización mediante un bucle

```
//arrays03.cpp
#include <iostream>
#include <stdlib.h> //función getch() y getche()

using namespace std;
int main ()
{
   char frase[4];
   int notas[5]={7,6,9,8,7}; //Inicialización en la declaración.
   int temperaturas[3];

   frase[0]='p';frase[1]='e';frase[2]='p';frase[3]='e';//Inicialización elemento a elemento

   for (int i=0; i<3; i++)   //Incialización bucle for
                cin>>temperaturas[i];
   system("pause");
   return 0;
}
```

ARRAYS MULTIDIMENSIONALES

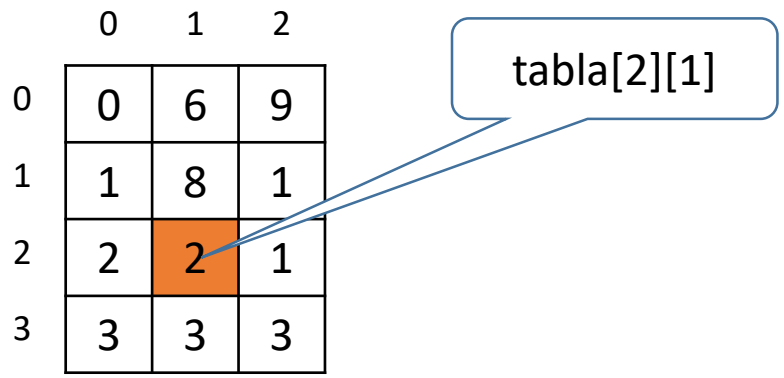
Arrays unidimensionales se conocen como listas o **vectores**.

Los **arrays multidimensionales** son aquellos que tienen más de una dimensión y por tanto tienen más de un índice. Los arrays de dos dimensiones de suelen denominar tablas o matices.

Declaración de de array bidimensional

```
<tipo_de_dato> <nombre_de_variable> [num_filas] [num_columnas];
```

int notas[4][3];




```
//arrays04.cpp
#include <iostream>
using namespace std;
int main ()
{
   int notas[4][3]={0,6,9,1,8,1,2,2,1,3,3,3};
   for (int i=0; i<4; i++)
   {
            for (int j=0; j<3; j++)
                     cout<<notas[i][j]<<"  ";
            cout << endl;
    }
   return 0;
}
```

ARRAY TRIDIMENSIONAL

int cubo[n][m][k] Variable array de **n x m x k** elementos enteros