BINF2111 – Introduction to Bioinformatics Computing

Python – slithering in your terminal

Richard Allen White III, PhD
RAW Lab

Lecture 17 – Tuesday Nov 2nd, 2021

Learning Objectives

- Review Quiz 16

- Reading files

- Writing files

- Striping

- Loops (review, for and while)

- Quiz 17

Quiz 16 answers

protein = 'VVKKKWWW'

This command print(protein[1])

Would provide me the first amino acid residue?

T or F

Quiz 16 answers

protein = 'VVKKKWWW'

This command print(protein[1])

Would provide me the first amino acid residue?

T or F

command print(protein[0])

ZERO point! It starts with ZERO.

Quiz 16 answers

name = 'RAW\nLab'

Which command would split into RAW and Lab?

print(name.split("\t')[0])

print(name.split(''\t")[0])

print(name.split('\t")[0])

None will work

Quiz 16 answers

name = 'RAW\nLab'

Which command would split into RAW and Lab?

print(name.split("\t')[0]) - 1

print(name.split(''\t")[0]) - 3

print(name.split('\t")[0]) - 0

None will work – 5

print(name.split(“\n”)[0]) = print(name) → T or F

Quiz 16 answers

name = 'RAW\nLab'

Which command would split into RAW and Lab?

print(name.split(“\n”)[0]) = print(name) → T or F

RAW

LAB

How do you split?

Quiz 16 answers

name = 'RAW\nLab'

Which command would split into RAW and Lab?

print(name.split("\n")[0]) = print(name) → T or F

RAW

LAB

How do you split?

print(name.split()) or print(name.replace('\n', ' '))

Opening files

Variables – variables allow you to store values within data types

Data types:

Numbers

Strings

List (Grocery List, can change)

Tuple (Fixed List Like Months, Jan… Feb… etc.)

Dictionary (Indexed List with Key>Value)

File types common in bioinfomatics

Text: .fna, .ffn, .faa, .txt, .csv, .tsv, .py, .sh, .sam

- May or not be human readable

Binary files: .bam

Other: Images, audio, video, compressed files.

Opening files

Open Function – Function returns (brings into existence for future use) a file object

Usage : open(“filename.txt”)

The argument is the name of the file as a string.
You can assign the file object to a variable.

ex: my_file = open(“filename.txt”)

Accessing files

Open Function – Function returns (brings into existence for future use) a file object

file objects are not used directly like variables

they are usually accessed via file methods

```
my_file = open(“example2.fasta”)
file_contents = my_file.read()
print(file_contents)
```

The read method is used to extract file contents into variable

Accessing files

hello.txt: contains ‘hello you.’

example2.fasta: example fasta nucleotide which you should rename to .fna (with bash!)

KO_nifH.faa: contains several protein sequences in real FASTA sequences from NifH protein

MultiN.fastq: The bad fastq with many N’s

Describing files

hello_file = (‘hello.txt’) ← (File name)

hello_id = open(hello_file) ← (File object)

hello_file_contents = hello_id.read() ← (File method)

Accessing files

example2_file = (‘example2.fasta’)

example2_file is a file object that represents the file on the computer hard drive. It does not represent the text

In order to store the text into a variable, use the .read() method.

```
example2_file =
open('example2.fasta')

example2seq = example2_file.read()
print(example2seq)
```

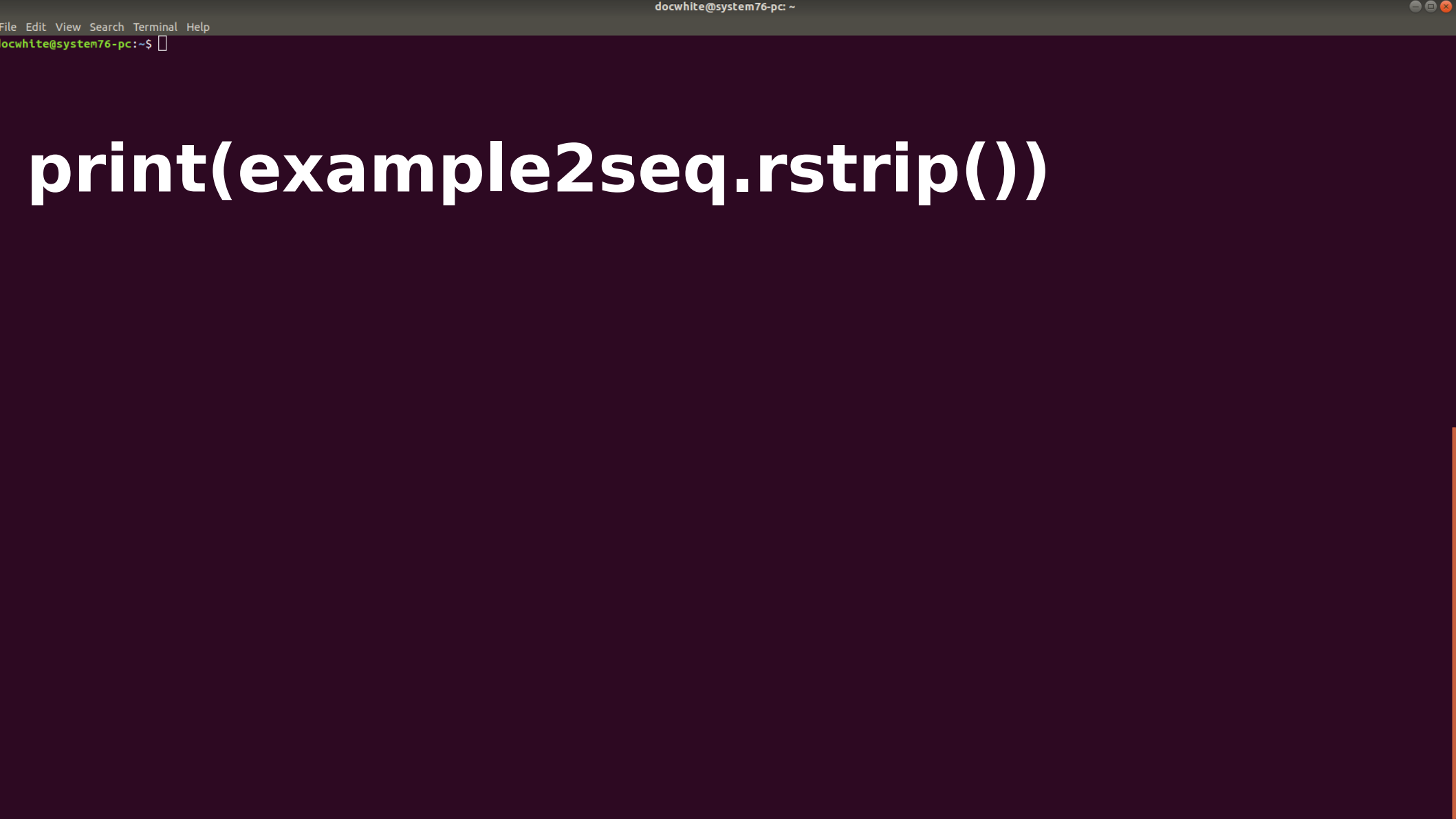
Strip Whitespace

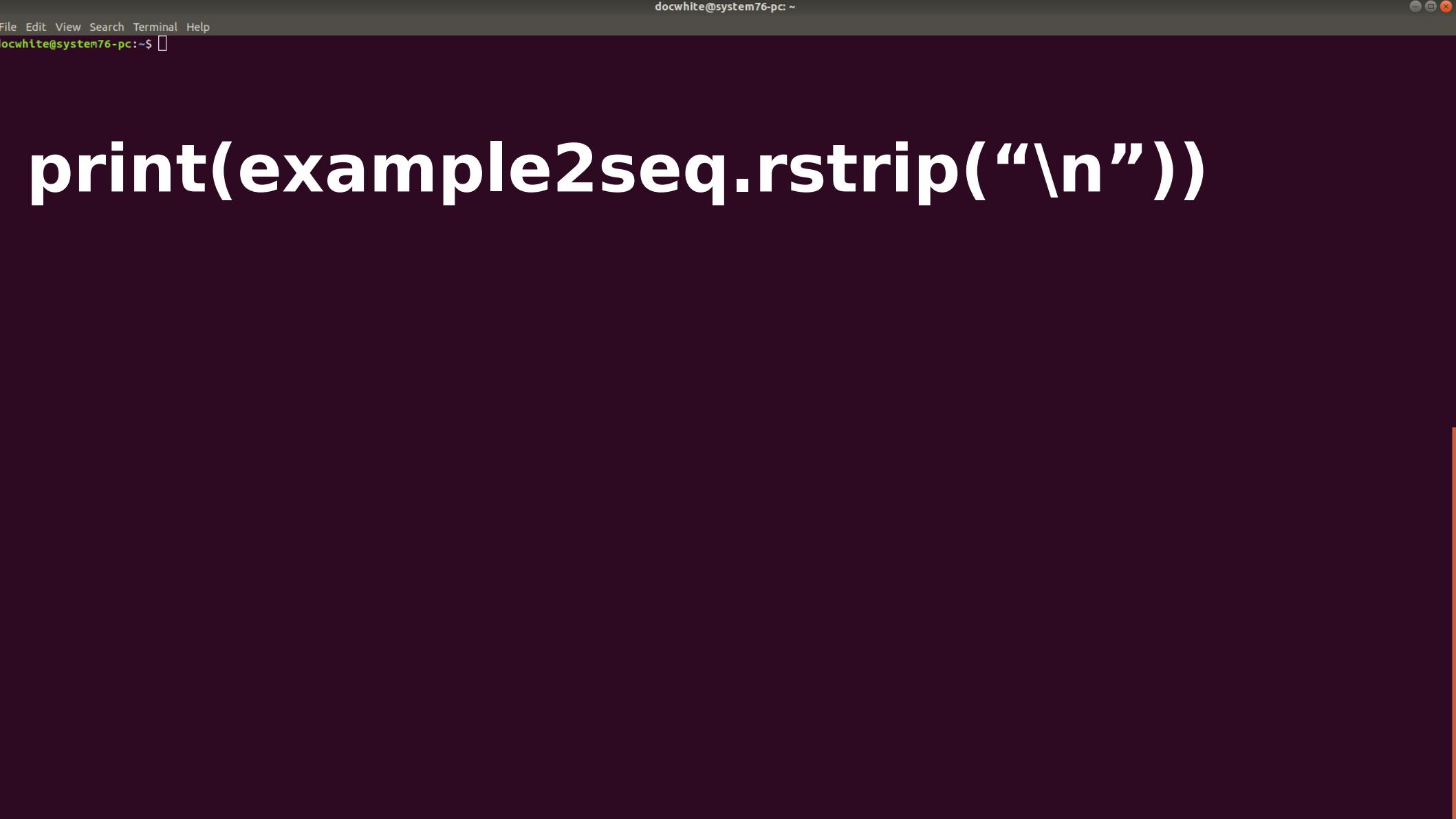
Opening example2.fasta:

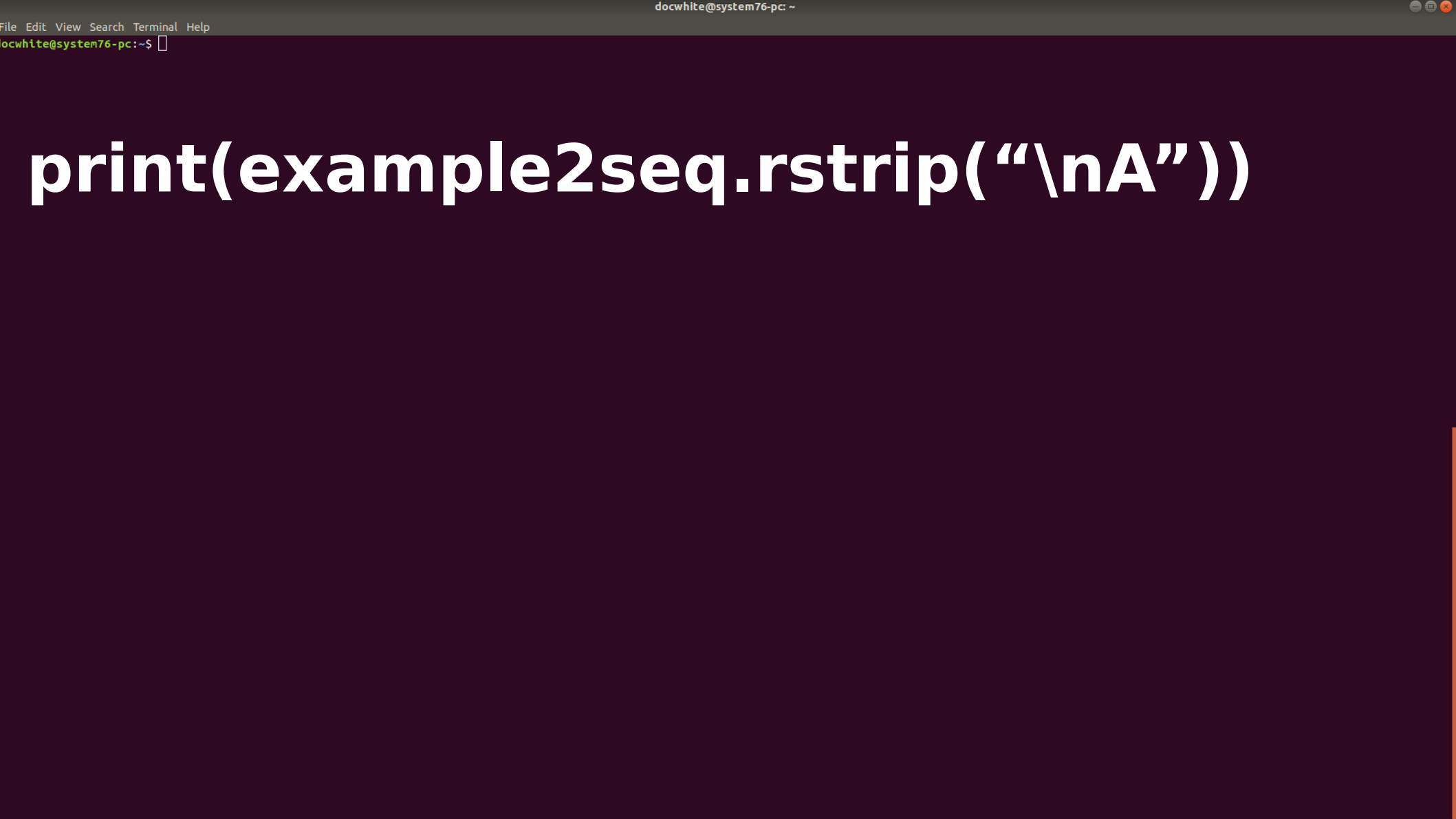
example2_file now contains the text contents of example2.fasta

However, in most text files, there is a large amount of whitespace characters, tabs, newlines etc.

In order to clean up the text and allow for easy editing of characters, use the rstrip() method.


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
print(example2seq.rstrip())


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
print(example2seq.rstrip("\n"))


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
print(example2seq.rstrip("\nA"))

Readline( )

Opening example2.fasta using readlines():

If you want to read in the entire file without separation as a single line, use read().

If you want to read in the file line by line into a list use readlines().

```
example2_file = open('example2.fasta')
example2_line = example2_file.readlines()
print(example2_line)

example2_line.close() #close your file
```

Writing to files

Opening hello.txt for writing:

```
output_file = open('hello.txt', 'w')
output_file.write("any string")
```

Note that you open the file the same way, but designate a second argument 'w', which says that you are writing to the file.

The write() method allows you to write to the file. You can write saved variables, raw input, etc.

Writing to files

Opening hello.txt for writing:

```
output_file = open('hello.txt', 'w')
output_file.write("any string")
```

The file method "open" takes two arguments – the filename, and "r", "w", or "a" depending on what you want to do with the file.

"r" is the default, and if you don't give an argument you are opening the file read-only by default.

"w" = write

"a" = append (add new contents to an existing file)

Writing to files

Opening hello.txt for writing:

```
output_file = open('hello.txt', 'w')
output_file.write("any string")

output_file.write(">" + description)
output_file.write(DNASeq)
output_file.write(">" + description + '\n' + DNASeq)

output_file.close()
```

The third example writes two lines worth of information at once with a newline '\n' in between

With statements

Another better way to open the file – the with statement

```
with open('output.txt', 'w') as f:
...      f.write('Hi there!')

quit()
```

With statements

Another better way to open the file – the with statement

Statements are commands to do something. Similar to functions, but functions can be called elsewhere.

The with command allows you to execute two related operations (In this case open output.txt for writing, and assign to the variable f).

With statements

Another better way to open the file – the with statement

Statements are commands to do something. Similar to functions, but functions can be called elsewhere.

The with command allows you to execute two related operations (In this case open output.txt for writing, and assign to the variable f).

This has the advantage that the file is properly closed after finishes, even if an exception is raised on the way. It is also much shorter.

BASH - for loop

```
for i in file.*;do
  command $i
done
```

BASH - for loop (C-style)

```
for ((i = 0 ; i < 100 ; i++)); do
  command $i
done
```

BASH - for loop (Java)

```
for (i = 0 ; i < 100 ; i++){
    command (i);
}
```

BASH - while loop

```
#!/bin/bash
x=1
while [ $x -le 5 ]
do
  echo "Welcome $x
times"
  x=$(( $x + 1 ))
done
```

BASH - while loop (one - liner)

```
x=1; while [ $x -le 5 ]; do echo "Welcome $x times" $(( x++ )); done
```

For loop (Python)

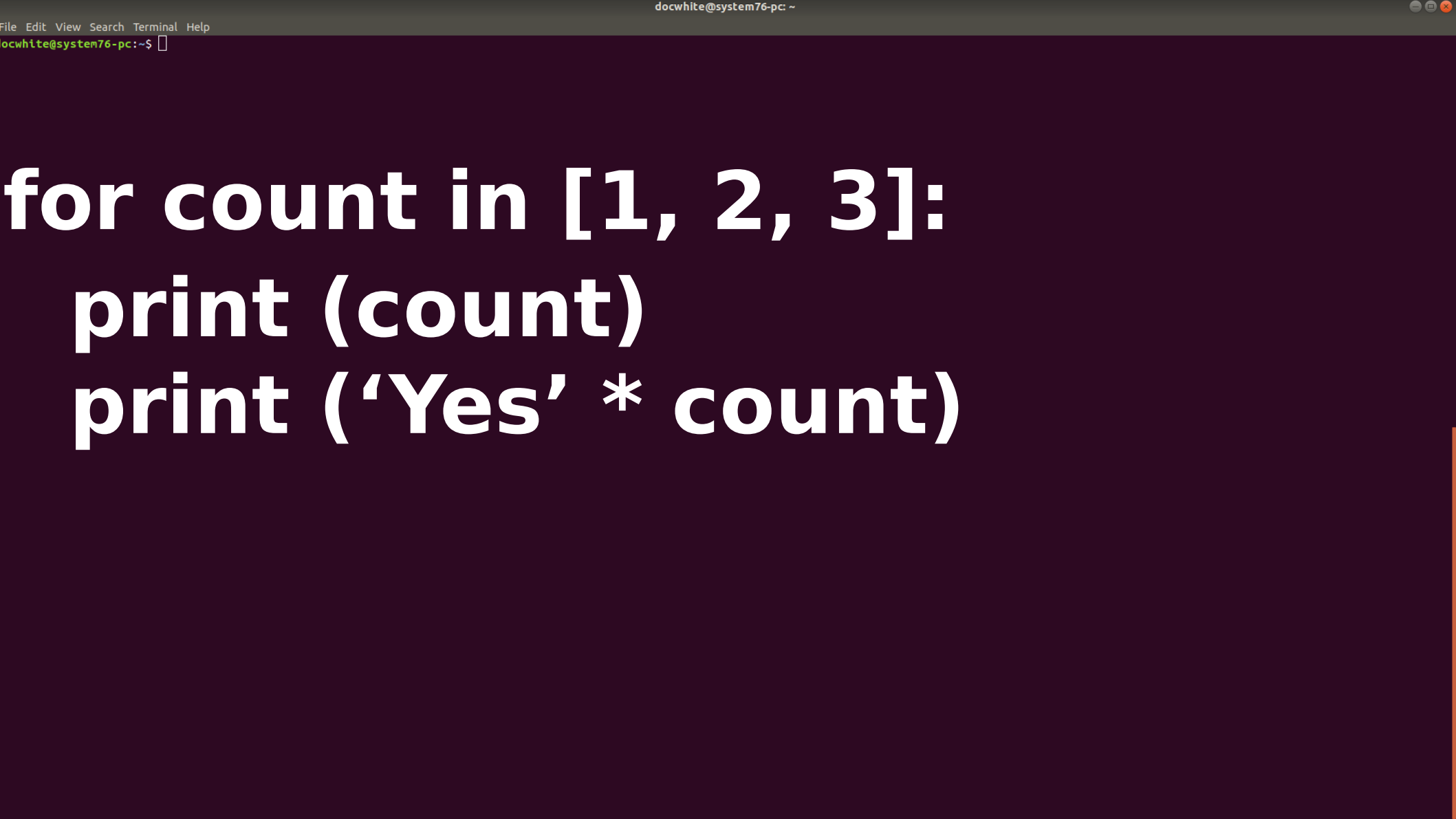
```
for x in file:
    command (x)
```

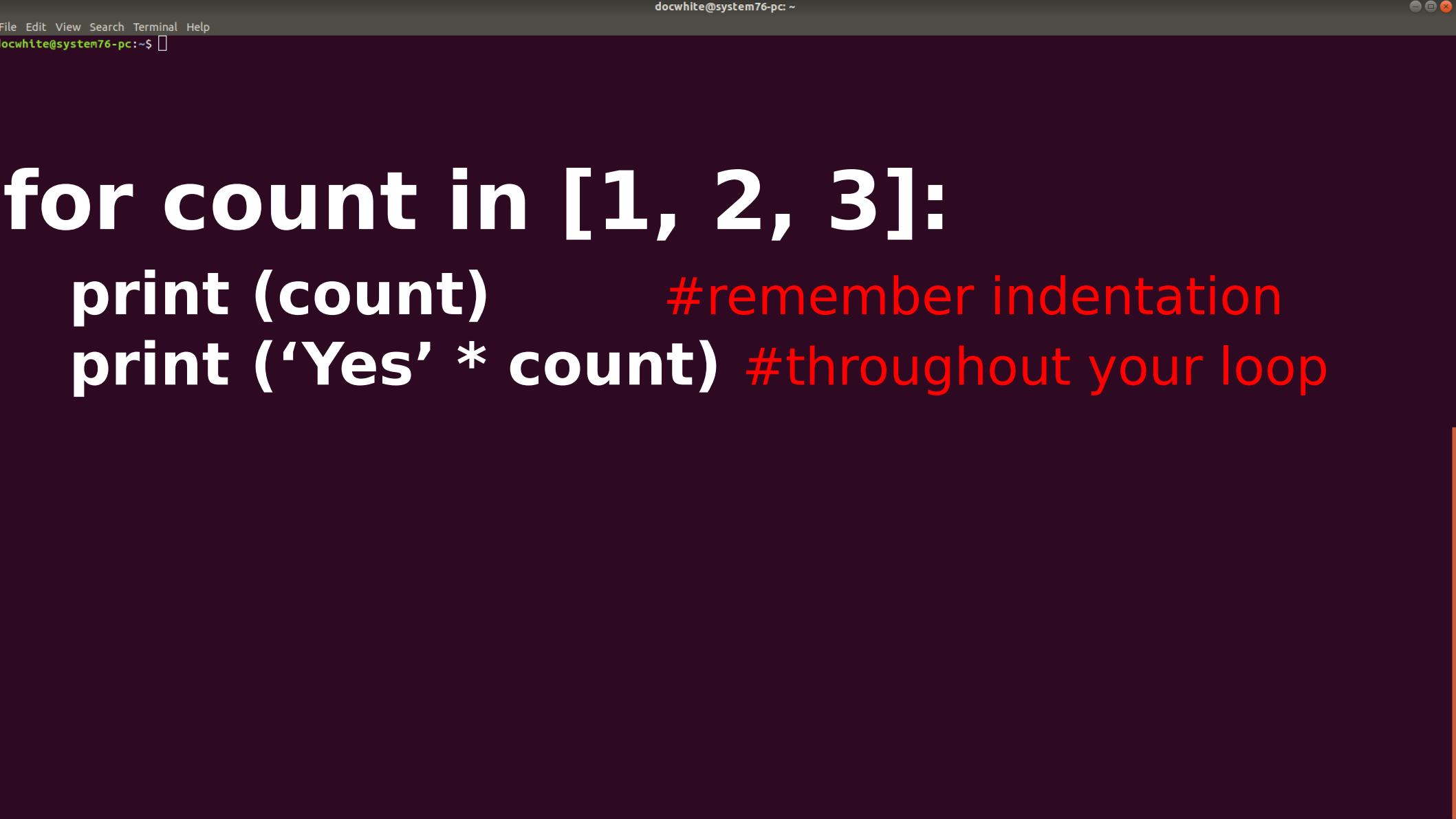
For loop (Python)

```
for x in file:
    command (x)
```

A for loop starts with the heading for, followed by the variable, the word in, some sequence, and a colon.

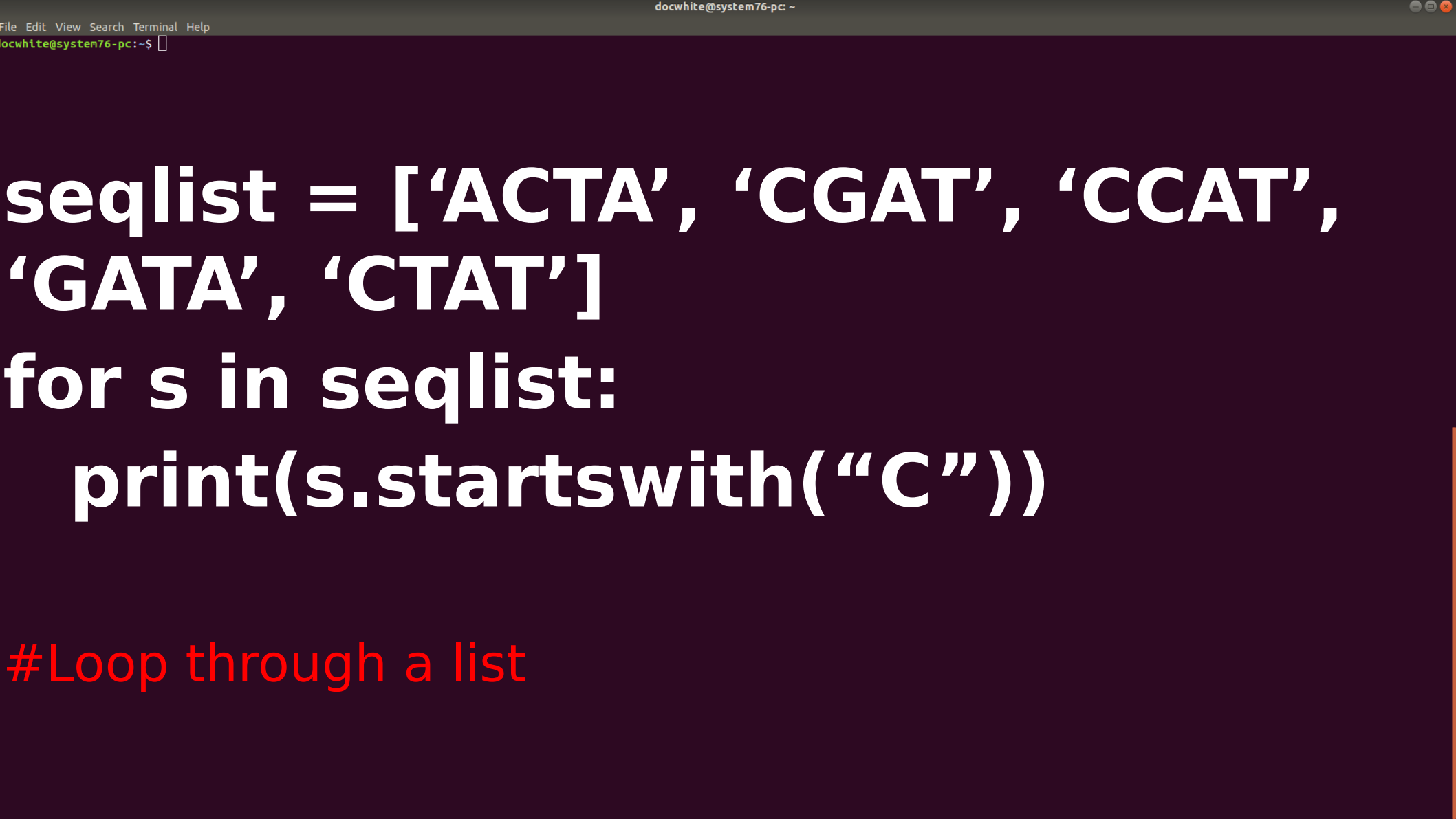
for item in sequence:
do something with the current item


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
for count in [1, 2, 3]:
print (count)
print ('Yes' * count)


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
for count in [1, 2, 3]:
print (count) #remember indentation
print ('Yes' * count) #throughout your loop

```
Numlist = [1, 2, 3, 5, 6000]
for n in numlist:
    n = n+100
    print(n)


#Loop through a list
```


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
seqlist = [‘ACTA’, ‘CGAT’, ‘CCAT’, ‘GATA’, ‘CTAT’]
for s in seqlist:
print(s.startswith(“C”))
#Loop through a list

While loop (Python)

While loops

Sometimes you want to execute a block of code until you reach some result.

A while loop starts with the heading while, followed by the condition, a colon, and your execution.

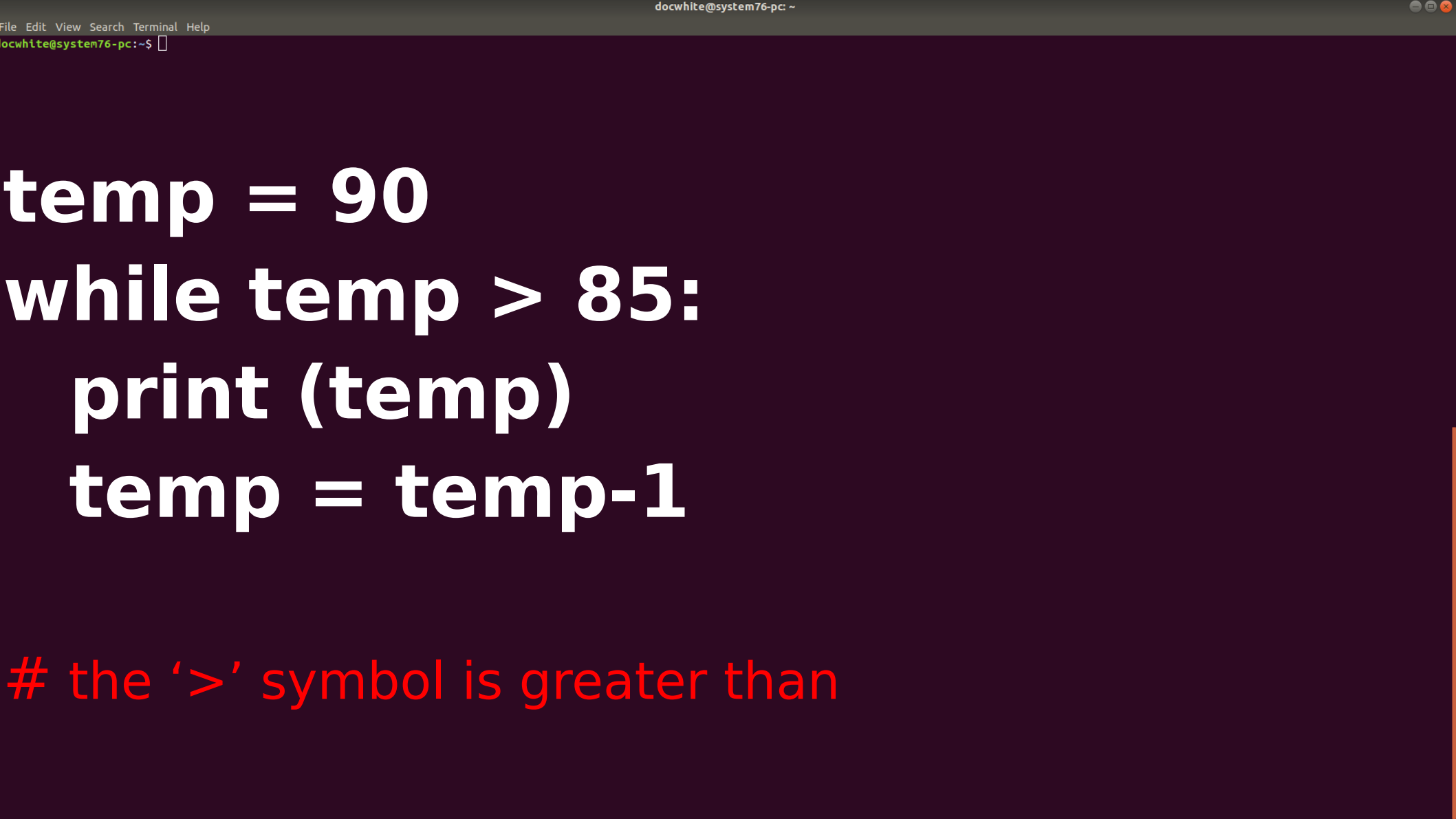
While loop (Python)

While loops

A while loop starts with the heading while, followed by the condition, a colon, and your execution.

while *condition*:

    indented block


docwhite@system76-pc: ~
File Edit View Search Terminal Help
docwhite@system76-pc:~$
temp = 90
while temp > 85:
print (temp)
temp = temp-1
# the ‘>’ symbol is greater than

While loop (Python)

While loops

```
>>> temp = 90
>>> while temp > 85:
...     print (temp)          # here we print before subtraction
...     temp = temp -1
...
90              # temp starts at 90 is greater than 85 (loop 1)
89              # temp is now 89 is greater than 85 (loop 2)
88              # temp is now 88 is greater than 85 (loop 3)
87              # temp is now 87 is greater than 85 (loop 4)
86              # temp is now 86 is greater than 85 (loop 5)
>>> temp
85              # the last line of the loop is the subtraction
>>>
```

While loop (Python)

While loops

```
>>> temp = 90
>>> while temp > 85:
...     temp = temp -1
...     print (temp) # here we print before subtraction
...
89
88
87
86
85
>>> temp                          # your variable has changed inside loop!
85
>>>
```

Quiz 17

- On canvas now