Universidad Nacional de Asunción
Facultad Politécnica

Algorítmica II

Licenciatura en Ciencias de la Informática

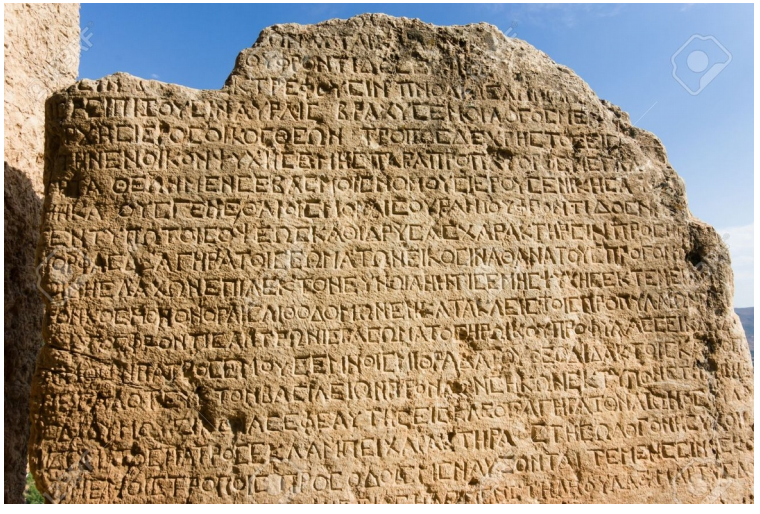
Archivos

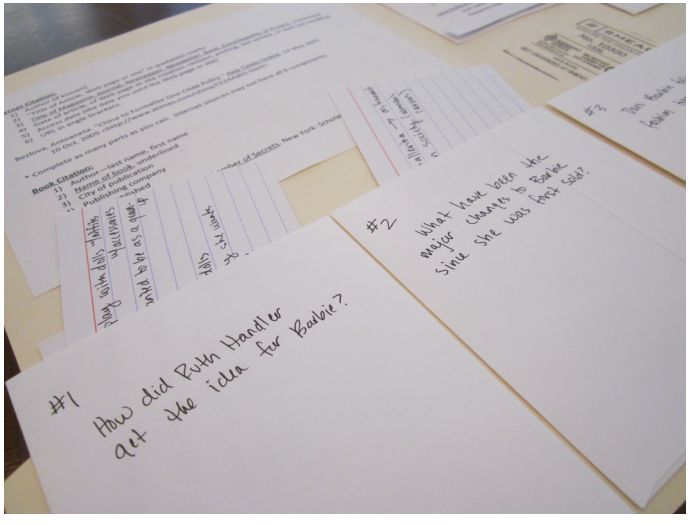
Prof. M.Sc. Gustavo Daniel Sosa Cabrera
<gdsosa@pol.una.py>

En esta presentación

- Archivos
  - Conceptos y definiciones
  - Organización
  - Procesamiento de archivos en C
  - El puntero a un archivo

¿Archivos?




#1
How did Ruth Handler get the idea for Barbie?
#2
What were some major changes in Barbie since she was first sold?


Files

Noción de archivo

- Un archivo o fichero es un conjunto de datos estructurados en una colección de entidades elementales o básicas llamadas *registros* los cuales a su vez se componen de *campos*.
- Los archivos pueden tener tratamiento posterior gracias a que son almacenados en *soportes externos* a la memoria principal como ser los discos.
- Hoy en día es posible almacenar archivos en las *nubes* (googledrive, dropbox, etc.).

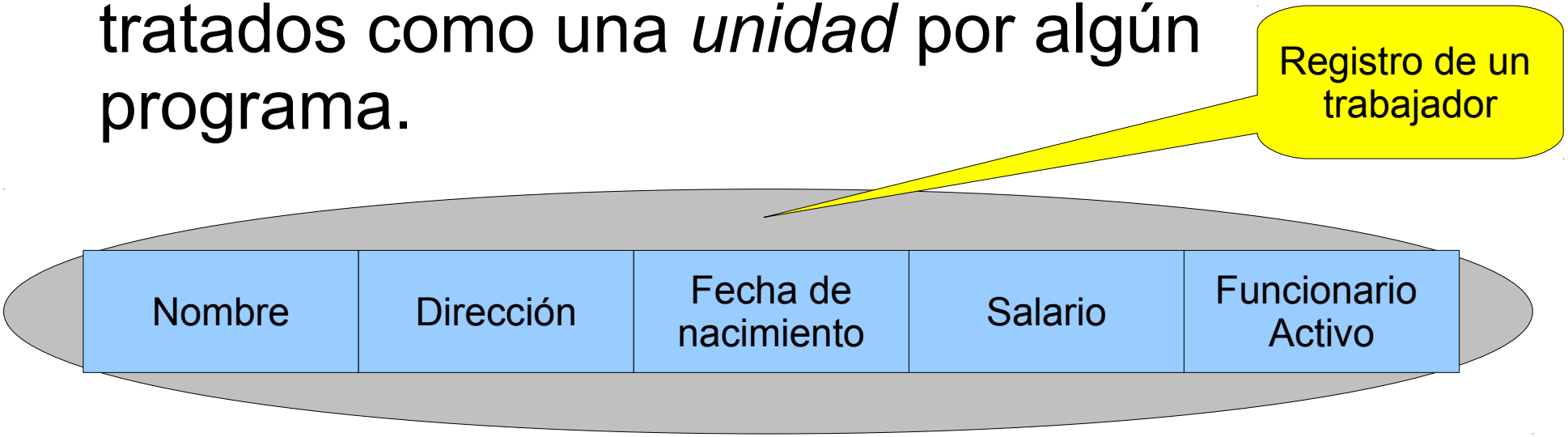
Campos

- Los caracteres se agrupan en campos de datos, el cual es un ítem o dato elemental que representan alguna información y que se encuentra caracterizado por su *tamaño* y su *tipo de dato*.

Nombre	Dirección	Fecha de nacimiento	Salario	Funcionario Activo

Registros

- Un registro es una colección de información, normalmente relativa a una entidad particular.
- Un registro es una colección de campos lógicamente relacionados que pueden ser tratados como una *unidad* por algún programa.



Organización de archivos

- La organización de un archivo define la forma en la que los registros se *disponen* sobre el soporte de almacenamiento, o bien la forma en que se estructuran los datos en un archivo, en general se consideran:
  - Organización secuencial (“sequential”)
  - Organización directa o aleatoria (“random”)
  - Organización secuencial indexada (“indexed”)

Organización secuencial

- Los registros se graban consecutivamente cuando el archivo se crea y se debe acceder consecutivamente cuando se leen dichos registros.
- El orden físico en que fueron grabados (escritos) los registros es el orden de lectura de los mismos.
- Esta organización es soportada por todos los tipos de dispositivos de *memoria secundaria*.

Organización directa

- Los datos son accedidos en forma *aleatoria* mediante su posición, es decir, el lugar relativo que ocupan.
- Los registros deben contener una clave.
- Debe haber una *correspondencia* entre los posibles valores de las claves y las direcciones disponibles en el soporte.
- Se necesita de una función de conversión de claves o **función hash**.

Organización secuencial indexada

- Un *diccionario* es un archivo secuencial, cuyos registros son las entradas y cuyas claves son las palabras definidas por las entradas.
- Las letras y las cabeceras de páginas se almacenarán en un archivo índice independiente del archivo de datos.
- El tipo de sus registros contiene un campo *clave identificador*.

Operaciones sobre archivos

- Las principales operaciones que se pueden realizar sobre un archivo luego de haber definido su organización, son:
  - Creación
  - Consulta
  - Actualización
    - Altas
    - Bajas
    - Modificaciones

Gestión de archivos en C

- Los archivos en C se manipulan según el siguiente protocolo:
  - Se abre el archivo en modo lectura, escritura o cualquier otro modo válido.
  - Se trabaja con el archivo escribiendo o leyendo datos según el modo de apertura escogido.
  - Se cierra el archivo.

El puntero a un archivo

- Un puntero a un archivo es una *referencia* a una información que define varias cosas sobre él, incluyendo el nombre, el estado y la posición actual del archivo.
- En esencia identifica un archivo especifico y utiliza la secuencia asociada para dirigir el funcionamiento de las funciones de E/S con *búffer*.

El puntero a un archivo

- Un puntero a un archivo es una variable de tipo puntero al tipo FILE que se define en la librería <*stdio.h*>.
- Un programa necesita utilizar punteros a archivos para leer o escribir en los mismos.
- Para obtener una variable de este tipo se utiliza una secuencia o declaración como esta:

```
FILE *fd;
```

Apertura de archivos en C

- fopen(): función que abre un fichero.
- Recibe la ruta de un fichero (una cadena) y el modo de apertura(otra cadena) y devuelve un objeto de tipo FILE *.
- Su prototipo es:

```
FILE * fopen (char ruta[ ], char modo[ ]);
```

- Si se produce un error cuando se está intentando abrir un archivo, *fopen()* devuelve un puntero nulo (NULL).

Modos de apertura

Modo	**Significado**
r	Abre un archivo de texto para lectura (read).
w	Crea un archivo de texto para escritura (write).
a	Abre un archivo de texto para añadir (append).
r+	Abre un archivo de texto para lectura/escritura.
w+	Crea un archivo de texto para lectura/escritura.
a+	Añade o crea un archivo de texto para lectura/escritura.

Lectura y escritura de archivos

- Se utilizan las funciones fscanf() y fprintf() los cuales se comportan exactamente igual a las funciones *scanf*() y *printf*() excepto que operan sobre **archivos**.
- Sus prototipos son:

  int fscanf (FILE * *fichero*, char formato[ ], *direcciones*);

  int fprintf (FILE * *fichero*, char formato[ ], *valores*);

- Donde *fichero* es un puntero al archivo devuelto por una llamada a fopen().

Fin y cierre de archivos

- La función feof() devuelve 1 si estamos al final del fichero y 0 en caso contrario.
- Su prototipo es:

```
int feof (FILE * fichero);
```

- La función fclose() cierra un fichero.

  Recibe el FILE * devuelto por una llamada previa a fopen().
- Su prototipo es:

```
int fclose (FILE * fichero);
```

Ejemplo de lectura de un archivo

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main( int argc, char *argv[] )
{
   FILE *fd;
   char cadena[50];
   // Obtenemos el nombre del archivo de la linea de comandos.
   if (argc != 2 ) {
       printf("Debe especificar el nombre del archivo...\n");
       exit(1);
   }
   if ( ( fd = fopen(argv[1], "r" ) ) == NULL ) {
       printf("No se pudo abrir el archivo...\n");
       exit(1);
   }
   fscanf(fd, "%s", cadena);
   while ( ! feof(fp) ) {
      printf("%s\n", cadena);
      fscanf(fd, "%s", cadena);
   }
   fclose(fd);
   return 0;
}
```

Ejemplo de escritura en un archivo

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
int main( int argc, char *argv[] )
{
   FILE *fp;
   char cadena[50];
   if (argc != 2 ) {
       printf("Debe especificar el nombre del archivo...\n");
       exit(1);
   }
   if ( ( fp = fopen(argv[1], "w" ) ) == NULL ) {
       printf("No se pudo abrir el archivo...\n");
       exit(1);
   }
   printf("Ingrese las cadenas a grabar y FIN para terminar\n ");
   scanf("%50s", cadena);
   while ( strcmp(cadena, "FIN") ) {
      fprintf(fp, "%s \n", cadena);
      scanf("%s", cadena);
   }
   fclose(fp);
   return 0;
}
```

Procesamiento de archivos

- Dado el archivo tipo CSV *(**C**omma **S**eparated **V**alues)* “notas_finales.csv” el cual posee por cada registro los siguientes campos separados por coma (,):
  - Cedula: numérico.
  - Nombre y Apellido: cadena con espacios.
  - Puntaje final: numérico de 0 a 100.
- Se debe producir con la escala (100-95: 5, 94-80:4, 79-70:3, 69-60:2, 59-0: 1) la siguiente lista:

  Alumno :Nota Final

Ejemplo de procesamiento

```c
#include <stdio.h>
#define N 30
#define NOMBRE_ARCHIVO "notas_finales.dat"
typedef struct{char nombre[N] ;short puntaje;} Alumno;
short obtenerNota(short);

int main(void){
   Alumno alumno;
   FILE *fd = fopen(NOMBRE_ARCHIVO, "r");
   if (fd == NULL){
        printf("No se pudo abrir el archivo...\n");
        return -1;
   }
   // Ignoramos el campo cedula en la lectura de los campos y leemos
   // el nombre completo del alumno hasta encontrar una coma.
   fscanf(fd,"%*ld,%[^,],%hd", alumno.nombre, &alumno.puntaje);
   while(!feof(fd)){
      printf("%s\t:%hd\n", alumno.nombre, obtenerNota(alumno.puntaje));
      fscanf(fd,"%*ld,%[^,],%hd", alumno.nombre, &alumno.puntaje);
   }
   return 0;
}
// Calcula la nota en base a la escala del 60%.
short obtenerNota(short puntaje){...}
```

¿ Preguntas ?

References

- Joyanes Aguilar, Luis. Fundamentos de Programación-Algorítmos y Estructuras de Datos / Luis Joyanes Aguilar. -- Madrid : McGraw-Hill, 1990.-- 702 p.
- Segovia Silvero, Juan. SL. Introducción al lenguaje - Referencia de subrutinas predefinidas – Ejemplos selectos. Centro Nacional de Computación, Universidad Nacional de Asunción. 1999.
- Schildt, Herbert. C Manual de referencia. -- Madrid : Osborne/McGraw-Hill, 1991.-- 749 p.