

Outline

MCS 260 L-22

17 October 2007

Files

- gzip and gunzip
- data compression

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Summary + Assignments

MCS 260 Lecture 22
Introduction to Computer Science
Jan Vershelde, 17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing words

Summary + Assignments

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Compressing Files

gzip or gunzip

File compression saves space on mass storage.

A session at the Linux command prompt \$:

```
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
$ gzip files.pdf
$ ls -g files.pdf
ls: files.pdf: No such file or directory
$ ls -g files.pdf.gz
-rwxr-xr-x  1 30003 236452 Oct 17 07:16 files.pdf.gz
```

Gzip reduces size of file from 487634 to 236452 bytes.

```
$ gunzip files.pdf.gz
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
```

Gzip uses the Lempel-Ziv encoding (LZ77).

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Compressing Files

gzip or gunzip

File compression saves space on mass storage.

A session at the Linux command prompt \$:

```
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
$ gzip files.pdf
$ ls -g files.pdf
ls: files.pdf: No such file or directory
$ ls -g files.pdf.gz
-rwxr-xr-x  1 30003 236452 Oct 17 07:16 files.pdf.gz
```

Gzip reduces size of file from 487634 to 236452 bytes.

```
$ gunzip files.pdf.gz
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
```

Gzip uses the Lempel-Ziv encoding (LZ77).

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Compressing Files

gzip or gunzip

File compression saves space on mass storage.

A session at the Linux command prompt \$:

```
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
$ gzip files.pdf
$ ls -g files.pdf
ls: files.pdf: No such file or directory
$ ls -g files.pdf.gz
-rwxr-xr-x  1 30003 236452 Oct 17 07:16 files.pdf.gz
```

Gzip reduces size of file from 487634 to 236452 bytes.

```
$ gunzip files.pdf.gz
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
```

Gzip uses the Lempel-Ziv encoding (LZ77).

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Compressing Files

gzip or gunzip

File compression saves space on mass storage.

A session at the Linux command prompt \$:

```
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
$ gzip files.pdf
$ ls -g files.pdf
ls: files.pdf: No such file or directory
$ ls -g files.pdf.gz
-rwxr-xr-x  1 30003 236452 Oct 17 07:16 files.pdf.gz
```

Gzip reduces size of file from 487634 to 236452 bytes.

```
$ gunzip files.pdf.gz
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
```

Gzip uses the Lempel-Ziv encoding (LZ77).

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Compressing Files

gzip or gunzip

File compression saves space on mass storage.

A session at the Linux command prompt \$:

```
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
$ gzip files.pdf
$ ls -g files.pdf
ls: files.pdf: No such file or directory
$ ls -g files.pdf.gz
-rwxr-xr-x  1 30003 236452 Oct 17 07:16 files.pdf.gz
```

Gzip reduces size of file from 487634 to 236452 bytes.

```
$ gunzip files.pdf.gz
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
```

Gzip uses the Lempel-Ziv encoding (LZ77).

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary +
Assignments

Compressing Files

gzip or gunzip

File compression saves space on mass storage.

A session at the Linux command prompt \$:

```
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
$ gzip files.pdf
$ ls -g files.pdf
ls: files.pdf: No such file or directory
$ ls -g files.pdf.gz
-rwxr-xr-x  1 30003 236452 Oct 17 07:16 files.pdf.gz
```

Gzip reduces size of file from 487634 to 236452 bytes.

```
$ gunzip files.pdf.gz
$ ls -g files.pdf
-rwxr-xr-x  1 30003 487634 Oct 17 07:16 files.pdf
```

Gzip uses the Lempel-Ziv encoding (LZ77).

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary +
Assignments

Outline

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing words

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Summary + Assignments

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Two different data compression schemes:

- ▶ **lossless:** no loss of information in compression
- ▶ **lossy:** minor errors are tolerated (e.g. images)

Popular compression schemes use dictionary encoding.
For example: replace all the's in a text file by a symbol.

The Lempel-Ziv encoding used in Gzip finds duplicated strings in the input data. The second occurrence of a string is replaced by a pointer to the previous string.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Two different data compression schemes:

- ▶ lossless: no loss of information in compression
- ▶ lossy: minor errors are tolerated (e.g. images)

Popular compression schemes use dictionary encoding.
For example: replace all the's in a text file by a symbol.

The Lempel-Ziv encoding used in Gzip finds duplicated strings in the input data. The second occurrence of a string is replaced by a pointer to the previous string.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Two different data compression schemes:

- ▶ lossless: no loss of information in compression
- ▶ lossy: minor errors are tolerated (e.g. images)

Popular compression schemes use dictionary encoding.
For example: replace all the's in a text file by a symbol.

The Lempel-Ziv encoding used in Gzip finds duplicated strings in the input data. The second occurrence of a string is replaced by a pointer to the previous string.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Two different data compression schemes:

- ▶ lossless: no loss of information in compression
- ▶ lossy: minor errors are tolerated (e.g. images)

Popular compression schemes use dictionary encoding.
For example: replace all `the`'s in a text file by a symbol.

The Lempel-Ziv encoding used in Gzip finds duplicated strings in the input data. The second occurrence of a string is replaced by a pointer to the previous string.

Files

gzip and gunzip
data compression

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing words

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Summary + Assignments

Format Conversions

problem statement

Convert the ad hoc formatting of the file `books`

```
1:1:The Art & Craft of Computing:
0:2:Making Use of Python:
```

into one that uses lists

```
[True, 1, 'The Art & Craft of Computing']
[False, 2, 'Making Use of Python']
```

and into using dictionaries

```
{'available': True, 'key': 1, \
 'title': 'The Art & Craft of Computing'}
{'available': False, 'key': 2, \
 'title': 'Making Use of Python'}
```

Benefit: the module `bkform` is no longer needed.

Files

- gzip and gunzip
- data compression

Python

- format conversions**
- encrypting text
- counting and replacing words

Summary + Assignments

Format Conversions

problem statement

Convert the ad hoc formatting of the file `books`

```
1:1:The Art & Craft of Computing:
0:2:Making Use of Python:
```

into one that uses lists

```
[True, 1, 'The Art & Craft of Computing']
[False, 2, 'Making Use of Python']
```

and into using dictionaries

```
{'available': True, 'key': 1, \
 'title': 'The Art & Craft of Computing'}
{'available': False, 'key': 2, \
 'title': 'Making Use of Python'}
```

Benefit: the module `bkform` is no longer needed.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Format Conversions

problem statement

Convert the ad hoc formatting of the file `books`

```
1:1:The Art & Craft of Computing:
0:2:Making Use of Python:
```

into one that uses lists

```
[True, 1, 'The Art & Craft of Computing']
[False, 2, 'Making Use of Python']
```

and into using dictionaries

```
{'available': True, 'key': 1, \
 'title': 'The Art & Craft of Computing'}
{'available': False, 'key': 2, \
 'title': 'Making Use of Python'}
```

Benefit: the module `bkform` is no longer needed.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Format Conversions

problem statement

Convert the ad hoc formatting of the file `books`

```
1:1:The Art & Craft of Computing:
0:2:Making Use of Python:
```

into one that uses lists

```
[True, 1, 'The Art & Craft of Computing']
[False, 2, 'Making Use of Python']
```

and into using dictionaries

```
{'available': True, 'key': 1, \
 'title': 'The Art & Craft of Computing'}
{'available': False, 'key': 2, \
 'title': 'Making Use of Python'}
```

Benefit: the module `bkform` is no longer needed.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Format Conversion Algorithm

Files

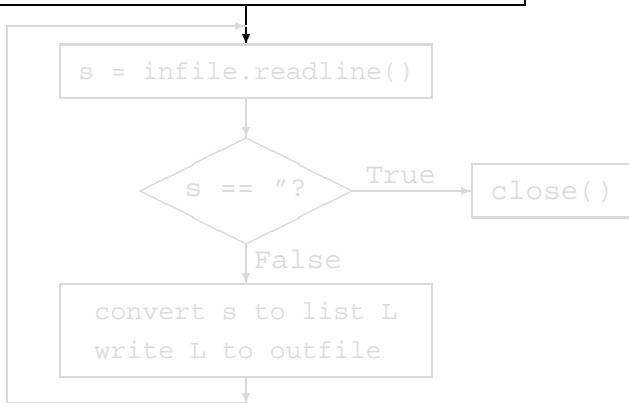
gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

```
infile = open('books', 'r')  
outfile = open('bookslists', 'w')
```



The Format Conversion Algorithm

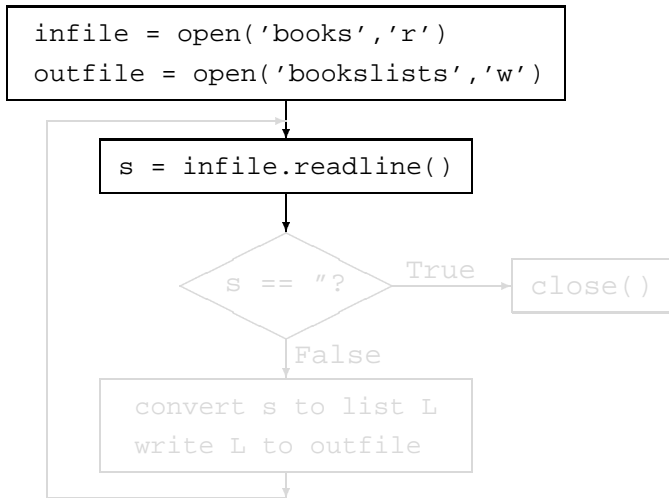
Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments



The Format Conversion Algorithm

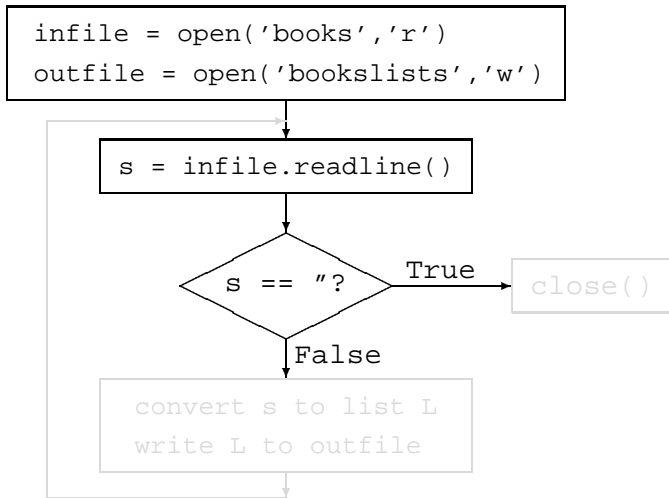
Files

- gzip and gunzip
- data compression

Python

- format conversions**
- encrypting text
- counting and replacing words

Summary + Assignments



The Format Conversion Algorithm

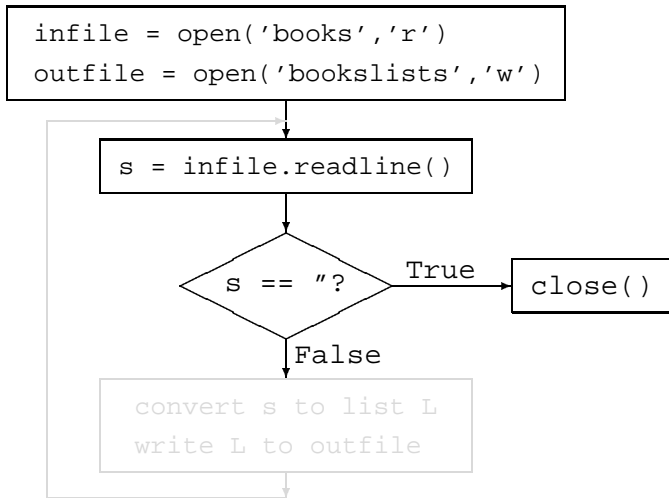
Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments



The Format Conversion Algorithm

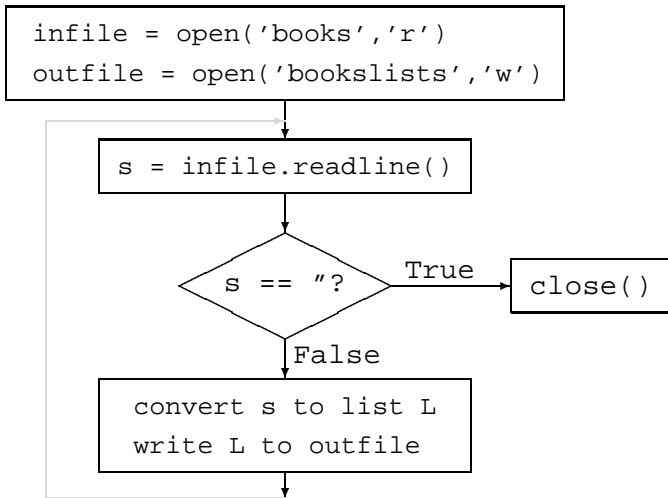
Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments



The Format Conversion Algorithm

MCS 260 L-22

17 October 2007

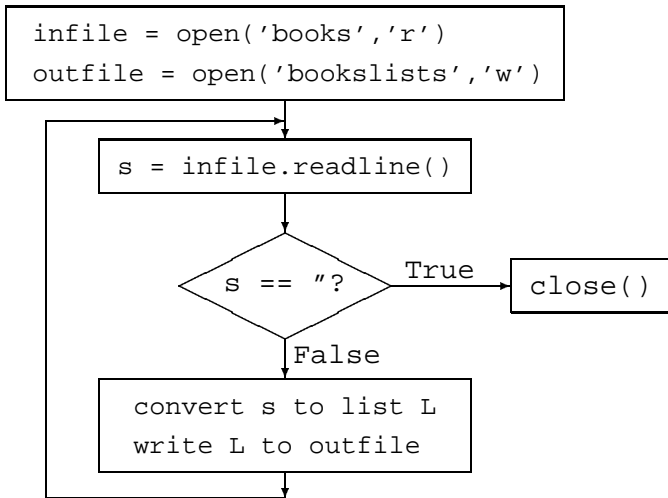
Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments



Python program bkform2list.py

MCS 260 L-22

17 October 2007

```
infile = open('books','r')
outfile = open('bookslists','w')
while True:
    s = infile.readline()
    if s == '': break
    L = s.split(':')
    t = [L[0] == '1',int(L[1]), L[2]]
    outfile.write(str(t) + '\n')
infile.close()
outfile.close()
```

Observe: `L[0] == '1'` returns bool for availability
`int(L[1])` returns integer for key

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bkform2list.py

MCS 260 L-22

17 October 2007

```
infile = open('books','r')
outfile = open('bookslists','w')
while True:
    s = infile.readline()
    if s == '': break
    L = s.split(':')
    t = [L[0] == '1',int(L[1]), L[2]]
    outfile.write(str(t) + '\n')
infile.close()
outfile.close()
```

Observe: `L[0] == '1'` returns bool for availability
`int(L[1])` returns integer for key

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bkform2list.py

MCS 260 L-22

17 October 2007

```
infile = open('books', 'r')
outfile = open('bookslists', 'w')
while True:
    s = infile.readline()
    if s == '': break
    L = s.split(':')
    t = [L[0] == '1', int(L[1]), L[2]]
    outfile.write(str(t) + '\n')
infile.close()
outfile.close()
```

Observe: `L[0] == '1'` returns bool for availability
`int(L[1])` returns integer for key

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bkform2list.py

MCS 260 L-22

17 October 2007

```
infile = open('books','r')
outfile = open('bookslists','w')
while True:
    s = infile.readline()
    if s == '': break
    L = s.split(':')
    t = [L[0] == '1',int(L[1]), L[2]]
    outfile.write(str(t) + '\n')
infile.close()
outfile.close()
```

Observe: `L[0] == '1'` returns bool for availability
`int(L[1])` returns integer for key

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bkform2list.py

MCS 260 L-22

17 October 2007

```
infile = open('books','r')
outfile = open('bookslists','w')
while True:
    s = infile.readline()
    if s == '': break
    L = s.split(':')
    t = [L[0] == '1',int(L[1]), L[2]]
    outfile.write(str(t) + '\n')
infile.close()
outfile.close()
```

Observe: `L[0] == '1'` returns bool for availability
`int(L[1])` returns integer for key

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bkform2list.py

MCS 260 L-22

17 October 2007

```
infile = open('books','r')
outfile = open('bookslists','w')
while True:
    s = infile.readline()
    if s == '': break
    L = s.split(':')
    t = [L[0] == '1',int(L[1]), L[2]]
    outfile.write(str(t) + '\n')
infile.close()
outfile.close()
```

Observe: `L[0] == '1'` returns bool for availability
`int(L[1])` returns integer for key

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Converting Lists into Dictionaries

Files

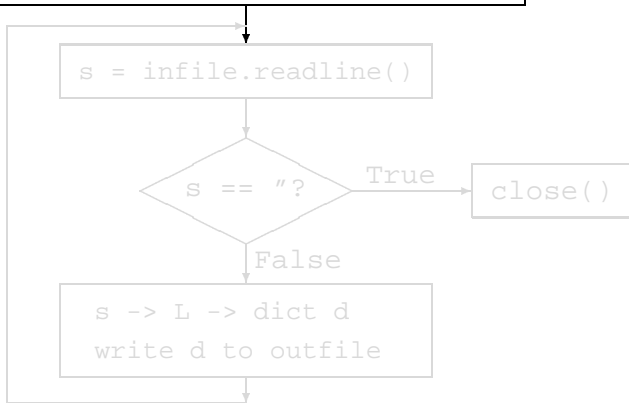
- gzip and gunzip
- data compression

Python

- format conversions**
- encrypting text
- counting and replacing words

Summary + Assignments

```
infile = open('bookslists','r')
outfile = open('booksdicts','w')
```



Converting Lists into Dictionaries

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

```
infile = open('bookslists','r')  
outfile = open('booksdicts','w')
```

```
s = infile.readline()
```

```
s == "?"
```

True

```
close()
```

False

```
s -> L -> dict d  
write d to outfile
```

Converting Lists into Dictionaries

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

```
infile = open('bookslists','r')  
outfile = open('booksdicts','w')
```

```
s = infile.readline()
```

```
s == "?"
```

True

```
close()
```

False

```
s -> L -> dict d  
write d to outfile
```

Converting Lists into Dictionaries

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

```
infile = open('bookslists','r')  
outfile = open('booksdicts','w')
```

```
s = infile.readline()
```

```
s == "?"
```

True

```
close()
```

False

```
s -> L -> dict d  
write d to outfile
```

Converting Lists into Dictionaries

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

```
infile = open('bookslists','r')
outfile = open('booksdicts','w')
```

```
s = infile.readline()
```

```
s == "?"
```

True

```
close()
```

False

```
s -> L -> dict d
write d to outfile
```

Converting Lists into Dictionaries

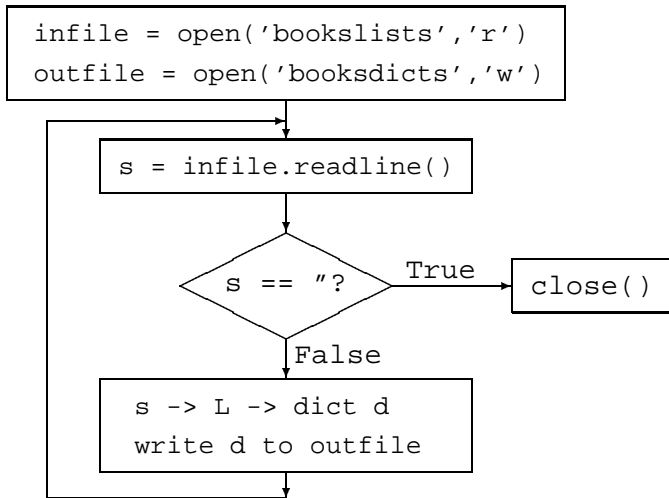
Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments



Python program bklist2dict.py

```
infile = open('bookslists','r')
outfile = open('booksdicts','w')
while True:
    s = infile.readline()
    if s == '': break
    L = eval(s)
    d = {'available':L[0], 'key':L[1], \
        'title':L[2] }
    outfile.write(str(d) + '\n')
infile.close()
outfile.close()
```

Observe: `eval(str(L)) == L`
`str(L)` converts a list to a string
`eval()` evaluates a string into an object

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bclist2dict.py

```
infile = open('bookslists', 'r')
outfile = open('booksdicts', 'w')
while True:
    s = infile.readline()
    if s == '': break
    L = eval(s)
    d = {'available':L[0], 'key':L[1], \
        'title':L[2] }
    outfile.write(str(d) + '\n')
infile.close()
outfile.close()
```

Observe: `eval(str(L)) == L`
`str(L)` converts a list to a string
`eval()` evaluates a string into an object

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bklist2dict.py

```

infile = open('bookslists','r')
outfile = open('booksdicts','w')
while True:
    s = infile.readline()
    if s == '': break
    L = eval(s)
    d = {'available':L[0], 'key':L[1], \
        'title':L[2] }
    outfile.write(str(d) + '\n')
infile.close()
outfile.close()

```

Observe: `eval(str(L)) == L`
`str(L)` converts a list to a string
`eval()` evaluates a string into an object

Files

gzip and gunzip
 data compression

Python

format conversions
 encrypting text
 counting and replacing
 words

Summary + Assignments

Python program bklist2dict.py

```
infile = open('bookslists', 'r')
outfile = open('booksdicts', 'w')
while True:
    s = infile.readline()
    if s == '': break
    L = eval(s)
    d = {'available':L[0], 'key':L[1], \
        'title':L[2] }
    outfile.write(str(d) + '\n')
infile.close()
outfile.close()
```

Observe: `eval(str(L)) == L`
`str(L)` converts a list to a string
`eval()` evaluates a string into an object

Files

- gzip and gunzip
- data compression

Python

- format conversions**
- encrypting text
- counting and replacing words

Summary + Assignments

Python program bklist2dict.py

```
infile = open('bookslists','r')
outfile = open('booksdicts','w')
while True:
    s = infile.readline()
    if s == '': break
    L = eval(s)
    d = {'available':L[0], 'key':L[1], \
        'title':L[2] }
    outfile.write(str(d) + '\n')
infile.close()
outfile.close()
```

Observe: `eval(str(L)) == L`
`str(L)` converts a list to a string
`eval()` evaluates a string into an object

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Python program bklist2dict.py

```
infile = open('bookslists', 'r')
outfile = open('booksdicts', 'w')
while True:
    s = infile.readline()
    if s == '': break
    L = eval(s)
    d = {'available':L[0], 'key':L[1], \
        'title':L[2] }
    outfile.write(str(d) + '\n')
infile.close()
outfile.close()
```

Observe: `eval(str(L)) == L`
`str(L)` converts a list to a string
`eval()` evaluates a string into an object

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Outline

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing words

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Summary + Assignments

Scrambling Text

encrypting information

MCS 260 L-22

17 October 2007

The content of the file `sometext`:

This is a sample text, used as an example
for a message whose vowels will be scrambled.

After scrambling vowels, we obtain `codetext`:

```
Thos os e semplu tuxt, isud es en uxemplu  
far e mussegu whasu vawuls woll bu scremblud.
```

The cipher used:

```
['a','i','e','u','o'] -> ['e','o','u','i','a']
```

Python: `file.read(1)` returns one byte read from file.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Scrambling Text

encrypting information

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The content of the file `sometext`:

This is a sample text, used as an example
for a message whose vowels will be scrambled.

After scrambling vowels, we obtain `codetext`:

Thos os e semplu tuxt, isud es en uxemplu
far e mussegu whasu vawuls woll bu scremblud.

The cipher used:

```
['a','i','e','u','o'] -> ['e','o','u','i','a']
```

Python: `file.read(1)` returns one byte read from file.

Scrambling Text

encrypting information

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The content of the file `sometext`:

This is a sample text, used as an example
for a message whose vowels will be scrambled.

After scrambling vowels, we obtain `codetext`:

Thos os e semplu tuxt, isud es en uxemplu
far e mussegu whasu vawuls woll bu scremblud.

The cipher used:

```
['a','i','e','u','o'] -> ['e','o','u','i','a']
```

Python: `file.read(1)` returns one byte read from file.

Scrambling Text

encrypting information

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The content of the file `sometext`:

This is a sample text, used as an example
for a message whose vowels will be scrambled.

After scrambling vowels, we obtain `codetext`:

```
Thos os e semplu tuxt, isud es en uxemplu  
far e mussegu whasu vawuls woll bu scremblud.
```

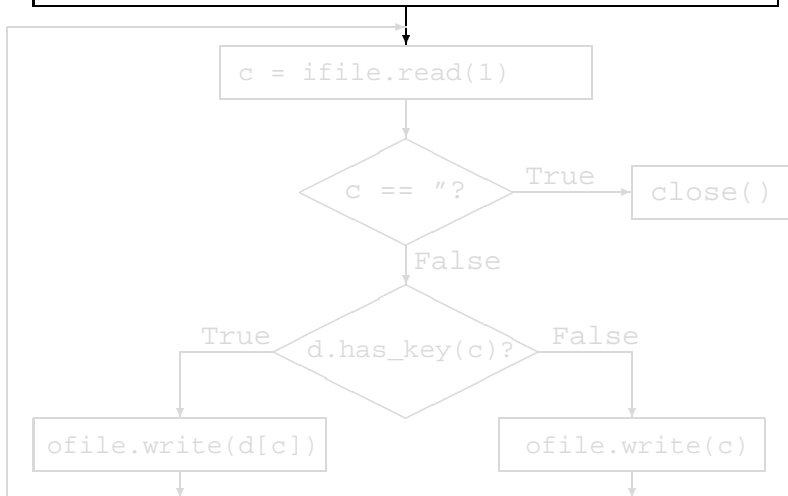
The cipher used:

```
['a','i','e','u','o'] -> ['e','o','u','i','a']
```

Python: `file.read(1)` returns one byte read from file.

The Scrambling Algorithm

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}  
infile = open('sometext','r')  
outfile = open('codetext','w')
```



Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

The Scrambling Algorithm

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}  
infile = open('sometext','r')  
outfile = open('codetext','w')
```

```
c = infile.read(1)
```

```
c == ""?
```

True

```
close()
```

False

True

```
d.has_key(c)?
```

False

```
outfile.write(d[c])
```

```
outfile.write(c)
```

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

The Scrambling Algorithm

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}  
ifile = open('sometext','r')  
ofile = open('codetext','w')
```

```
c = ifile.read(1)
```

```
c == "?"
```

True

```
close()
```

False

True

```
d.has_key(c)?
```

False

```
ofile.write(d[c])
```

```
ofile.write(c)
```

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

The Scrambling Algorithm

```
d = {'a':'e', 'e':'u', 'i':'o', 'o':'a', 'u':'i'}  
ifile = open('sometext', 'r')  
ofile = open('codetext', 'w')
```

```
c = ifile.read(1)
```

```
c == "?"
```

True

```
close()
```

False

True

```
d.has_key(c)?
```

False

```
ofile.write(d[c])
```

```
ofile.write(c)
```

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

The Scrambling Algorithm

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}  
infile = open('sometext','r')  
outfile = open('codetext','w')
```

```
c = infile.read(1)
```

```
c == "?"
```

True

```
close()
```

False

True

```
d.has_key(c)?
```

False

```
outfile.write(d[c])
```

```
outfile.write(c)
```

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

The Scrambling Algorithm

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}  
infile = open('sometext','r')  
outfile = open('codetext','w')
```

```
c = infile.read(1)
```

```
c == ""?
```

True

```
close()
```

False

True

```
d.has_key(c)?
```

False

```
outfile.write(d[c])
```

```
outfile.write(c)
```

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

The Scrambling Algorithm

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}  
ifile = open('sometext','r')  
ofile = open('codetext','w')
```

```
c = ifile.read(1)
```

```
c == "?"
```

True

```
close()
```

False

True

```
d.has_key(c)?
```

False

```
ofile.write(d[c])
```

```
ofile.write(c)
```

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text**
- counting and replacing words

Summary + Assignments

Scrambling Vowels

the program `scramvow.py`

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}
print d.keys() , '->' , d.values()
name = raw_input('Name of input file : ')
infile = open(name,'r')
name = raw_input('Name of output file : ')
ofile = open(name,'w')
while True:
    c = infile.read(1)
    if c == '': break
    if d.has_key(c):
        ofile.write(d[c])
    else:
        ofile.write(c)
infile.close()
ofile.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Scrambling Vowels

the program `scramvow.py`

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}
print d.keys() , '->' , d.values()
name = raw_input('Name of input file : ')
ifile = open(name,'r')
name = raw_input('Name of output file : ')
ofile = open(name,'w')
while True:
    c = ifile.read(1)
    if c == '': break
    if d.has_key(c):
        ofile.write(d[c])
    else:
        ofile.write(c)
ifile.close()
ofile.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Scrambling Vowels

the program `scramvow.py`

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}
print d.keys() , '->' , d.values()
name = raw_input('Name of input file : ')
ifile = open(name,'r')
name = raw_input('Name of output file : ')
ofile = open(name,'w')
while True:
    c = ifile.read(1)
    if c == '': break
    if d.has_key(c):
        ofile.write(d[c])
    else:
        ofile.write(c)
ifile.close()
ofile.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Scrambling Vowels

the program `scramvow.py`

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}
print d.keys() , '->' , d.values()
name = raw_input('Name of input file : ')
infile = open(name,'r')
name = raw_input('Name of output file : ')
ofile = open(name,'w')
while True:
    c = infile.read(1)
    if c == '': break
    if d.has_key(c):
        ofile.write(d[c])
    else:
        ofile.write(c)
infile.close()
ofile.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Scrambling Vowels

the program `scramvow.py`

```
d = {'a':'e','e':'u','i':'o','o':'a','u':'i'}
print d.keys() , '->' , d.values()
name = raw_input('Name of input file : ')
infile = open(name,'r')
name = raw_input('Name of output file : ')
ofile = open(name,'w')
while True:
    c = infile.read(1)
    if c == '': break
    if d.has_key(c):
        ofile.write(d[c])
    else:
        ofile.write(c)
infile.close()
ofile.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Outline

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing words

Python

format conversions
encrypting text
counting and replacing words

Summary + Assignments

Summary + Assignments

Counting Words in a Text

problem and algorithm

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Problem: count number of times `the` occurs.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then the number of times we count `the` is 4.

Algorithm:

1. maintain a buffer of 3 consecutive letters on file
2. after each new byte read, compare with `the`

Counting Words in a Text

problem and algorithm

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Problem: count number of times `the` occurs.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then the number of times we count `the` is 4.

Algorithm:

1. maintain a buffer of 3 consecutive letters on file
2. after each new byte read, compare with `the`

Counting Words in a Text

problem and algorithm

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Problem: count number of times `the` occurs.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then the number of times we count `the` is 4.

Algorithm:

1. maintain a buffer of 3 consecutive letters on file
2. after each new byte read, compare with `the`

Counting Words in a Text

problem and algorithm

MCS 260 L-22

17 October 2007

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Problem: count number of times `the` occurs.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then the number of times we count `the` is 4.

Algorithm:

1. maintain a buffer of 3 consecutive letters on file
2. after each new byte read, compare with `the`

Updating a Buffer

the function `add_to_buffer`

A buffer is a list of 3 characters.

Recall that we cannot assign to a function argument.

```
def add_to_buffer(b,c):  
    "adds c to a 3-letter buffer b"  
    B = b  
    if len(b) == 3:  
        B[0] = B[1]  
        B[1] = B[2]  
        B[2] = c  
    else:  
        B.append(c)  
    return B
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Updating a Buffer

the function `add_to_buffer`

A buffer is a list of 3 characters.

Recall that we cannot assign to a function argument.

```
def add_to_buffer(b,c):  
    "adds c to a 3-letter buffer b"  
    B = b  
    if len(b) == 3:  
        B[0] = B[1]  
        B[1] = B[2]  
        B[2] = c  
    else:  
        B.append(c)  
    return B
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Updating a Buffer

the function `add_to_buffer`

A buffer is a list of 3 characters.

Recall that we cannot assign to a function argument.

```
def add_to_buffer(b,c):  
    "adds c to a 3-letter buffer b"  
    B = b  
    if len(b) == 3:  
        B[0] = B[1]  
        B[1] = B[2]  
        B[2] = c  
    else:  
        B.append(c)  
    return B
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Updating a Buffer

the function `add_to_buffer`

A buffer is a list of 3 characters.

Recall that we cannot assign to a function argument.

```
def add_to_buffer(b,c):
    "adds c to a 3-letter buffer b"
    B = b
    if len(b) == 3:
        B[0] = B[1]
        B[1] = B[2]
        B[2] = c
    else:
        B.append(c)
    return B
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Counting Words

the main program cntword.py

```
def add_to_buffer(b,c):
    "adds c to a 3-letter buffer b"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
cnt = 0
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search: cnt += 1
file.close()
print '#occurrences of the : %d' % cnt
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Counting Words

the main program cntword.py

```
def add_to_buffer(b,c):
    "adds c to a 3-letter buffer b"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
cnt = 0
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search: cnt += 1
file.close()
print '#occurrences of the : %d' % cnt
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Counting Words

the main program cntword.py

```
def add_to_buffer(b,c):
    "adds c to a 3-letter buffer b"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
cnt = 0
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search: cnt += 1
file.close()
print '#occurrences of the : %d' % cnt
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Counting Words

the main program cntword.py

```
def add_to_buffer(b,c):
    "adds c to a 3-letter buffer b"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
cnt = 0
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search: cnt += 1
file.close()
print '#occurrences of the : %d' % cnt
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Counting Words

the main program cntword.py

```
def add_to_buffer(b,c):
    "adds c to a 3-letter buffer b"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
cnt = 0
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search: cnt += 1
file.close()
print '#occurrences of the : %d' % cnt
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Locating a Word

using the method `tell()`

Suppose we want to know the positions in the file of all occurrences of the word `the`.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `locword.py` will print

```
the occurs at [3L, 14L, 45L, 56L]
```

because `the` occurs at byte 3, 14, 45, and 56.

Same algorithm as counting words.

`file.tell()` returns current position of cursor in file.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Locating a Word

using the method `tell()`

Suppose we want to know the positions in the file of all occurrences of the word `the`.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `locword.py` will print

```
the occurs at [3L, 14L, 45L, 56L]
```

because `the` occurs at byte 3, 14, 45, and 56.

Same algorithm as counting words.

`file.tell()` returns current position of cursor in file.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Locating a Word

using the method `tell()`

Suppose we want to know the positions in the file of all occurrences of the word `the`.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `locword.py` will print

```
the occurs at [3L, 14L, 45L, 56L]
```

because `the` occurs at byte 3, 14, 45, and 56.

Same algorithm as counting words.

`file.tell()` returns current position of cursor in file.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Locating a Word

using the method `tell()`

Suppose we want to know the positions in the file of all occurrences of the word `the`.

For example: if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `locword.py` will print

```
the occurs at [3L, 14L, 45L, 56L]
```

because `the` occurs at byte 3, 14, 45, and 56.

Same algorithm as counting words.

`file.tell()` returns current position of cursor in file.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

The Program locword.py

uses function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
locations = []
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        locations.append(file.tell()-3)
file.close()
print 'the occurs at ', locations
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program locword.py

uses function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
locations = []
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        locations.append(file.tell()-3)
file.close()
print 'the occurs at ', locations
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program locword.py

uses function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
locations = []
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        locations.append(file.tell()-3)
file.close()
print 'the occurs at ', locations
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program locword.py

uses function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r')
locations = []
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        locations.append(file.tell()-3)
file.close()
print 'the occurs at ', locations
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Replacing Words

introducing the method `seek()`

Suppose we want to replace all occurrences of the word `the` by `one` in a text.

For example, if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `repword.py` will change this file into

```
At one end of one lecture we go home,  
taking one bus or one train.
```

Again same algorithm, using a 3-letter buffer.

`file.seek(offset,direction)` moves the cursor as many bytes as the value of `offset`, starting from

- (0) the beginning of the file, if `direction = 0`;
- (1) the current position, if `direction = 1`; or
- (2) the end of the file, if `direction = 2`.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Replacing Words

introducing the method `seek()`

Suppose we want to replace all occurrences of the word `the` by `one` in a text.

For example, if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `repword.py` will change this file into

```
At one end of one lecture we go home,  
taking one bus or one train.
```

Again same algorithm, using a 3-letter buffer.

`file.seek(offset,direction)` moves the cursor as many bytes as the value of `offset`, starting from

- (0) the beginning of the file, if `direction = 0`;
- (1) the current position, if `direction = 1`; or
- (2) the end of the file, if `direction = 2`.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Replacing Words

introducing the method `seek()`

Suppose we want to replace all occurrences of the word `the` by `one` in a text.

For example, if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `repword.py` will change this file into

```
At one end of one lecture we go home,  
taking one bus or one train.
```

Again same algorithm, using a 3-letter buffer.

`file.seek(offset,direction)` moves the cursor as many bytes as the value of `offset`, starting from

- (0) the beginning of the file, if `direction = 0`;
- (1) the current position, if `direction = 1`; or
- (2) the end of the file, if `direction = 2`.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Replacing Words

introducing the method `seek()`

Suppose we want to replace all occurrences of the word `the` by `one` in a text.

For example, if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `repword.py` will change this file into

```
At one end of one lecture we go home,  
taking one bus or one train.
```

Again same algorithm, using a 3-letter buffer.

`file.seek(offset,direction)` moves the cursor as many bytes as the value of `offset`, starting from

- (0) the beginning of the file, if `direction = 0`;
- (1) the current position, if `direction = 1`; or
- (2) the end of the file, if `direction = 2`.

Files

- gzip and gunzip
- data compression

Python

- format conversions
- encrypting text
- counting and replacing words

Summary + Assignments

Replacing Words

introducing the method `seek()`

Suppose we want to replace all occurrences of the word `the` by `one` in a text.

For example, if the file `sometext2` has content

```
At the end of the lecture we go home,  
taking the bus or the train.
```

then `repword.py` will change this file into

```
At one end of one lecture we go home,  
taking one bus or one train.
```

Again same algorithm, using a 3-letter buffer.

`file.seek(offset,direction)` moves the cursor as many bytes as the value of `offset`, starting from

- (0) the beginning of the file, if `direction = 0`;
- (1) the current position, if `direction = 1`; or
- (2) the end of the file, if `direction = 2`.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program repword.py

uses the function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r+') # reading AND writing
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        file.seek(-3,1)
        file.write('one')
file.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program repword.py

uses the function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r+') # reading AND writing
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        file.seek(-3,1)
        file.write('one')
file.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program repword.py

uses the function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r+') # reading AND writing
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        file.seek(-3,1)
        file.write('one')
file.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

The Program repword.py

uses the function `add_to_buffer`

```
def add_to_buffer(b,c):
    "adds c to 3-letter buffer"
    ...
search = ['t','h','e']
buffer = []
name = raw_input('give file name : ')
file = open(name,'r+') # reading AND writing
while True:
    c = file.read(1)
    if c == '': break
    buffer = add_to_buffer(buffer,c)
    if buffer == search:
        file.seek(-3,1)
        file.write('one')
file.close()
```

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments

Summary + Assignments

We covered more of chapter 7 in *Making Use of Python*; for Unix, see §15.6.2 in *The Art & Craft of Computing*.

Assignments:

1. Write a program `bkform2dict.py` to convert the formatting in the file `books` directly to `booksdicts`.
2. Rewrite the program `library.py` of Lecture 21, using dictionaries as formats for the file `books`.
3. Modify the `cntword.py` so that it considers a word as separated from others by spaces or commas and other punctuation symbols, i.e.: the `the` in `there` or `then` does not count as an occurrence of `the`.
4. Design a permutation of the vowels so that after applying `scramvow.py` twice we get the original message.

Homework will be collected on Monday 22 October.

Files

gzip and gunzip
data compression

Python

format conversions
encrypting text
counting and replacing
words

Summary + Assignments