Outline

1. Files and Databases
   - mass storage
   - hash functions

2. Dictionaries
   - logical key values
   - managing persistent data with dbm files

MCS 260 Lecture 7
Introduction to Computer Science
Jan Verschelde, 21 June 2023
mass storage and dictionaries

1. Files and Databases
   - mass storage
   - hash functions

2. Dictionaries
   - logical key values
   - managing persistent data with dbm files
Mass Storage
tapes and disks

Mass storage means

1. the data is persistent (opposed to volatile),
2. large capacity: giga or terabytes.

We distinguish modes of access:

- *sequential* access: one must rewind tapes,
- *direct* access: read disks from any position.

We distinguish two different technologies:

- a *magnetic* file covers disk and tape surfaces,
- optical disc media rely on *laser* technology.

Compression software also helps increasing capacity.
Units to measure Capacity

1 byte = 8 bits. Large quantities are expressed in thousands (kilo), millions (mega), billions (giga), and trillions (tera).

<table>
<thead>
<tr>
<th>units</th>
<th>value</th>
<th>value in full</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kb = kilobyte</td>
<td>$2^{10} \approx 10^3$</td>
<td>1,024</td>
</tr>
<tr>
<td>Mb = megabyte</td>
<td>$2^{20} \approx 10^6$</td>
<td>1,048,576</td>
</tr>
<tr>
<td>Gb = gigabyte</td>
<td>$2^{30} \approx 10^9$</td>
<td>1,073,741,824</td>
</tr>
<tr>
<td>Tb = terabyte</td>
<td>$2^{40} \approx 10^{12}$</td>
<td>1,099,511,627,776</td>
</tr>
</tbody>
</table>

The same prefixes (kilo, mega, giga, tera) measure clock speed of the CPU, or other frequencies.

- 1 hertz = 1 cycle per second
- 1 kilohertz = $2^{10}$ cycles per second
- 1 megahertz = $2^{20}$ cycles per second
- 1 gigahertz = $2^{30}$ cycles per second
I/O Disk Operations
reading from and writing information to disk

- A disk consists of a number of horizontal platters, covered by a magnetic coating to store data on a surface.
- A buffer in main memory holds the entire block of data prior to writing to or after being read from disk.
- The seek is the movement of the heads towards the required track. The seek time is the time of a seek.
- The latency time is the time to wait for the required sector to pass beneath the read/write head. On average this equals half the rotation time.
- Time needed for one i/o operation:

\[ t_{i/o} = t_{\text{seek}} + t_{\text{latency}} + t_{\text{transfer}}. \]
Flash Drives
the memory stick

Commonly used portable mass storage.

- connect to USB port, which powers the drive
  USB = Universal Serial Bus
- capacity goes to several gigabytes
- sends electronic signals to chambers of silicon dioxide,
  altering the characteristics of small electronic circuits

Advantages and disadvantages:

+ unlike a disk drive, there is no movement,
  sometimes faster than optical disks
- can sustain only limited number
  of write and erase cycles

Linux commands to check disk usage: `df`, `du`.
mass storage and dictionaries

1. **Files and Databases**
   - mass storage
   - hash functions

2. **Dictionaries**
   - logical key values
   - managing persistent data with dbm files
Data is organized in *logical records*. One record in a phone book has three fields:

```
| name | address | phone number |
```

An input/output block can contain several records.

The usage factor is

\[
\frac{\text{# bytes allocated to logical records}}{\text{# bytes of physical blocks on file}}
\]
Sequential File Organization

order records sequentially

- Every record on file has a key. Records are stored in order of the keys. In a phone book, with names sorted alphabetically, the key is usually the name.

- Binary search is an efficient way to search through a sorted data collection.

- The main problem with sequential file organization is the insertion of new elements.

- Solutions to this problems are
  1. store changes in a separate file that is then periodically merged with the main file
  2. leave free blocks between records
  3. use an overflow zone to insert new data
Hash-based File Organization

Keys are generated by a *hash algorithm*. The hash algorithm uses a *hash function*, mapping logical key values (for example, a name) to a physical address (or a position). Goal: even distribution of keys over addresses.

- Mapping names into addresses via combinations of the ASCII codes of the characters in the strings representing the names is a first step.

- Advantage: fast access, reduced search speed. Disadvantage: two different key values could be mapped to the same address.
mass storage and dictionaries

1 Files and Databases
- mass storage
- hash functions

2 Dictionaries
- logical key values
- managing persistent data with dbm files
Dictionaries

- With lists or tuples, the index must be a number.
  
  But very often, we list data using names as indices. Consider for example a telephone directory.

- A **dictionary** is an unordered set of `key:value` pairs, where `value` can be of any data type. The type of `key` must admit an ordering, it must be "hashable".

For example, list summer sales according to month:

```python
>>> sales = { 'jun':123, 'aug':342, 'sep' : 212 }
>>> sales
{'jun': 123, 'aug': 342, 'sep': 212}
>>> sales['aug']
342
```
mass storage and dictionaries

1. Files and Databases
   - mass storage
   - hash functions

2. Dictionaries
   - logical key values
   - managing persistent data with dbm files
DBM File Operations
an overview

<table>
<thead>
<tr>
<th>Python code</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>import dbm</td>
<td>load module dbm</td>
</tr>
<tr>
<td>f = dbm.open('n','c')</td>
<td>create or open dbm file with name n</td>
</tr>
<tr>
<td>f['key'] = 'value'</td>
<td>assign value for key</td>
</tr>
<tr>
<td>f.keys()</td>
<td>returns the keys</td>
</tr>
<tr>
<td>value = f['key']</td>
<td>load value for key</td>
</tr>
<tr>
<td>count = len(f)</td>
<td>number of entries stored</td>
</tr>
<tr>
<td>found = 'key' in f</td>
<td>see if entry for key</td>
</tr>
<tr>
<td>del f['key']</td>
<td>remove entry for key</td>
</tr>
<tr>
<td>f.close()</td>
<td>close dbm file</td>
</tr>
</tbody>
</table>

Typical use:

- every record in database has unique key,
- values are dictionaries, stored as strings.
Exercises

1. Use a dictionary to record state capitols.
2. Store the money exchange rates between dollar, euro, and yen in a dictionary and illustrate how to convert any sum of money.
3. Make a dictionary to store the antiderivation rules for common trigonometric functions, $\sin$, $\cos$, and $\tan$.
4. Define a dictionary that has as keys the name of the months and as values the number of the month. Use that dictionary to convert '21 June 2023' into '21/06/2023'. To insert the '0' for the '6', put 02.0f in the second part of the f-string.
5. Use dbm for storing a mileage table.
6. In Italian, the numerals are written as zero, uno, due, tre, etc. Setup a dictionary with the written numerals as keys and their corresponding values. Make a quiz prompting the user to give the corresponding value of a number written in Italian.