

Resolución de Problemas y Algoritmos

Clase 1000 (8 en decimal)
Almacenamiento en Memoria.
Sistemas Operativos.
Archivos secuenciales en Pascal.



Joseph C.R. Licklider

Dr. Alejandro J. García
<http://cs.uns.edu.ar/~ajg>




Departamento de Ciencias e Ingeniería de la Computación
 Universidad Nacional del Sur
 Bahía Blanca - Argentina

Conceptos: memoria principal y memoria secundaria

Memoria principal
 RAM (Random Access Memory)
 Contiene: programas en ejecución y datos

Memoria secundaria
 Ejemplos: disco rígido, pen-drive, DVD, tarjeta SD (Secure Digital), unidad de estado sólido(SSD), "la nube".
 Contiene: Archivos de programas, textos, datos, fotos, videos, música.



bus: canal de comunicación

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 2

Memoria secundaria: disco duro



Un disco duro es un sistema de grabación magnética con uno o más platos o discos rígidos, unidos por un mismo eje que gira a gran velocidad

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 3

Memoria secundaria: DDS (Solid State Drive)



Una unidad de estado sólido o SSD (acrónimo de solid-state drive) es un dispositivo de almacenamiento de datos que usa una memoria "flash" no volátil (no es un disco).

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 4

Memoria secundaria (SSD - solid state drive)

Una **unidad de estado sólido** o SSD (acrónimo de solid-state drive) es un dispositivo de almacenamiento de datos que usa una memoria no volátil como la memoria flash en lugar de los platos giratorios magnéticos encontrados en los discos duros convencionales.

En comparación con los discos duros, las SSD son menos sensibles a los golpes, son prácticamente inaudibles, y son más rápidas.

A veces se traduce erróneamente en español la "D" de SSD como "disco" cuando, en realidad, representa la palabra "drive", que podría traducirse como unidad o dispositivo.

A partir de 2010, la mayoría de las SSDs utilizan memoria flash basada en compuertas NAND, que retiene los datos sin alimentación.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 5

Almacenamiento en "la nube"

El **almacenamiento en la nube** (*cloud storage*) es un servicio de almacenamiento remoto de datos. Este servicio le da al usuario la ilusión de que tiene sus datos en forma remota en un único repositorio, pero en realidad puede estar compuesto por muchos recursos y servidores distribuidos en diferentes lugares geográficos.

De esta manera, posee gran tolerancia a fallos porque permite redundancia y distribución de datos.

El concepto de **computación en la nube** (cloud computing) fue introducido en los años '60 por el científico **Joseph Carl Robnett Licklider**.

http://es.wikipedia.org/wiki/Almacenamiento_en_nube



Joseph C.R. Licklider



Dropbox



Google Drive



iCloud



OneDrive (SkyDrive)

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 6

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
"Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016

Conceptos: almacenamiento en memoria

Memoria RAM: (Random Access Memory)
 El procesador debe accederla lo más rápido posible. Para lograr que la memoria tenga gran velocidad de acceso su tecnología es de alto costo (en 2016, 4Gb a \$600). ¿Cuánto costaría 1 TB de RAM?
 Por su tecnología es volátil (los datos no perduran al cortar la energía del sistema). Los valores de las variables de tipos integer, char, real y boolean se almacenan en RAM.
http://es.wikipedia.org/wiki/Memoria_de_acceso_aleatorio

Memoria secundaria:
 Por su tecnología (magnética, óptica, flash) permite que los valores perduren aún cuando no tenga energía eléctrica.
 Como contrapartida, es una memoria de menor velocidad de acceso y por ello de mucho menor costo. Ejemplos en 2016: Disco rígido 1Tb \$1000 (\$1 el Gb) - Pendrive de 16Gb \$100 - SSD de 250Gb \$2500
 Los valores de los archivos están en memoria secundaria.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 7

Jerarquía de Memorias

En un sistema, combinando diferentes tipos de memorias, se busca tener la mayor capacidad de almacenamiento, equilibrando velocidad y costo.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 8

¿Quién conoce todo el Hardware?

El Sistema Operativo

¿Qué es un Sistema Operativo?

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 9

Conceptos: Sistema Operativo (Operating System)

- Un **sistema operativo (SO)** es un programa que gestiona los recursos de hardware y provee servicios a los programas de aplicación. Se ejecuta en modo privilegiado respecto de los restantes programas.
- Es un programa que actúa como un intermediario entre un usuario y el hardware de la computadora.

Cada carrera tiene una materia para SO:

- **Ing:** "Sistemas operativos"
- **Lic:** "Sistemas operativos y distribuidos"

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 10

Sistemas operativos

- Es importante el "concepto" y no el "producto" porque en su carrera y trabajo profesional usará muchos sistemas operativos.
- Estos son algunos de **los que he usado** hasta ahora:
 - RSTS (en PDP-11) y VMS (en VAX)
 - MS-DOS, Windows (3, 3.11, NT, XP, Vista, 7, 8),
 - Unix, Linux, AIX, Symbian, Android,
- Puede mirar más sobre sistemas operativos en:
 - http://es.wikipedia.org/wiki/Sistema_operativo
 - http://en.wikipedia.org/wiki/Mobile_operating_system
- Un resumen de la cronología histórica de los sistemas operativos
 - http://en.wikipedia.org/wiki/Timeline_of_operating_systems

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 11

Sistema de Archivos (file system)

Por ejemplo:

- Un sistema operativo conoce los detalles del hardware del almacenamiento secundario (como un disco rígido) y provee servicios a los programas de aplicación para el **manejo de archivos**.
- Puede haber archivos de video, de música, de imágenes, de texto, de programas ejecutables, etc.
- Observe que el nombre del archivo le permite al sistema operativo asociarlo con una aplicación.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 12

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016

Nombres de archivos en un SO

- Un nombre válido para un archivo dependerá del SO.
- Por ejemplo, en las primeras versiones del sistema operativo **MS-DOS** (1981-1995), un nombre de archivo tenía el formato: **nnnnnnnn.EEE** (8 caracteres para el **nombre** y 3 para la **extensión**)
- La extensión sigue siendo usada por los SO como una forma rudimentaria de identificar el tipo de archivo (ej: MP3, AVI, PAS, EXE, JPG), y asociarlo a una aplicación.
- Windows** actualmente limita a 260 caracteres, incluyendo el camino o ruta (path) y el nombre. No se pueden usar los símbolos **V/?:*"><|** La extensión la forman los caracteres después del último punto (no es obligatorio que sean 3).
- Ejemplo: **C:\usuarios\ale\RPA\clase-1(ale).pru.num.s.pas**

camino o ruta
nombre

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 13

Archivos en Pascal

- Muchas veces es útil que determinados **valores puedan perdurar** aunque el programa termine, y que estos valores **puedan ser utilizados** en el futuro por **otro programa**.
- En Pascal se puede crear un **tipo de dato estructurado** para manejar un **archivo** (en inglés **FILE**) y de esta forma almacenar valores que pueden perdurar aún cuando la ejecución del programa termine.
- De esta manera, un programa podrá "leer" elementos generados por otro programa; y además "escribir" datos que podrá leer otro programa (o él mismo pero en otra ejecución posterior).

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 14

Conceptos: Archivos

- Para que el contenido de un archivo perdure más allá de la ejecución de un programa y aún cuando la computadora estuviera apagada por un tiempo, los archivos **residen en memoria secundaria** (como un disco rígido o DVD).
- Es importante notar que el acceso a memoria secundaria **depende del Sistema Operativo** usado.
- El manejo de archivos en Pascal también puede tener **diferencias de un compilador a otro**.
- En esta materia **se verán algunos conceptos de archivos secuenciales** y algunos detalles estarán ligados al sistema operativo o al compilador.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 15

CONCEPTOS: Archivos secuenciales

- Un Sistema Operativo (SO) puede manejar distintas clases de archivos (de texto, fotos, películas, música, ejecutables).
- Cada Lenguaje de Programación (LP) pueden manejar algunas de estas clases de archivos.
- En Pascal hay una clase de archivo que se denomina archivo secuencial**.
- Un archivo secuencial es una **estructura compuesta por una secuencia de elementos** (componentes) donde hay un orden lineal entre ellos.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 16

Archivos secuenciales en Pascal

- En Pascal, un **archivo secuencial** es una **sucesión finita** de **componentes** que pueden accederse una a una, comenzando de la primera.
- Todos los **componentes** deben ser del **mismo tipo** de datos (ej. todos **integer**, o todos **char**).
- Por ser un archivo secuencial**, una vez accedida la primera componente, el acceso a la componente de la posición P se logra luego de haber accedido previamente a la posición P-1.
- La **cantidad** de componentes es **potencialmente infinita** (su límite estará dado por la cantidad de espacio en la computadora donde está el archivo).
- Son almacenados en **Memoria Secundaria**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 17

Conceptos:

Tipo de Dato: define el **conjunto de valores** posibles que puede tomar una variable, **las operaciones** que pueden aplicarse, y cual es la **representación interna**.

Los tipos de datos en Pascal se pueden dividir en:

- Simple**
Ejemplos que vimos en RPA: **INTEGER, REAL, CHAR, BOOLEAN**
- Estructurados**
Ejemplos que veremos en RPA: **FILE OF ...** (archivos de datos) y **TEXT** (archivo de texto)

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 18

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
“Resolución de Problemas y Algoritmos. Notas de Clase”. Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016

Archivos de datos

La palabras reservadas **FILE** y **OF** se utilizan en Pascal como un constructor de un nuevo tipo de dato (estructurado) creado por el programador sobre la base de otros tipos ya existentes. Esto permite al programador trabajar con **archivos de datos**. Como se puede ver abajo, el programador puede crear un nuevo tipo de dato archivo y luego declarar variables de ese tipo, como así también directamente declarar variables de tipo archivo.

El identificador reservado **TYPE** indica que lo que sigue son declaraciones de nuevos tipos de datos creados por el programador.

Program ejemplo;
TYPE nuevo_tipo = FILE OF integer;
 archivo_letras = FILE OF char;
VAR numeros, valores : nuevo_tipo;
 temperaturas, llluvias: FILE OF real;
 letras: archivo_letras;
 decisiones: FILE OF boolean;

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 19

Observaciones

En Pascal, la palabra reservada **TYPE** permite al programador crear nuevos tipo de datos, sobre la base de otros tipos ya existentes. Vea el diagrama sintáctico que está en "bloque".

Ejemplo: **Program ejemplo;**
TYPE nuevo_tipo = FILE OF integer;
VAR valores : nuevo_tipo;

Observación importante sobre los archivos declarados con FILE OF: estos archivos no son archivos de texto, son archivos de datos, y por lo tanto no podrá ver o editar su contenido con editores de texto como por ejemplo el "notepad". Para disponer de archivos de texto se debe usar el tipo predefinido **TEXT** (que veremos pronto).

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 20

Representación gráfica

Program ejemplo;
VAR numeros : FILE OF integer;

Dado que una variable de tipo archivo es una secuencia de componentes del mismo tipo, es usual usar la siguiente **representación gráfica**:

Primer elemento.

Último elemento.

En el archivo, cada componente es de tipo integer.

numeros : 11 -2 5 -12 4 (EOF) Marca que significa End Of File (fin del archivo)

Aunque la capacidad de un archivo es potencialmente infinita, en cualquier momento dado, el archivo tendrá un número finito de componentes.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 21

Nombres de archivos secuenciales en Pascal

- El **identificador** de una variable de **tipo archivo** es un **nombre interno** usado por el programa en Pascal para referirse a un archivo secuencial (llamado manejador de archivo).
- Como los valores almacenados en los **archivos** van a estar en memoria secundaria (por ejemplo: disco rígido), el **Sistema Operativo necesita asignarle un nombre válido**.
- Este archivo puede ser usado en el futuro por otro programa usando ese nombre dado en el Sistema Operativo.
- La **primitiva predefinida de Pascal ASSIGN** permite vincular el manejador de archivo (identificador de variable) y el nombre del archivo en el Sistema Operativo.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 22

Primitiva ASSIGN

Program ejemplo;
VAR numeros: FILE OF integer;
Begin
ASSIGN(numeros, 'mis-numeros.datos');
 ...

- La primitiva **ASSIGN(F, N)** tiene dos parámetros: **F** que es un identificador de variable de tipo archivo, y **N** que es una secuencia de caracteres que representa un nombre válido de archivo en el sistema operativo.
- El identificador **F** es llamado **manejador del archivo** de nombre **N** (*file handler*), y en el código fuente toda otra referencia al archivo se hace usando el manejador **F**.
- Una vez ejecutada **assign** vincula a **F** con **N** (el nombre real del archivo en memoria secundaria).

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 23

Primitivas de Pascal para archivos secuenciales

- assign(F,N):** vincula F con N (nombre del archivo en SO).
- rewrite(F):** crea un archivo nuevo con el nombre N que está vinculado a F (si ya existe otro archivo con ese nombre N se sobrescribe y se pierde el viejo archivo).
- write(F,e):** en un archivo F creado con rewrite, escribe el valor de "e" a continuación del último elemento de F.
- close(F):** cierra el archivo vinculado al *manejador* F.

(veamos un ejemplo antes de ver las tres restantes)

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 24

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016

Ejemplo

Problema: escriba un programa para crear un archivo llamado "mis-numeros.dat" y escribir en él enteros de 1 a un tope ingresado por el usuario.

Algoritmo

Crear el archivo "mis-numeros.dat" para escribir en él.
 Solicitar un entero tope
 Escribir en el archivo "tope" enteros desde el 1 hasta tope
 Cerrar el archivo

fin.

Observación: En este algoritmo conozco de antemano cuanto elementos quiero escribir en el archivo, entonces en Pascal puedo usar FOR.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 25

Programa "crear"

Problema: escriba un programa para crear un archivo llamado "mis-numeros.dat" y escribir en él enteros de 1 a un tope ingresado por el usuario.

Program crear;
VAR nuevo: FILE OF integer;
 valor,tope: integer;

Observación: Si tope=100 generará un archivo de 400 bytes.

begin
 assign(nuevo, 'mis-numeros.dat');
 rewrite(nuevo); {crea archivo vacío y permite escribir en él}
 writeln('Cantidad de enteros a escribir en el archivo ');
 readln(tope);
for valor:= 1 **to** tope **do**: write(nuevo,valor);
 close(nuevo);
end.

Escribe un entero al final del archivo

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 26

Primitivas de Pascal para archivos secuenciales

- **assign(F,N):** vincula F con N (nombre del archivo en SO).
- **rewrite(F):** crea un archivo nuevo con el nombre N que está vinculado a F (si ya existe otro archivo con ese nombre N se sobrescribe y se pierde el viejo archivo).
- **write(F,e):** en un archivo F creado con rewrite, escribe el valor de "e" a continuación del último elemento de F.
- **close(F):** cierra el archivo vinculado al *manejador* F.
- **reset(F):** abre un archivo existente de nombre N para leer, y queda preparado para leer el primer elemento.
- **read(F,e):** lee un elemento del archivo F, copia el valor leído en "e" y queda preparado para leer el siguiente elemento (si existe) o queda en el fin del archivo.
- **eof(F)** (end of file – fin de archivo): retorna TRUE si se llegó al final de un archivo o FALSE en caso contrario.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 27

Lectura de datos de un archivo

Considere las siguientes variables:
VAR F: FILE OF integer; e: integer;
reset(F): abre el archivo asociado al manejador F para leer, y queda preparado para leer el primer elemento.

F: 11 -2 5 -12 (EOF) e: ?

read(F,e): lee el elemento del archivo F que está listo para ser leído, copia el valor leído en la variable "e" y queda preparado para leer el siguiente elemento (si existe) o queda en el fin del archivo.

F: 11 -2 5 -12 (EOF) e: 11

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 28

Lectura de datos de un archivo

Considere las siguientes variables:
VAR F: FILE OF integer; e: integer;

F: 11 -2 5 -12 (EOF) e: 5

Si ya fueron leídos los tres primeros elementos, como se muestra en la figura de arriba, al hacer **read(F,e)** se lee el último elemento y se llega al final del archivo (end of file) como muestra la figura siguiente:

F: 11 -2 5 -12 (EOF) e: -12

¿Qué ocurre si ejecuto **read(F,e)** ahora?

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 29

Lectura de datos de un archivo

Considere las siguientes variables:
VAR F: FILE OF integer; e: integer;

F: 11 -2 5 -12 (EOF) e: -12

MUY IMPORTANTE: si la primitiva "**read(F,E)**" es usada sobre el fin de un archivo (o un archivo vacío) es considerado un **error de programación** ya que dará un error y el programa dejará de ejecutarse.

Por lo tanto antes de usar "**read(F,e)**" debe asegurarse no estar al final del archivo con la función **eof(F)**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 30

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016

Lectura de datos de un archivo

Considere las siguientes variables:
VAR F: FILE OF integer; e: integer;

Al hacer "reset(F)" puede ocurrir que el archivo esté vacío y se tiene una situación como se muestra a continuación:

F: (EOF)


MUY IMPORTANTE: si la primitiva "read(F,E)" es usada sobre el fin de un archivo (o un archivo vacío) es considerado un **error de programación** ya que dará un error y el programa dejará de ejecutarse.

Por lo tanto antes de usar "read(F,e)" debe asegurarse no estar al final del archivo con la función **eof(F)**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 31

Lectura de datos de un archivo

```

...
assign(F, 'enteros');
reset(F);
read(F,e);
writeln(' primero:', e);
...
    
```

MAL Error de programación: no controla si el archivo está o no vacío

MUY IMPORTANTE: si la primitiva "read(F,E)" es usada sobre el fin de un archivo (o un archivo vacío) es considerado un **error de programación** ya que dará un error y el programa dejará de ejecutarse.

Por lo tanto antes de usar "read(F,e)" debe asegurarse no estar al final del archivo con la función **eof(F)**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 32

Lectura de datos de un archivo

<pre> ... assign(F, 'enteros'); reset(F); if not eof(F) then begin read(F,e); writeln(' primero:', e); end ... </pre> <p style="text-align: center;">OK</p>	<pre> ... assign(F, 'enteros'); reset(F); while not eof(F) do begin read(F,e); writeln(' encontré:', e); end ... </pre> <p style="text-align: center;">OK</p>
--	--

IMPORTANTE: antes de usar "read(F,e)" debe asegurarse no estar al final del archivo con la función **eof(F)**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 33

Lectura de datos de un archivo

```

...
assign(F, 'enteros');
reset(F);
repeat
read(F,e);
writeln(' encontré:', e);
until eof(F)
...
    
```

MAL Error de programación: no controla si el archivo está o no vacío

IMPORTANTE: antes de usar "read(F,e)" debe asegurarse no estar al final del archivo con la función **eof(F)**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 34

Ejemplo "leer"

Problema: escriba un programa para abrir un archivo ya existente llamado "mis-numeros.dat", leer todos sus componentes, y mostrarlos por pantalla.

Algoritmo

Abrir el archivo "mis-numeros.dat" para leer sus elementos
 Leer uno a uno los elementos y mostrarlos en pantalla
 Cerrar el archivo.

fin.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 35

Ejemplo "leer"

Problema: escriba un programa para abrir un archivo ya existente llamado "mis-numeros.dat", leer todos sus componentes, y mostrarlos por pantalla.

Program leer;

VAR arch_num: FILE OF integer; elemento: integer;

begin

```

assign(arch_num, 'mis-numeros.dat');
reset(arch_num); {abre el archivo para leer de él}
while not eof(arch_num) do {mientras no llegue al fin}
begin
read(arch_num,elemento); writeln(' fue leído:', elemento);
end;
close(arch_num);
end.
    
```

Lee un elemento del archivo y queda preparado para leer el siguiente.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 36

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 "Resolución de Problemas y Algoritmos. Notas de Clase". Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016

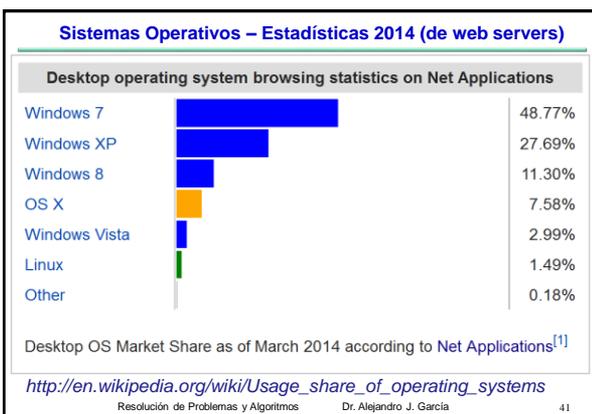
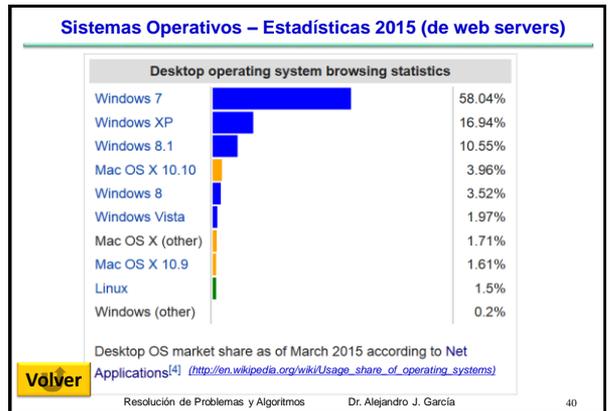
Problemas propuestos

- **Problema:** escriba un programa que cuente **cuantos elementos** tiene el archivo "mis-numeros.dat" (ya creado y con números en él).
- **Problema:** escriba un programa que busque **cuantas veces está** el elemento E (ingresado por el usuario) en el archivo "mis-numeros.dat" (ya creado y con números en él).
- **Problema:** escriba un programa que vea si **primer** elemento es **igual** al **último** en el archivo "mis-numeros.dat" (ya creado y con números en él).
- **Problema:** escriba un programa que vea si los elementos del archivo "mis-numeros.dat" (ya creado y con números en él) **están ordenados de menor a mayor**.

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 37

Información adicional

Resolución de Problemas y Algoritmos Dr. Alejandro J. García 38

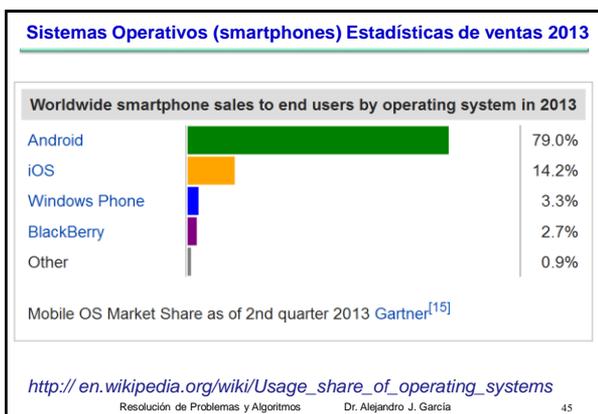
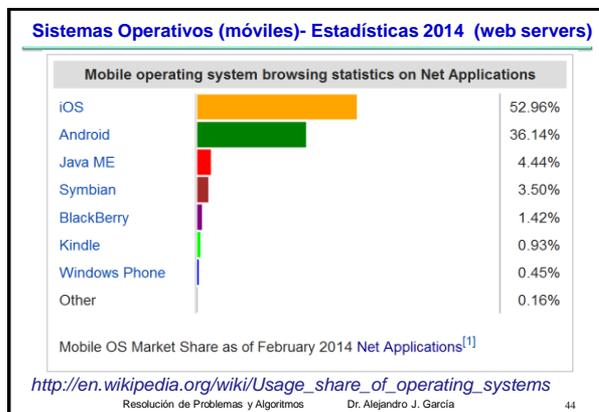
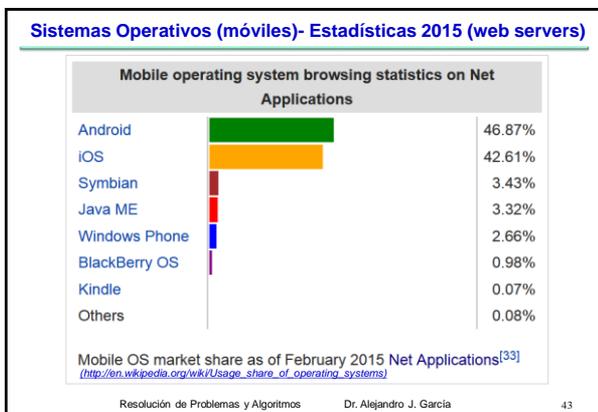


Sistemas Operativos – Estadísticas (web servers)

Operating Systems 2012			Operating Systems 2013		
1	Windows 7	39.47%	1	Windows 7	43.69%
2	Windows XP	29.38%	2	Windows XP	22.59%
3	Apple OS X	8.79%	3	iOS	9.50%
4	Windows Vista	7.77%	4	Apple OS X	8.45%
5	Apple iOS	5.25%	5	Windows Vista	5.28%
6	Android	1.78%	6	Android	4.19%
7	Linux	1.76%	7	Windows 8	2.72%
8	BlackBerry	0.57%	8	Linux	1.89%
9	SymbianOS	0.18%	9	BlackBerry	0.57%
10	Windows 8	0.08%	10	SymbianOS	0.26%

<http://www.w3counter.com/globalstats.php>
Resolución de Problemas y Algoritmos Dr. Alejandro J. García 42

El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
“Resolución de Problemas y Algoritmos. Notas de Clase”. Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016



El uso total o parcial de este material está permitido siempre que se haga mención explícita de su fuente:
 “Resolución de Problemas y Algoritmos. Notas de Clase”. Alejandro J. García. Universidad Nacional del Sur. (c) 20/04/2016