

Lecture 16, 13 October 2026

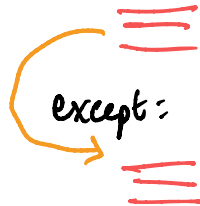
Madhavan Mukund

<https://www.cmi.ac.in/~madhavan>

Programming and Data Structures with Python

Exception handling

try:
 ≡
 ≡
except:
 ≡
 ≡



Examples using input()

always returns str

```
try:  
    ≡  
except ValueError:  
    ≡  
except KeyError:  
    ≡  
except: — catch everything, last
```

try:

≡

except (IndexError, KeyError):

≡

except ValueError:

≡

except:

≡

User defined exceptions

"catching" exceptions

"throwing" exception - in Python "raising" exception

class List

- only nonnegative numbers

raise ValueError ("message")

Can also "create" new exceptions, but look it up

Using exceptions "positively"

if k in d :

$d[k] = d[k] + 1$

else:

$d[k] = 1$

try:

$d[k] = d[k] + 1$

except KeyError:

$d[k] = 1$

String format

- Substitute arguments
- Control how it is displayed

"width" of output

"type" of output

int



str



width +

no. of digits after .

← float

Terminology

Archane & obscure

C $\approx$ 1970

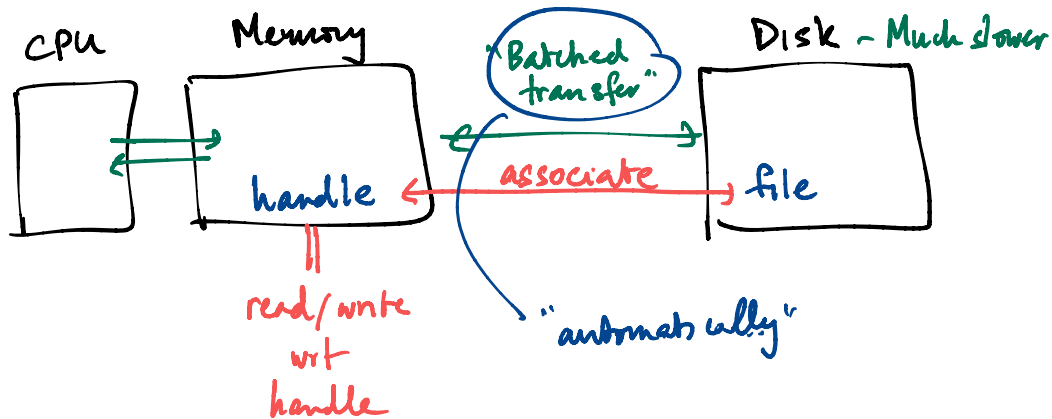
d = "decimal" = int

s = str

f = float

Back to input & output

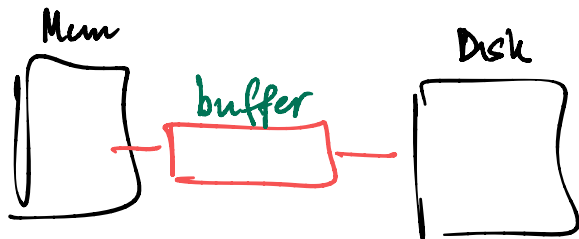
large amount of data - read & write files



"open a file" — associate a variable ("handle")
to filename

read/write wrt handle

"close a file" — disconnect handle, propagate all
updates to disk



Can "flush" the buffer

Read vs write

Transferring data sequentially

Cannot both read & write a file at the same time

Open the file — specify read or ^{over}write/append

Sequential read

