Computer Science Fundamentals

Seminar 6: OS| Python: Dictionaries

Lecturer: Olga Yugay

Agenda

- Lecture based Q&A
- File system using Terminal

Python

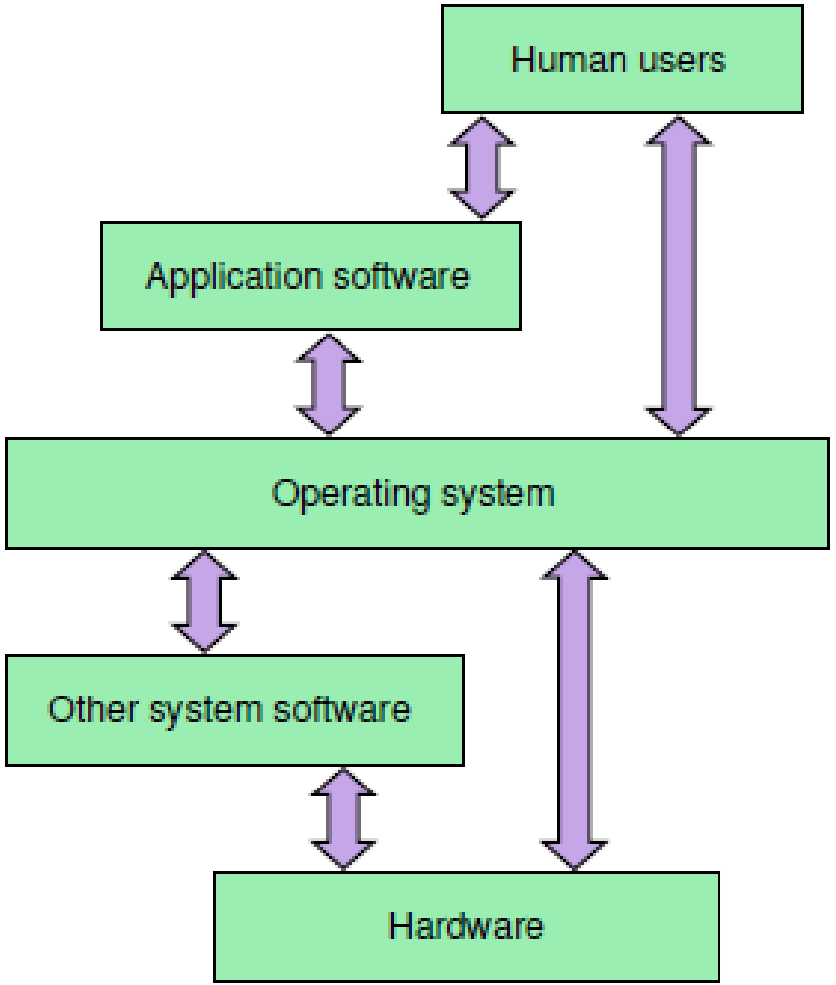
- Dictionaries
- Add item to a dictionary
- Create a new dictionary
- Look up value
- len and in
- values()
- Looping and dictionaries
- Reverse lookup

Part A:

Lecture recap

Refer to the image and

- define OS
- list OS main roles



File manager: file systems

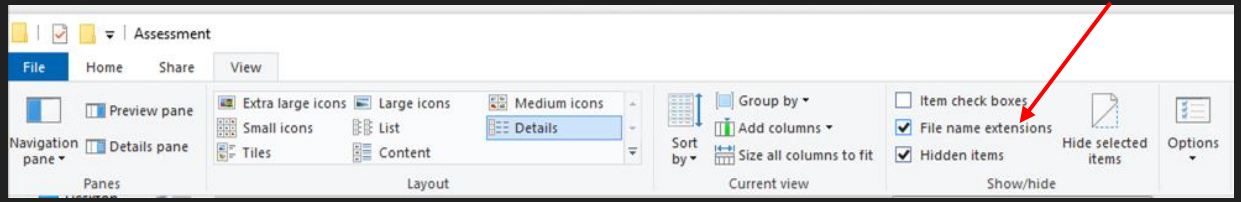
- **File** - a named collection of data, used for organizing secondary memory
- **File system** - the operating system's logical view of the files it manages
- **Directory** - a named group of files
- **File type** - the specific kind of information contained in a file, such as a Java program or a Microsoft Word document
- **File extension** - Part of a file name that indicates the file type

Extensions	File type
txt	text data file
mp3, au, wav	audio file
gif, tiff, jpg	image file
doc, wp3	word processing document
java, c, cpp	program source files

Task 1: Display the extensions

1. Click the Start menu. …
2. Type "folder options" (without the quotes). …
3. A dialog box with the title "Folder Options" will appear. …
4. Click to uncheck the box for "Hide extensions for known file types".
5. Click the "OK" button at the bottom of the dialog box.
6. Navigate to a folder with many different files, e.g. Downloads folder
7. Note their extensions.

Alternatively:



For MacOS
https://www.howtogeek.com/714021/how-to-show-all-filename-extensions-on-the-mac/

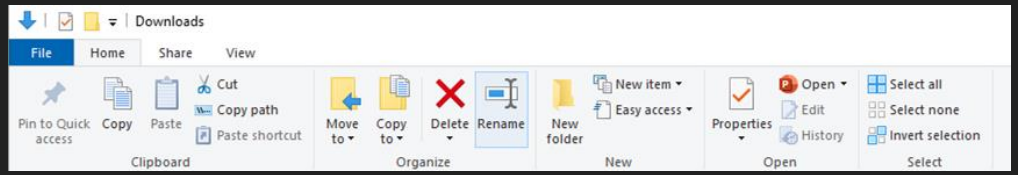
Task 2: Rename multiple files in bulk*

1. Open File Explorer.
2. Browse to the folder that includes all the files you want to rename.
3. Select all the files.

Quick Tip: You can use the Ctrl + A keyboard shortcut to select all the files.

1. Press the F2 key to rename it.
2. Type a new name for the file(s) and press Enter.

OR



*It may be useful for uploading files to an environment that does not support space or non-English characters in filenames, and batch renaming can be used to substitute characters with acceptable one. Special programs can be used for advanced cases

For MacOS: https://www.imore.com/how-rename-multiple-files-once-mac

https://www.howtogeek.com/207503/how-to-quickly-batch-rename-files-on-windows-mac-os-x-or-linux/

Task 3: See file metadata

Display File metadata (file owner, permissions, file size, date modified, date created, file type). Right click on file or folder and access Properties.

How to change file associations with a program in Windows (1 method)

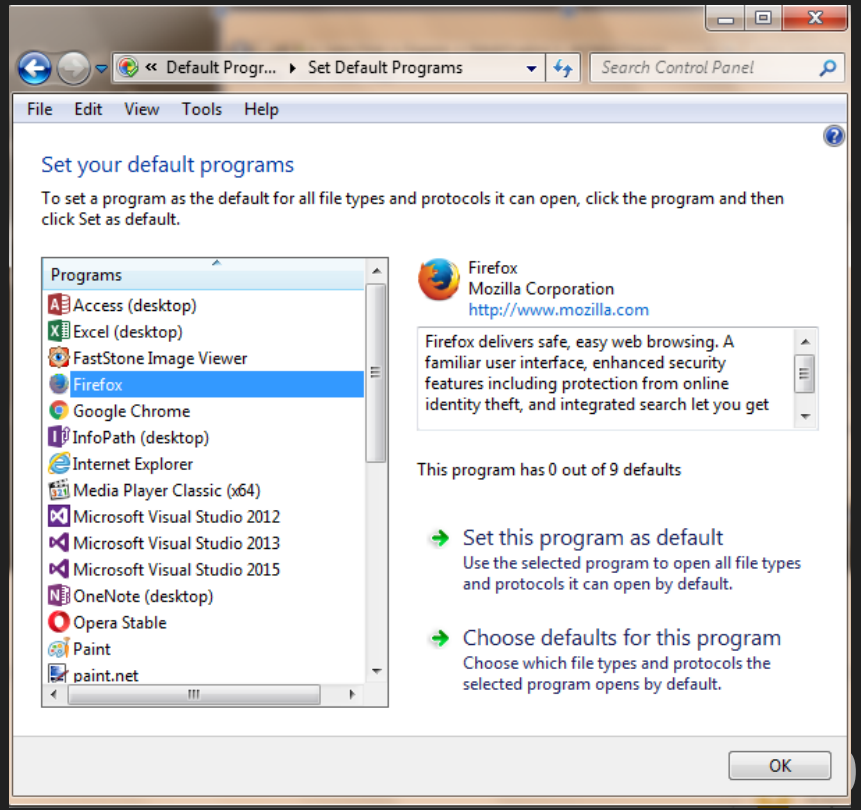
1. Open My Computer or Explorer. (Windows key + E to open Explorer)
2. Right-click on the file you want to change the file association and click Properties.
3. In the file Properties, click the Change button next to the "Opens with" option.
4. After clicking Change, you'll be given a list of programs to open the file.
5. Select the program you want to use and then click OK or Apply. If the program you want to use is not listed, click the Browse button, find the program's executable (.exe) file on the computer, and click OK to select that program.

How to change file associations with a program in Windows (2 method)

1. Open the Control Panel.
2. In Control Panel, search for Default Programs and open the icon.
3. Click on a program and either se this program as default

   OR

   Choose defaults for this program



Task 4

Change the default program for opening py and related files (e.g. switch to PyCharm)

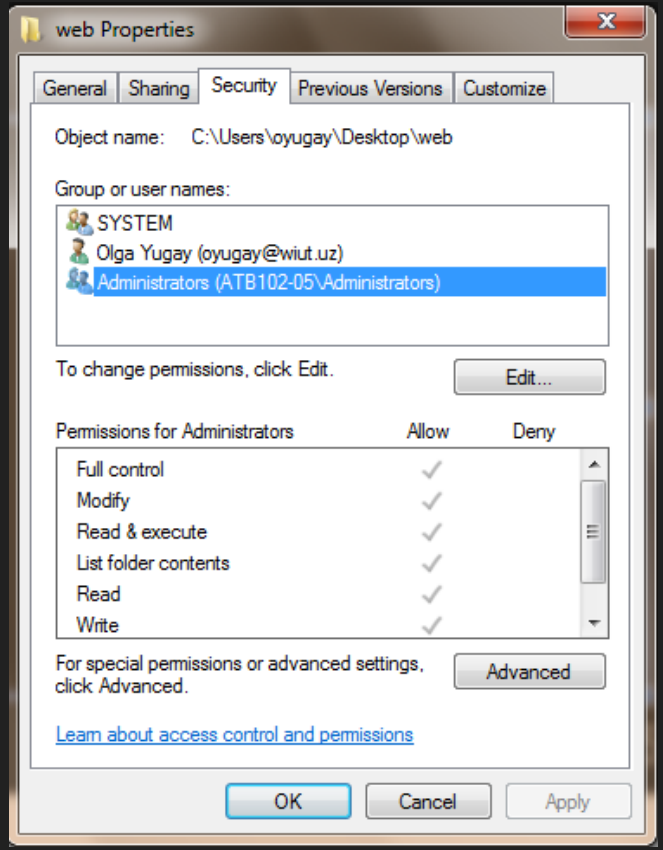
File Manager: file protection

- In multiuser systems, file protection is of primary importance. That is, we don't want one user to be able to access another user's files unless the access is specifically allowed.
- It is the operating system's responsibility to ensure valid file access.
- Different operating systems administer their file protection in different ways.

	Read	Write/Delete	Execute
Owner	Yes	Yes	No
Group	Yes	No	No
World	No	No	No

Task 5 Change access permissions

1. Navigate to Documents folder
2. Right click on folder, access the security tab and change the access permissions for Administrators/other users



File Manager: directories (recap seminar 2)

- A directory, in most operating systems, is represented as a file. The directory file contains data about the other files in the directory.

- For any given file, the directory contains the file name, the file type, the address on disk where the file is stored, and the current size of the file.

- The directory also contains information about the protections set up for the file. It may also contain information about when the file was created and when it was last modified.

Directories: path

- Path - a text designation of the location of a file or subdirectory in a file system
- Absolute path - path that begins at the root and includes all successive subdirectories

Linux/Unix	Windows
/etc/samba.smb.conf /boot/grub/grub.conf	C:\Program Files\Microsoft Office\Office12\VISSHE.DLL D:\MyFolder\CSF\Lecture7.pptx

- Relative path - path that begins at the current working directory

Linux/Unix	Windows
../samba.smb.conf ../grub.conf	..\Office12\VISSHE.DLL ..\CSF\Lecture7.pptx

Task 6: review absolute vs relative path

1. Open command line
2. Navigate to desired folder using relative path
3. Navigate to desired folder using absolute path
4. Which method is most applicable when?

Windows vs Unix directory tree

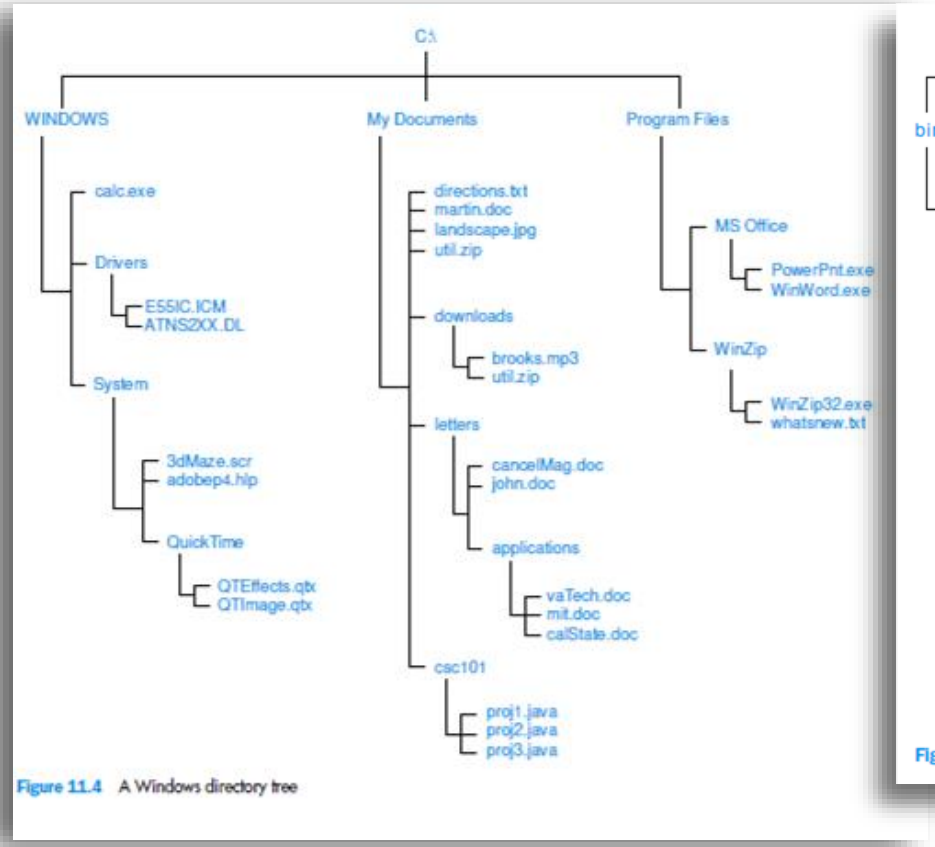


Figure 11.4 A Windows directory tree

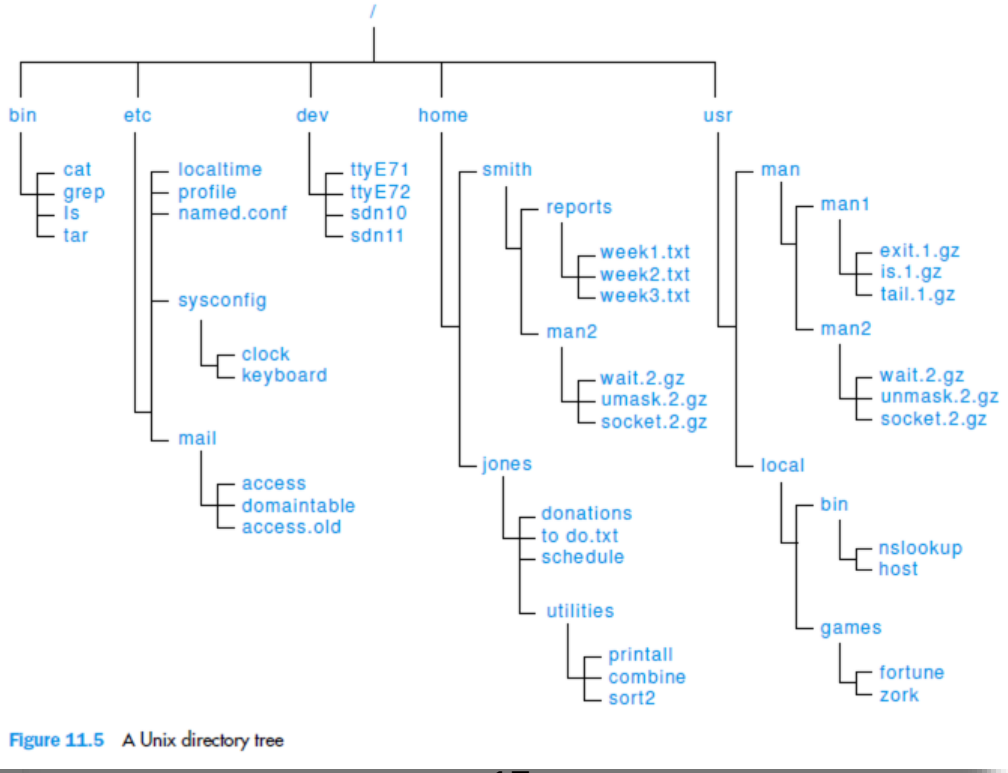


Figure 11.5 A Unix directory tree

Part B: Python

Dictionary

Built-in type, which is building blocks of many efficient algorithms

Dictionary is like a list but more general

A dictionary contains a collection of indices, which are called **keys**, and a collection of **values**

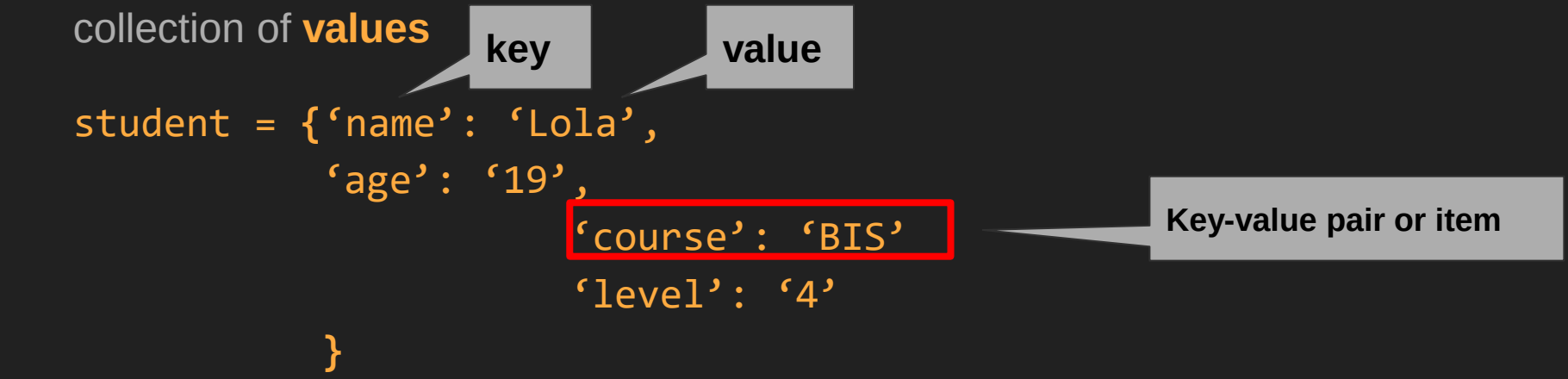

```
student = {‘name’: ‘Lola’,
           ‘age’: ‘19’,
                  ‘course’: ‘BIS’
                  ‘level’: ‘4’
           }
```

Dictionary

Built-in type, which is building blocks of many efficient algorithms

Dictionary is like a list but more general

A dictionary contains a collection of indices, which are called **keys**, and a collection of **values**



Dictionary is a mapping

A dictionary represents a mapping from keys to values, so you can also say that each key “maps to” a value

```
>>> eng2uz = dict()

>>> eng2uz

{}
```

Curly brackets represent an empty dictionary

Add items to dictionary

```
>>> eng2uz['one'] = 'bir'

>>> eng2uz

{'one': 'bir'}
```

Create a new dictionary

```
>>> eng2uz = {'one': 'bir', 'two': 'ikki', 'three': 'uch'}
```

Try print eng2uz

```
>>> eng2uz
```

```
{'one': 'bir', 'three': 'uch', 'two': 'ikki'}
```

The order of the key-value pairs might not be the same. In general, the order of items in a dictionary is unpredictable.

It is not a problem because the elements of a dictionary are never indexed with integer indices. Instead, **keys** are used to look up the corresponding values

Looking up value

You use the keys to look up the corresponding values

```
>>> eng2uz['two']

'ikki'


>>> eng2uz['four']

KeyError: 'four'
```

len()

Remember len function? It works on dictionaries as well!

```
>>> len(eng2uz)

3
```

in

in operator works on dictionaries, it tells you whether something appears as a key in the dictionary (appearing as a value is not good enough).

```
>>> 'one' in eng2uz
```

```
True
```

```
>>> 'uno' in eng2uz
```

```
False
```

values()

To see whether something appears as a value in a dictionary, you can use the method **values**, which returns a collection of values, and then use the in operator:

```
>>> vals = eng2uz.values()
```

```
>>> 'bir' in vals
```

```
True
```

Python dictionaries use a data structure called a **hashtable** that has a remarkable property: the **in** operator takes about the same amount of time no matter how many items are in the dictionary

Task 7

Write a function that would calculate average for marks presumably received for a set of modules. Use in operator and len() method

For example:

```
myFinalMarks = {'CSF': 75, 'FunPro': 80, 'WT': 85}
```

You can enhance task by getting input from user and storing it in the dictionary

Git push updates

Dictionary as a collection of counters

Suppose you are given a string and you want to count how many times each letter appears. Your options:

1. Create 26 variables, one for each letter of the alphabet. Then traverse the string and, for each character, increment the corresponding counter, probably using a chained conditional.

2. Create a list with 26 elements. Then convert each character to a number (using the built-in function `ord`), use the number as an index into the list, and increment the appropriate counter.

3. Create a dictionary with characters as keys and counters as the corresponding values. The first time you see a character, you would add an item to the dictionary. After that you would increment the value of an existing item.

Potential solution

```
def histogram(s):
    d = dict()
    for c in s:
        if c not in d:
            d[c] = 1
        else:
            d[c] += 1
    return d
```

empty dictionary is created

for loop traverses the string

if the character c is not in the dictionary, we create a new item with key c and the initial value 1

If c is already in the dictionary we increment d[c].

Try it

```
>>> h = histogram('brontosaurus')
```

```
>>> h
```

```
{'a': 1, 'b': 1, 'o': 2, 'n': 1, 's': 2, 'r': 2, 'u': 2, 't': 1}
```

The histogram indicates that the letters 'a' and 'b' appear once; 'o' appears twice, and so on.

get ()

Dictionaries have a method called get that takes a key and a default value. If the key appears in the dictionary, get returns the corresponding value; otherwise it returns the default value

```
>>> h = histogram('a')
>>> h
{'a': 1}
>>> h.get('a', 0)
1
>>> h.get('c', 0)
0
```

Task 8

Write code that would calculate final mark for the module. Marks and weighting are stored in the dictionary. Use get() method

Example:

```
csf = {
'cw1-weight': 0.4,
'cw1-mark':79,
'exam-weight':0.6,
'exam-mark':65
}
```

Git push the task to github repository

Looping and dictionaries

If you use a dictionary in a for statement, it traverses the keys of the dictionary. For example, print_hist prints each key and the corresponding value:

```
def print_hist(h):
    for c in h:
        print(c, h[c])
>>> h = histogram('parrot')
>>> print_hist(h)
a 1
p 1
r 2
t 1
o 1
```

You have to have histogram code from slide 32

Why are keys are in no particular order?

Reverse lookup

Given a dictionary **d** and a key **k**, it is easy to find the corresponding value

<=This operation is called a **lookup**.

what if you have **v** and you want to find **k**?

Problems:

1. there might be more than one key that maps to the value v. Depending on the application, you might be able to pick one, or you might have to make a list that contains all of them.
2. there is no simple syntax to do a reverse lookup; you have to search.

Reverse lookup

```
def reverse_lookup(d, v):
    for k in d:
        if d[k] == v:
            return k
    raise LookupError()
```

The raise statement causes an **exception.**

In this case it causes a LookupError, which is a built-in exception used to indicate that a lookup operation failed.

Example of a successful reverse lookup:

```
>>> h = histogram('parrot')

>>> key = reverse_lookup(h, 2)

>>> key

'r'
```

Recall slide 37

```
def print_hist(h):
    for c in h:
        print(c, h[c])
>>> h = histogram('parrot')
>>> print_hist(h)
a 1
p 1
r 2
t 1
o 1
```

Example of a unsuccessful reverse lookup:

```
>>> key = reverse_lookup(h, 3)
Traceback (most recent call last):
    File "", line 1, in
    File "", line 5, in reverse_lookup
LookupError
```

Explore Linux family

Ubuntu 20.10: What's New?

Homework 1: Explore Virtual box and Ubuntu

1. Download and install https://www.virtualbox.org/wiki/Downloads
2. Open Oracle VM Virtual Box Manager
3. Install Desktop Ubuntu version https://ubuntu.com/#download
4. Explore basic apps
5. Run basic apps using Run command and shortcuts
6. Read more on Ubuntu
   https://www.lifewire.com/beginners-guide-to-ubuntu-2205722
   https://www.makeuseof.com/tag/ubuntu-an-absolute-beginners-guide/
1. Create files and directories using LX Terminal

Virtual Box beginner's guide https://www.youtube.com/watch?v=sB_5fqiysi4

Install and experiment with Ubuntu https://www.youtube.com/watch?v=lmeDvSgN6zY

Homework 2: Explore terminal commands

Open terminal and explore the following commands

1. Pwd - Print working directory
2. Ls - list the files in current directory
3. Create a file

   touch myfile.txt - to create a file

   echo "Hello world!" > myfile.txt - to write to file

   cat myfile.txt - to print the contents of a file

   gedit myfile.txt - to open the file in Text editor gedit

1. Mkdir seminar7 - to make a directory seminar7
2. Cd seminar7 - to change directory, access seminar7 directory

Homework 3: Explore terminal commands

Create several .txt files in seminar7 directory. Use commands below to accomplish the tasks listed:

cp - copy files

mv - move files

rm - delete files

rmdir - remove directory

1. Create a new directory task10
2. Copy 1 file to task10
3. Move 1 file from seminar 7 to task10
4. Delete files and then directory task10

Homework 4

Explore and experiment with commands in Ubuntu terminal:

https://www.howtogeek.com/412055/37-important-linux-commands-you-should-know/

https://garywoodfine.com/linux-terminal-command-cheat-sheets/

https://help.ubuntu.com/lts/installation-guide/armhf/apcs02.html

Homework 5

- Downey, Think Python, Chapter 11 end of the Chapter exercises
- https://www.jetbrains.com/help/pycharm/contribute-to-projects.html

Solution to seminar 6 task 6 (set as homework)

It is one of many possible options. Feel free to share your solutions

https://github.com/wiut-tutor/sem-6/blob/main/task6.py

References

1. Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 11
2. Downey, A. (2015). Think python. O'Reilly. Chapter 6, 8
3. Hamedani, M., 2020. Python for Beginners - Learn Python in 1 Hour. [online] Youtube.com.
4. *Contribute to projects on github: Pycharm*. PyCharm Help. (n.d.). Retrieved June 8, 2022, from https://www.jetbrains.com/help/pycharm/contribute-to-projects.html
5. McKay, D. (2019, May 8). *37 important linux commands you should know*. How. Retrieved June 8, 2022, from https://www.howtogeek.com/412055/37-important-linux-commands-you-should-know/
6. Gary WoodfineTechnical Director at threenine.co.ukGary is Technical Director at threenine.co.uk. (2021, April 28). *Linux and Ubuntu Terminal Command Reference Cheat Sheets*. Gary Woodfine. Retrieved June 8, 2022, from https://garywoodfine.com/linux-terminal-command-cheat-sheets/
7. *Ubuntu documentation*. Official Ubuntu Documentation. (n.d.). Retrieved June 8, 2022, from https://help.ubuntu.com/