Strings and Files

David H Smith IV

University of Illinois Urbana-Champaign

Wed, July 13 2021

Reminders

Reminders

- Homework 5 due Friday
- Homework 4 grace period due Friday
- Signup for Quiz 4 via the CBTF

Strings

Poll Question: Slicing

What is the result of running this code?

```
1 my_str = "CS 105"
2 print(my_str[1:2])
```

A. 'C'
B. 'CS'
C. 'CS '
D. 'S'
E. 'S '

Poll Question: Slicing

What is the result of running this code?

```
1 my_str = "CS 105"
2 print(my_str[-4:-2])
```

A. 'S 1'
B. 'S 10'
C. ' 1'
D. ' 10'

Poll Question: Slicing

What is the result of running this code?

```
1 my_str = "CS 105"
2 print(my_str([::2]))
```

A. 'C'

B. 'CS'

C. 'S'

D. 'C 0'

Poll Question: Splitting

What is the result of running this code?

```
1 my_str = "CS 105 rox"
2 result = my_str.split()
```

A. `("CS", "105", "rox")`

B. `["CS 105 rox""]`

C. `["CS", "105 rox"]`

D. `["CS", "105", "rox"]`

Poll Question: Splitting

What is the result of running this code?

```
1 csv = "1, 2, 3, 4"
2 result = csv.split(",")
```

A. ['1']

B. ['1, 2, 3, 4']

C. ['1', '2', '3', '4']

D. ['1,', '2,', '3,', '4']

Poll Question: Joining

What is the result of running this code?

```
1 numlist = [1, 2, 3, 4]
2 result = ",".join(numlist)
```

A. '1234'

B. '1,2,3,4'

C. '1, 2, 3, 4

D. TypeError

A Common Pattern

The generic pattern:

```
mylist = input_data.split(<separator>)
... data processing ...
outputstring "<separator>".join(my_list)
```

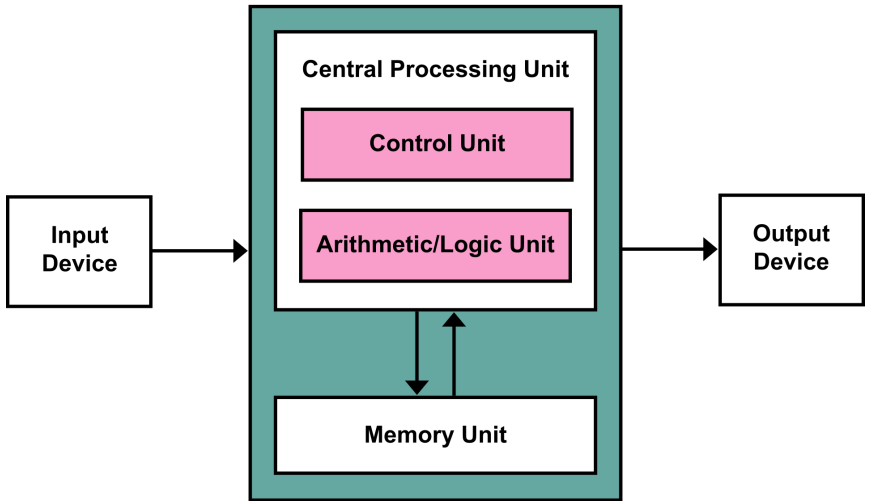
An example of this being done on one line:

```
output = ",".join(input.split(",")[::2])
```

Files

Why do we use files?

- Files are how data are stored on external storage
- Managed by O/S
- Disk is slow
- Opening files moves them to memory for the program
- Data is buffered (temporarily stored) in memory



Poll Question: Opening a File

By default what does this code allow?

A. `Reading`
B. `Writing`
C. `Appending`
D. `Reading and Writing`
E. `Writing and Appending`
F. `reading, Writing, and Appending`

Poll Question: Files

Continue writing to existing files?

A. `outf = open('filename', 'r')`

B. `outf = open('filename', 'x')`

C. `outf = open('filename', 'i')`

D. `outf = open('filename', 'a')`

E. `outf = open('filename', 'e')`

Reading from Files

Method 1:

```
file_object = open('filename')
lines = file_object.readlines()
for line in lines:
  print(line)
file_object.close()
```

Method 2:

```
with open('filename') as inf:
  lines = inf.readlines()
  for line in lines:
    print(line)
  #automatic file close
```

Writing to Files

Method 1:

```
file_object = open('filename', 'w')
file_object.write('thing to write')
file_objet.close()  #automatic at program end
file_object.flush() #optional
```

Method 2:

```
with open('filename', 'w') as outf:
  outf.write('thing to write')
  #automatic file close
```