Situación problemática

Se necesita tomar la lectura de temperatura y humedad de una estación meteorológica. La misma, toma una muestra cada una hora.
Al final del día, se debe imprimir una tabla con los valores tomados e informar los valores máximos y mínimos de cada una de las variables.

Solución propuesta

```
1  int menu (void)
2  {
3     float temp1=0;   float temp2=0;
4     float temp3=0;   float temp4=0;
5     float temp5=0;   float temp6=0;
6     float temp7=0;   float temp8=0;
7     float temp9=0;   float temp10=0;
8     float temp11=0;  float temp12=0;
9     float temp13=0;  float temp14=0;
10    float temp15=0;  float temp16=0;
11    float temp16=0;  float temp18=0;
12    float temp17=0;  float temp20=0;
13    float temp19=0;  float temp22=0;
14    float temp23=0;  float temp24=0;
15    float hum1=0;    float hum2=0;
16    float hum3=0;    float hum4=0;
17    //.....//
18 }
```

Arreglos unidimensionales

Introducción

Un arreglo es una colección de variables del **mismo tipo** que están referenciadas por un nombre en común.

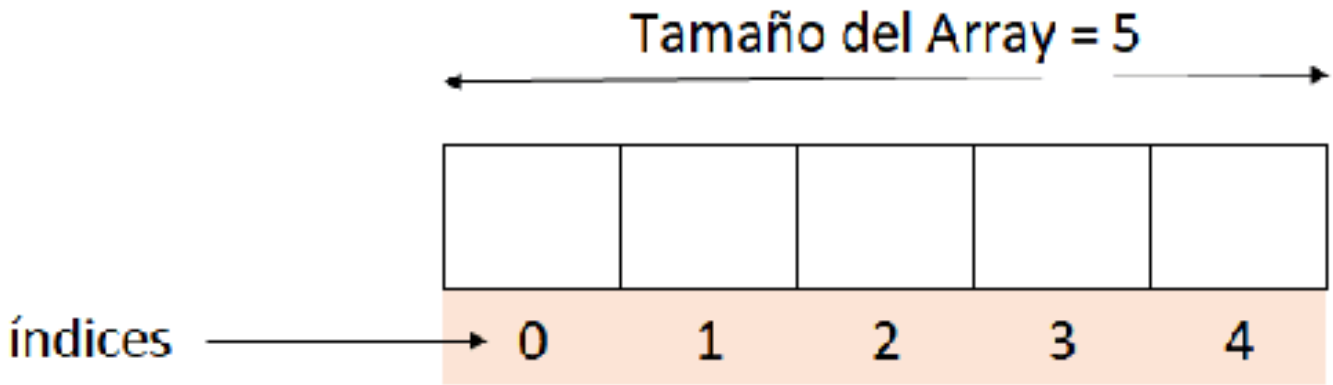

Figure: Representación de un arreglo unidimensional de cinco elementos.

Arreglos unidimensionales

¿Cómo lo vemos desde C?

Se puede pensar a un arreglo como un conjunto de posiciones de memoria contiguas, agrupadas bajo el mismo nombre. La dirección mas baja corresponde al primer elemento y las mas alta al último.

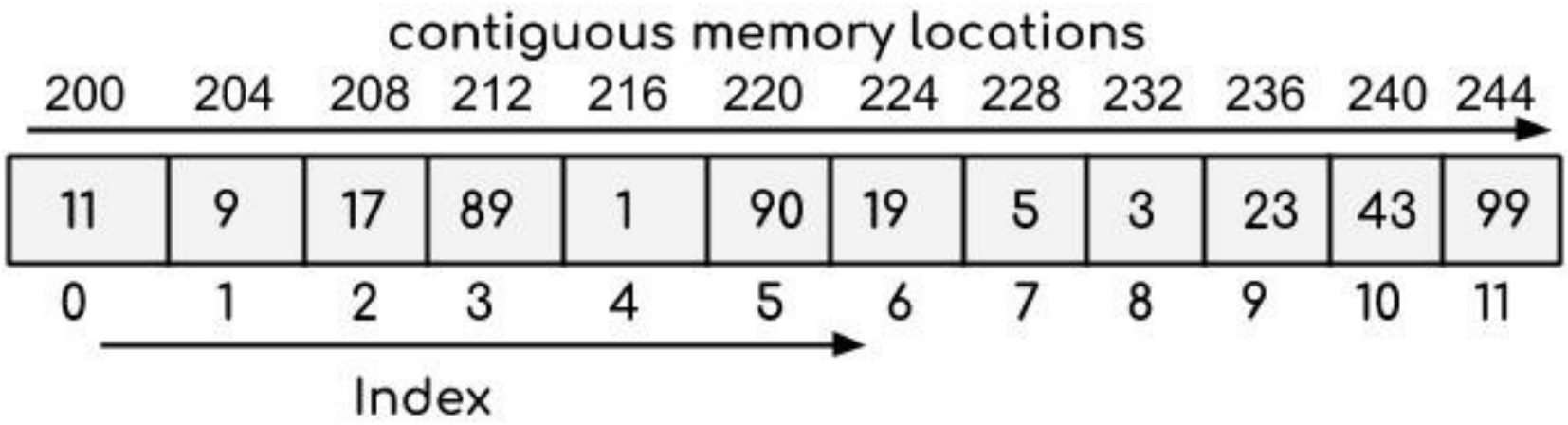

Figure: Representación de un arreglo unidimensional de cinco elementos.

Notar que el primer elemento de un arreglo es el 0.

Arreglos unidimensionales

¿Cuánto ocupa cada dato en memoria?

Los arreglos en C tienen tamaño estático, es decir que al ejecutarse el programa mantendrá una cantidad fija de memoria reservada.

Tipo de datos	Tamaño en bytes
char - unsigned char	1
short int, unsigned short int	2
int, unsigned int, long int, unsigned long int	4
float	4
double	8
long double	12

El tamaño en bytes puede cambiar de una plataforma a otra.

▸ Más información sobre bits y bytes

Declaración de un arreglo unidimensional

La forma general de declarar a un arreglos es:

```
1  tipo nombre [cantidad_de_elementos];
2
3  float temp       [24];
4  float hum        [24];
5  int   id_muestra [24];
```

Los nombres de los arreglos pueden contener letras, números y guiones bajos.

Para declarar e inicializar un arreglo:

```
1  float temp[5]={2.3, -5.5, 8.9123, -12.35, 9.204}
2  float hum []  ={10.00, 20.40, 15.98, 100.00, 87.55}
```

¿Cómo accedemos a los elementos de un arreglo?

Se puede hacer referencia a cualquiera de los elementos de un arreglo utilizando el nombre del mismo y el operador [].
En su interior, debe ir un número entero o una expresión entera.
Por ejemplo:

```
1  temp[0]=10.30;
2  temp[1]=50.29;
3  temp[2]=10.22;
4  temp[3]=-10.21;
5  temp[4]=35.00;
6
7  int indice=4;
8  hum[4]=10.89;
9  hum[2+2]=10.89;
10 hum[indice]=10.89;
```

Las lineas 8,9 y 10 son equivalentes.
¿Si tenemos que cargar, recorrer y procesar un arreglo de 1000 elementos?

Vamos a C I

Carga de datos a un arreglo:

```
1  int main(void)
2  {
3      int ii;
4      int array[10];
5      int dato=0;
6      for(ii=0;ii<10;i++)
7      {
8         printf("Ingrese el elemento %d",ii);
9         scanf("%d",&dato):
10        array[ii]=dato;
11     }
12 }
```

Vamos a C II

Impresión del arreglo anterior:

```
1  for(ii=0;ii<10;ii++)
2  {
3     printf("El elemento %d vale %d\n",ii,array[ii]);
4  }
```

▸ Ver ejemplo en github

Se debe tener en cuenta que C no tiene mecanismos de control de acceso a memoria, es decir que si un arreglo tiene 10 elementos y se intenta escribir la posición 19 podrían sobre-escribirse valores de otras variables.
Es responsabilidad del desarrollador hacer este control.

La directiva Define

Una directiva (o macro) es resuelta por el preprocesador antes del proceso de compilación y reemplaza cada una de las etiquetas definidas, por su valor definido en la etiqueta.

Ejemplo antes de la compilación:

```
1 #define tam 10
2
3 int array [tam];
```

Ejemplo luego de la compilación:

```
1 int array [10];
```

Como se ve, utilizando esta directiva se pueden desarrollar algoritmos genéricos independientes de la cantidad de elementos del arreglo.

Ejemplos de arreglos unidimensionales

1. Diseñar y codificar un programa que cargue un arreglo de 1000 elementos con números aleatorios comprendidos entre 10 y 20. Luego el programa debe recorrer el arreglo y contar las veces que un número ingresado por teclado aparece en el arreglo.
   ▸ Ver en github
2. Diseñar y codificar un programa que solicite el ingreso por teclado de 10 números enteros y los almacene en un arreglo unidimensional. Luego, se deben convertir los números positivos en negativos y los negativos en positivos. Finalmente, se debe imprimir el arreglo original y el arreglo convertido. ▸ Ver en github
3. Diseñar y codificar un programa que permita el ingreso de las mediciones de temperatura y humedad de una estación meteorológica. La misma, toma una muestra cada 1h. Al final del día se debe imprimir la temperaturas y humedades máximas y mínimas. ▸ Ver en github

Arreglos Bidimensionales

Los arreglos bidimensionales son útiles cuando se debe trabajar con información ordenadas en filas y columnas.
Por convención, el primer índice indica la fila del elemento y el segundo la columna.

	Column 0	Column 1	Column 2
Row 0	x[0][0]	x[0][1]	x[0][2]
Row 1	x[1][0]	x[1][1]	x[1][2]
Row 2	x[2][0]	x[2][1]	x[2][2]

Figure: Representación de un arreglo bidimensional de 3x3.

Declaración de un arreglo bidimensional

```
1    tipo nombre [cantidad_de_filas][cantidad_de_columnas];
2
3    float temp        [30][24];
4    float hum         [30][24];
```

Para declarar e inicializar un arreglo bidimensional,

```
1   int matrix[2][3]={{1,2,3}  ,{4,5,6}  };
```

Carga de un arreglo bidimensional

```
1  #DEFINE FIL 1
2  #DEFINE COL 3
3  #include<stdio.h>
4  int main(void)
5  {
6       int ii;
7       int jj;
8       int array[FIL][COL];
9       int dato=0;
10      for(ii=0;ii<FIL;ii++)
11      {
12         for(jj=0;jj<COL;jj++)
13         {
14            printf("Ingrese el elemento %d %d \n",ii,jj);
15            scanf("%d",&dato);
16            array[ii][jj]=dato;
17         }
18      }
19
```

Impresión de un arreglo bidimensional

```
1  for(ii=0;ii<FIL;ii++)
2  {
3      for(jj=0;jj<COL;jj++)
4      {
5          printf("%d\t",ii,jj,array[ii][jj]);
6      }
7      printf("\n");
8  }
```

▸ Ver ejemplo en github

Ejemplos de arreglos bidimensionales

1. Diseñar y codificar un programa que cargue un arreglo bidimensional de enteros de [2][5] y lo imprima ordenadamente como una matriz del álgebra lineal. ▸ Ver en github
2. Diseñar y codificar un programa que cargue un arreglo bidimensional de [10][8] e imprima la suma de todos los elementos de cada una de sus filas.
   ▸ Ver en github
3. Diseñar y codificar un programa que permita el ingreso de un número de legajo y 3 notas de exámenes de un curso de 30 alumnos.
   La información debe ser almacenada en un arreglo de [30][4], donde la primera columna está reservada para el número de legajo.
   Luego de la carga, se debe imprimir toda la información ordenada y finalmente el promedio de cada alumno.