Introducción a la Algorítmica y Programación (3300)

Prof. Ariel Ferreira Szpiniak - aferreira@exa.unrc.edu.ar
Departamento de Computación
Facultad de Cs. Exactas, Fco-Qcas y Naturales
Universidad Nacional de Río Cuarto

Teoría 10

Tipos de Datos Estructurados: Arreglos de registros

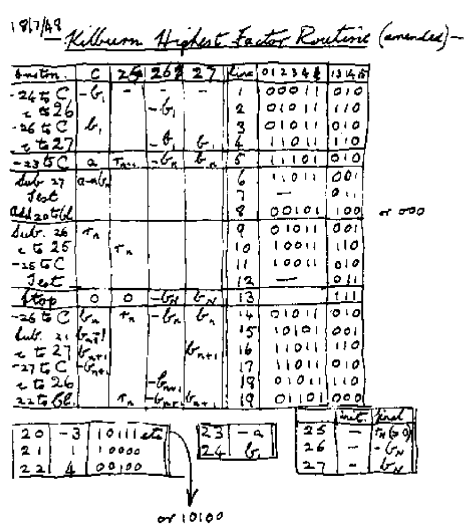
Noticias
Día del programador

Instituido en conmemoración al día en que Kilburn y Williams lograron hacer correr el primer programa sobre una arquitectura de von Neuman. Fue el 21 de junio de 1948.

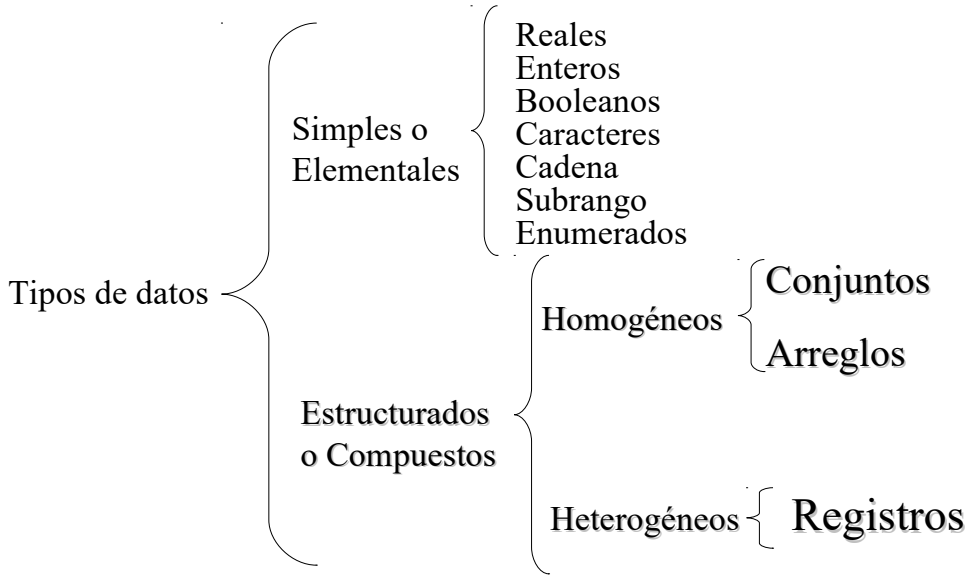
El primer programa fue escrito por Tom Kilburn. Era un programa para encontrar el factor propio más alto de cualquier número a.

Alrededor 130.000 números fueron probados, que tomaron cerca de 2.1 millones de instrucciones

Se ha perdido el programa original, pero Tom Kilburn y Geoff Tootill reconstruyeron el primer programa.



Tipos de datos



Tipos de datos

Pero hay otras clasificaciones también

Tipos de datos

Simples

Estándar: entero, real, carácter, lógico

Definidos por el programador: subrango, enumerado

Estructurados

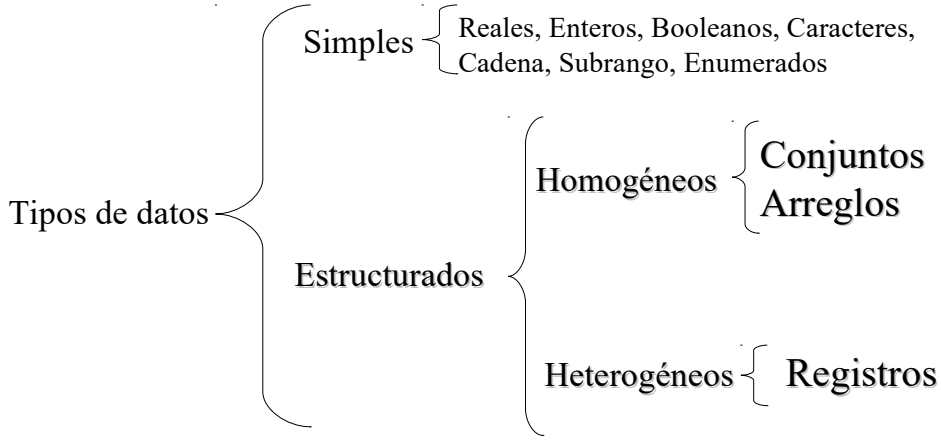
Estáticos: arreglos, registros, conjuntos, cadenas

Dinámicos: listas (pilas/colas), listas enlazadas, árboles, grafos

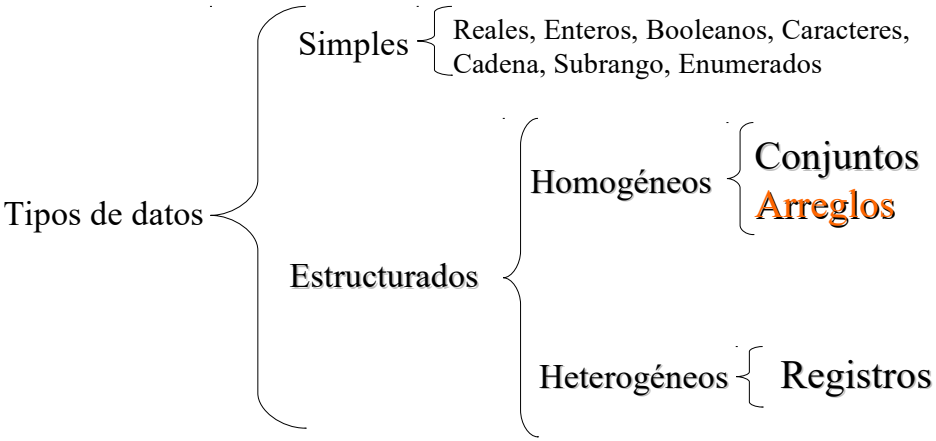
Las **estructuras estáticas** son aquellas en las que el tamaño de memoria ocupado se define antes de que el programa se ejecute y no puede modificarse durante la ejecución.
Las **estructuras dinámicas** son aquellas en las que no se debe definir previamente el tamaño de memoria.
Los **datos simples** tienen en común que cada variable representa un elemento, mientras que en los **estructurados** un identificador puede representar múltiples datos individuales, pudiendo cada uno de estos ser referenciados independientemente.

Tipos de datos

A continuación nos centraremos en el estudio de los *tipos de datos estructurados*. Para ello adoptaremos la primer clasificación:

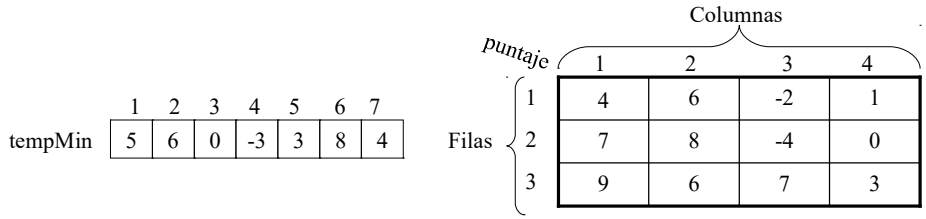


Tipos de datos Arreglos



Arreglos Retomando….

Definición: un arreglo es una colección **finita** (tiene un límite), **homogénea** (elementos del mismo tipo) y "**ordenada"** (se sabe quien es el primero, el segundo, el tercero, etc) de elementos.



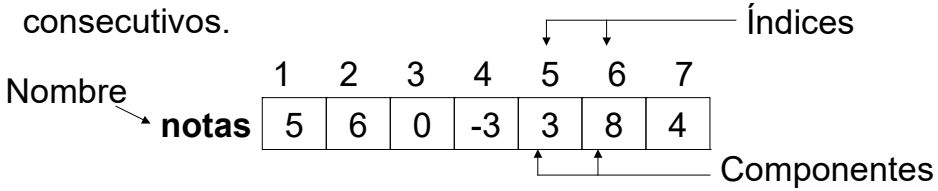
Tiene dos partes importantes: las ***componentes*** y los ***índices***.

Los ***componentes*** hacen referencia a los elementos y los ***índices*** a la posición donde se encuentra el elemento.

Arreglos Unidimensionales

La estructura más simple es el arreglo de una dimensión, también llamado *vector*.

El arreglo está formado por una sucesión de elementos consecutivos.



Para referirse a una componente de un vector se utilizará el nombre y un *índice*.

El índice se encierra entre corchetes ([índice]).

Ejemplo: **notas[5]**

Arreglos Bidimensionales

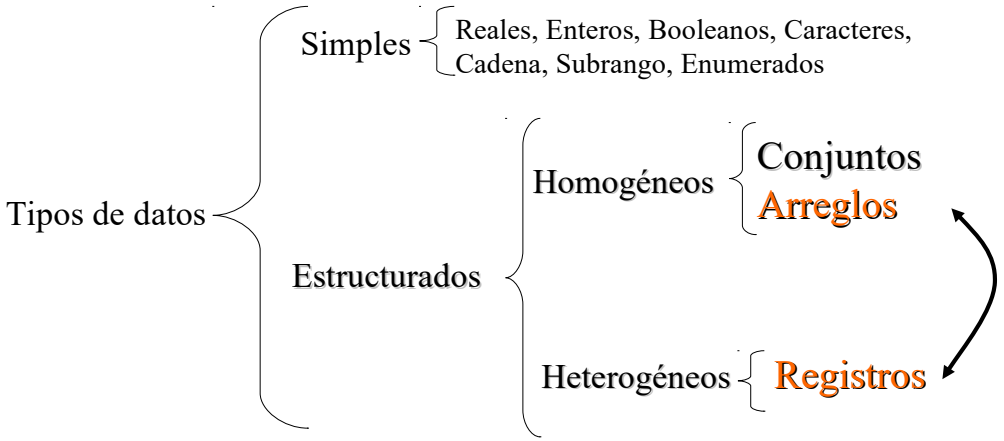
Los arreglos de dos dimensiones, también llamados *matrices*, se utilizan para representar tablas de valores.

Se requieren dos índices, uno para las **filas** y otro para las **columnas**.

Columnas

puntaje	1	2	3	4
Filas 1	4	6	-2	1
2	7	8	-4	0
3	9	6	7	3

Tipos de datos Arreglos y Registros



Arreglos de registros

Aunque los registros pueden ser útiles, muchas aplicaciones requieren de una colección de registros.

- Por ejemplo, se podría necesitar almacenar la información de todos los empleados de una empresa. Para evitar definir una variable para cada empleado, simplemente definimos un arreglo cuyas componentes son registros de tipo empleado.
- Otro caso sería por ejemplo si se desea almacenar la cantidad de milímetros llovidos en Río Cuarto en cada uno de los meses del año, la temperatura máxima registrada en cada mes, la temperatura mínima. En este caso no sería necesario definir 3 arreglos por separado sino que podría definirse:
  - Un registro de tres campos donde cada campo sea un arreglo.
  - O, lo más indicado, un solo **arreglo de registros** donde cada registro contenga 3 campos (lluvia, tempMax y tempMin).

Arreglos de registros
Ejemplo I

Supongamos que se desean guardar los datos de hasta 20 empleados (nombre, teléfono, dirección y edad).

- ¿Cómo definiríamos el tipo empleado?
- ¿Cómo definiríamos el arreglo de empleados?
- ¿Cómo realizaríamos la carga de los 15 registros del arreglo de empleados?
- ¿Cómo traduciríamos todo eso a C?

Arreglos de registros - Ejemplo I

```
Algoritmo CargaEmpleados
Léxico
  Max = 20
  TEmpleado = <nombre ∈ Cadena, telefono ∈ Cadena,
               direccion ∈ Cadena, edad ∈ Entero>
  TArregloEmpleados = arreglo[1..Max] de TEmpleado
  TData = <a ∈ TArregloEmpleados, cant ∈ (0..Max+1)>
  i ∈ Z
  datosEmpleados ∈ TData
Inicio
  datosEmpleados.cant ← 15
  para (i←1, i<=datosEmpleados.cant, i←i+1) hacer
    Entrada:datosEmpleados.a[i].nombre //Ingreso de nombre
    Entrada:datosEmpleados.a[i].telefono //Ingreso de tel
    Entrada:datosEmpleados.a[i].direccion //Ingreso de dir
    Entrada:datosEmpleados.a[i].edad //Ingreso de edad
  fpara
Fin
```


Arreglos de registros - Ejemplo I

```
// CargaEmpleados
#include <stdio.h>
#include <string.h>
#define Max 20
struct empleado{
         char nombre[100];
         char telefono[100];
         char direccion[100];
         int edad;
};
struct data{
         int cant;
         struct empleado a[Max];
};
int main(){
  struct data datosEmpleados;
  int i;
  datosEmpleados.cant=15;
  for (i=0; i<datosEmpleados.cant; i++){
    printf("Ingrese nombre: \n ");
    scanf("%s",datosEmpleados.a[i].nombre);
    printf("Ingrese tel:\n ");
    scanf("%s",datosEmpleados.a[i].telefono);
    printf("Ingrese dir:\n ");
    scanf("%s",datosEmpleados.a[i].direccion);
    printf("Ingrese edad:\n ");
    scanf("%d",&(datosEmpleados.a[i].edad));
  }
return 0;
}
```


Arreglos de registros - Ejemplo II

Una vez que el arreglo (cuyas componentes son registros de tipo empleado) está cargado, podríamos consultar sus valores, modificarlos, etc.

Esta manipulación se realiza de la misma manera que para arreglos cuya componentes son tipos simples, teniendo en cuenta que cada componente es un registro y por lo tanto tengo que acceder a él usando el nombre de cada campo.

Ejemplo: Supongamos que luego de la carga de los 15 empleados deseamos visualizar por pantalla el nombre y la edad de todos ellos. Agregaremos el código necesario al final del anterior...

Arreglos de registros - Ejemplo II

```
Algoritmo CargaMuestraEmpleados
Léxico
  Max = 20
  TEmpleado = <nombre ∈ Cadena, telefono ∈ Cadena,
               direccion ∈ Cadena, edad ∈ Entero>
  TArregloEmpleados = arreglo[1..Max] de TEmpleado
  TData = <a ∈ TArregloEmpleados, cant ∈ (0..Max+1)>
  i ∈ Z
  datosEmpleados ∈ TData
Inicio
  datosEmpleados.cant ← 15
  para (i←1, i<=datosEmpleados.cant, i←i+1) hacer
    Entrada:datosEmpleados.a[i].nombre //Ingreso de nombre
    Entrada:datosEmpleados.a[i].telefono //Ingreso de tel
    Entrada:datosEmpleados.a[i].direccion //Ingreso de dir
    Entrada:datosEmpleados.a[i].edad //Ingreso de edad
  fpara
  para (i←1, i<=datosEmpleados.cant, i←i+1) hacer
    Salida:datosEmpleados.a[i].nombre //Mostrar nombre
    Salida:datosEmpleados.a[i].edad //Mostrar edad
  fpara
Fin
```


Arreglos de registros - Ejemplo II

```
//CargaMuestraEmpleados
#include <stdio.h>
#include <string.h>
#define Max 20
struct empleado{
  char nombre[100]; char telefono[100]; char direccion[100]; int edad;
};
struct data{
  int cant; struct empleado a[Max];
};

int main(){
  struct data datosEmpleados;
  int i;
  datosEmpleados.cant=15;
  for (i=0; i<datosEmpleados.cant; i++){
    printf("Ingrese nombre: \n "); scanf("%s",datosEmpleados.a[i].nombre);
    printf("Ingrese tel:\n "); scanf("%s",datosEmpleados.a[i].telefono);
    printf("Ingrese dir:\n "); scanf("%s",datosEmpleados.a[i].direccion);
    printf("Ingrese edad:\n "); scanf("%d",&(datosEmpleados.a[i].edad));
  }
  for (i=0; i<datosEmpleados.cant; i++){
      printf("Nombre del empleado: %s \n",datosEmpleados.a[i].nombre);
      printf("Edad del empleado: %d \n",datosEmpleados.a[i].edad);
  }
return 0;
}
```


Arreglos de registros
Ejemplo III

Un docente necesita almacenar hasta 120 evaluaciones de sus estudiantes. Cada evaluación está compuesta por el nombre del estudiante y la nota.

```
Algoritmo Evaluaciones
Léxico
  Max = 250
  TEval = <nombre ∈ Cadena, nota ∈ (1..10)>
  TNumeros = arreglo[1..Max] de TEval
  TData = <a ∈ TNumeros, cant ∈ (0..Max+1)>
  misNotas ∈ TData
Inicio




Fin
```


Arreglos de registros
Ejemplo IV

Un meteorólogo necesita almacenar la cantidad de milímetros llovidos en Río Cuarto la temperatura máxima registrada en cada mes y la temperatura mínima, para cada uno de los meses del año. Luego desea saber cuánto es el total de precipitación anual, el mes en que aconteció la temperatura máxima y el mes en que aconteció la mínima. Finalmente necesita informarlo.

Arreglos de registros
Ejercicio V

1. Rehacer el Ejemplo II definiendo dos acciones con parámetros, una para la carga, y otra para listar el nombre y edad de cada empleado.
2. Agregar una nueva acción que permita al usuario listar los **n** primeros empleados, con **n** a elección.
3. Agregar al algoritmo principal un menú que permita la usuario elegir:
   a) Cargar 15 y luego Mostrar nombre y edad de los 15.
   b) Cargar entre 1 y 20 (a elección) y luego listar todos lo empleados recientemente cargados.
4. Traducir a C

Bibliografía

- Scholl, P. y J.-P. Peyrin, "Esquemas Algorítmicos Fundamentales: Secuencias e iteración", Barcelona, Ed. Masson, 1991.

- Lucas, M., J.-P. Peyrin y P. Scholl, "Algorítmica y Representación de Datos. Tomo 1: Secuencia, Autómata de estados finitos", Barcelona, Ed. Masson, 1985.

- Watt, David, "Programming Language Concepts and Paradigms", Prentice-Hall International Series in Computer Science (1990).

- Biondi, J. y G. Clavel, "Introducción a la Programación. Tomo 1: Algorítmica y Lenguajes", 2° ed., Barcelona: Masson, 1985.

- Clavel, G. y Biondi, J., "Introducción a la Programación. Tomo 2: Estructuras de Datos", 2° ed., Barcelona: Masson, 1985.

- De Guisti, A. "Algoritmos, datos y programas. Con aplicaciones en Pascal, Delphi y Visual Da Vinci. Prentice Hall.

- Joyanes Aguilar, L., "Programación en Turbo Pascal". Mc Graw Hill, 1993.