
9. ΕΓΓΡΑΦΗ ΚΑΙ ΑΝΑΓΝΩΣΗ ΑΡΧΕΙΩΝ

9.0.1 Εγγραφή αρχείων

Τα αρχεία κειμένου τα μεταχειριζόμαστε σαν τετράδια. Ένα τετράδιο πρώτα το ανοίγουμε και, όταν τελειώσουμε το γράψιμο, το κλείνουμε. Όταν είναι ανοικτό, μπορούμε να διαβάσουμε από αυτό ή να γράψουμε σε αυτό. Σε κάθε περίπτωση ξέρουμε πού ακριβώς βρισκόμαστε μέσα στο τετράδιο.

Όλα αυτά εφαρμόζονται και στα αρχεία. Για να ανοίξουμε ένα αρχείο, δηλώνουμε εάν θέλουμε να το διαβάσουμε (read) ή να το γράψουμε (write). Όταν τελειώσουμε την εργασία μας μ' αυτό, θα πρέπει να το κλείσουμε (close).

Σ' αυτό το μάθημά μας θα διαχειριστούμε αρχεία κειμένου (text files), τα οποία δεν είναι τίποτε άλλο παρά μια σειρά από χαρακτήρες (συμβολοσειρές) που είναι αποθηκευμένοι σε ένα μόνιμο μέσο αποθήκευσης (σκληρό δίσκο, κ.λπ.).

Τα αρχεία κειμένου μπορούν να διαβαστούν από έναν άνθρωπο, σε αντίθεση με τα δυαδικά αρχεία (binary files) για τα οποία απαιτούνται κατάλληλα προγράμματα για την ανάγνωση και την χρήση τους.

Ο πρώτος τύπος αρχείων τον οποίο θα διαβάσουμε, όπως είπαμε είναι τα απλά αρχεία κειμένου (.txt). Πρώτα ας δημιουργήσουμε ένα αρχειάκι myfile.txt με λίγες γραμμές κειμένου (που δανειζόμαστε από το παραπάνω κείμενό μας):

«Σ' αυτό το μάθημά μας θα διαχειριστούμε αρχεία κειμένου (text files), τα οποία δεν είναι τίποτε άλλο παρά μια σειρά από χαρακτήρες (συμβολοσειρές) που είναι αποθηκευμένοι σε ένα μόνιμο μέσο αποθήκευσης.»

```
f = open("C:\\Users\\NK\\Desktop\\myfile.txt", "w")

line1 = "Σ' αυτό το μάθημά μας θα διαχειριστούμε αρχεία  
κειμένου\\n"

f.write(line1)

line2 = "κειμένου (text files), τα οποία δεν είναι τίποτε άλλο παρά  
\\n"

f.write(line2)

line3 = "μια σειρά από χαρακτήρες (συμβολοσειρές) που είναι  
\\n"

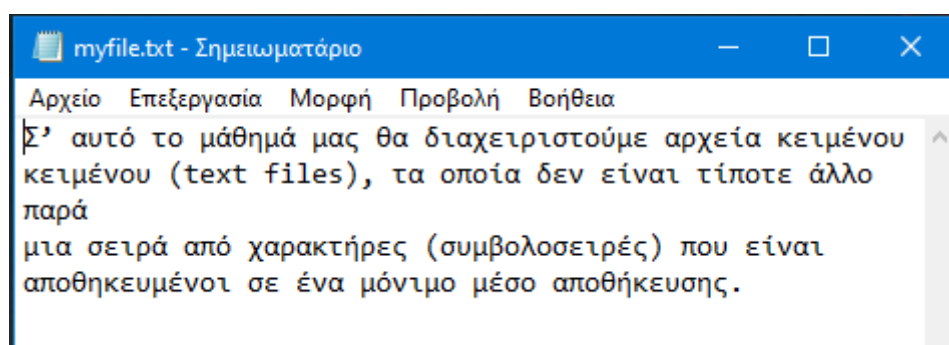
f.write(line3)

line4 = "αποθηκευμένοι σε ένα μόνιμο μέσο αποθήκευσης.\\n"

f.write(line4)

f.close()
```

Έτσι δημιουργούμε το παρακάτω αρχείο:



Η συνάρτηση open δέχεται δύο ορίσματα: το πρώτο είναι το όνομα του

αρχείου (χωρίς αναφορά μονοπατιού, εξ ορισμού ο φάκελος του αρχείου θα ήταν ο φάκελος στον οποίο είναι εγκατεστημένη η Python) και το δεύτερο όρισμα είναι ο τρόπος ανοίγματος (mode).

Το mode 'w' (write, εγγραφή) σημαίνει ότι ανοίγουμε το αρχείο για γράψιμο. Αν το αρχείο που ανοίγουμε για εγγραφή υπάρχει ήδη, τότε τα δεδομένα που περιέχει σβήνονται και αντικαθίστανται από τα νέα.

Αν το αρχείο δεν υπάρχει, τότε αυτό απλά δημιουργείται και είναι έτοιμο για να γράψουμε δεδομένα σε αυτό. Η εκτέλεση της εντολής open δημιουργεί ένα αντικείμενο αρχείου πάνω στο οποίο μπορούμε να καλέσουμε μεθόδους. Με τη μέθοδο write μπορούμε να γράψουμε δεδομένα σε ένα αρχείο.

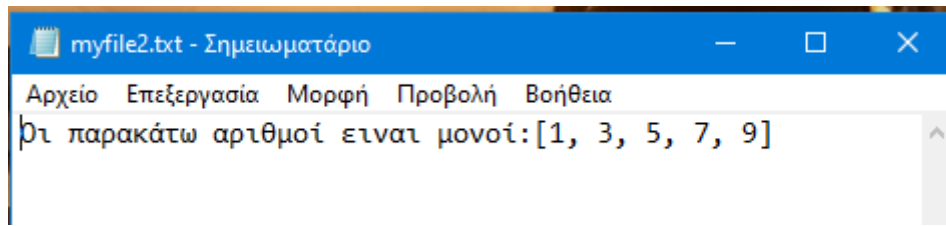
Επίσης, το όρισμα της write πρέπει να είναι συμβολοσειρά. Έτσι, αν θέλουμε να γράψουμε και άλλες τιμές (π.χ. αριθμούς), θα πρέπει πρώτα να τις μετατρέψουμε σε συμβολοσειρές με χρήση της συνάρτησης str (casting).

Αν τρέξουμε κάποιες γραμμές κώδικα στον IDLE και καλέσουμε την write(), θα δούμε πως εμφανίζεται ο αριθμός των χαρακτήρων που γράφεται στο αρχείο με κάθε εντολή.

Παράδειγμα:

```
>>> f = open('C:\\Users\\NK\\Desktop\\myfile2.txt', 'w')
...
>>> f.write('Οι παρακάτω αριθμοί είναι μονοί:')
...
32
>>> f.write(str([1,3,5,7,9]))
...
15
>>> f.close()
```

ενώ τώρα, το νέο αρχείο μας θα είναι:



Για να αλλάξουμε γραμμή σε ένα αρχείο, χρησιμοποιούμε τον χαρακτήρα διαφυγής `\n`.

Στο τέλος το κλείσιμο του αρχείου γίνεται με την κλήση της μεθόδου `close` πάνω στο αντικείμενο του αρχείου.

Επίσης πρέπει να πούμε πως το μονοπάτι φακέλων που δίνεται ως όρισμα στη συνάρτηση `open` (π.χ. στα προηγούμενα παραδείγματα το `C:\myfolder\`), πρέπει να προϋπάρχει, διαφορετικά θα εμφανιστεί μήνυμα λάθους.

9.0.2 Ανάγνωση αρχείων

Για να ανοίξουμε ένα αποθηκευμένο αρχείο κειμένου για ανάγνωση, πρέπει να χρησιμοποιήσουμε τον `mode 'r'` (read, ανάγνωση). Αν προσπαθήσουμε να ανοίξουμε για ανάγνωση ένα αρχείο που δεν υπάρχει, τότε θα εμφανιστεί μήνυμα λάθους. Το περιεχόμενο ενός αρχείου διαβάζεται ως συμβολοσειρά.

Τώρα που έχουμε μάθει να γράφουμε ένα αρχείο, πάμε να το ανοίξουμε για ανάγνωση.

```
f = open("C:\\Users\\NK\\Desktop\\myfile.txt", "r")
```

```
firstline = f.readline()
```

```
secondline = f.readline()
```

```
print (firstline)
```

```
print (secondline)
```

f.close()

Τα συχνά χρησιμοποιούμενα modes είναι:

'r' mode:

Μόνο για ανάγνωση.

'w' mode:

Μόνο για εγγραφή.

Αν το συγκεκριμένο αρχείο δεν υπάρχει, θα δημιουργηθεί.

Αν το συγκεκριμένο αρχείο υπάρχει, οτιδήποτε υπάρχει μέσα θα διαγραφεί.

'a' mode:

Για προσθήκη κειμένου.

Αν το συγκεκριμένο αρχείο δεν υπάρχει, θα δημιουργηθεί.

Αν υπάρχει, οτιδήποτε υπάρχει μέσα θα προστεθεί στο τέλος του αρχείου.

'r+' mode:

Και για γραφή και για ανάγνωση.

'x' mode:

Δημιουργεί ένα αρχείο κι επιστρέφει λάθος αν ήδη υπάρχει.

Αφού ανοίξουμε το αρχείο, η επόμενη δήλωση

firstline = f.readline()

διαβάζει την πρώτη γραμμή του αρχείου και την αναθέτει στη μεταβλητή ***firstline***.

Κάθε φορά που καλείται η συνάρτηση ***readline()***, διαβάζει μια νέα γραμμή από το αρχείο.

Στο παραπάνω προγραμματάκι η ***readline()*** καλείται τέσσερις φορές.

Έτσι, θα έχουμε την έξοδο:

Σ' αυτό το μάθημά μας θα διαχειριστούμε αρχεία

κειμένου (text files), τα οποία δεν είναι τίποτε άλλο παρά

μια σειρά από χαρακτήρες (συμβολοσειρές) που είναι

αποθηκευμένοι σε ένα μόνιμο μέσο αποθήκευσης.

Θα παρατηρήσετε ότι μετά από κάθε γραμμή εισάγεται μια αλλαγή γραμμής. Αυτό συμβαίνει επειδή η συνάρτηση `readline()` προσθέτει τους χαρακτήρες `'\n'` στο τέλος κάθε γραμμής.

Αν δεν θέλουμε την επιπλέον γραμμή μεταξύ κάθε γραμμής κειμένου, μπορούμε να πούμε:

`print (firstline, end= " ").` Αυτό θα αφαιρέσει τους χαρακτήρες `'\n'`.

Σημείωση:

Το " αποτελείται από δύο μονά εισαγωγικά κι όχι από ένα διπλό εισαγωγικό.

Έτσι τώρα έχουμε ένα πολύ πιο σωστό αποτέλεσμα εκτύπωσης:

***Σ' αυτό το μάθημά μας θα διαχειριστούμε αρχεία
κειμένου (text files), τα οποία δεν είναι τίποτε άλλο παρά
μια σειρά από χαρακτήρες (συμβολοσειρές) που είναι
αποθηκευμένοι σε ένα μόνιμο μέσο αποθήκευσης.***

Αφού διαβάσουμε και εκτυπώσουμε τις τέσσερις πρώτες γραμμές, η τελευταία πρόταση `f.close()` κλείνει το αρχείο.

Πρέπει πάντα να κλείνουμε το αρχείο μόλις ολοκληρώσουμε την ανάγνωσή του για να ελευθερώσουμε τους πόρους του συστήματος.

Μπορούμε να δούμε και τα παρακάτω συμπληρωματικά παραδείγματα:

```
# Άνοιγμα και προσθήκη κειμένου στο αρχείο myfile.txt:
```

```
f = open("myfile.txt", "a")
f.write("Τώρα το αρχείο έχει περισσότερο περιεχόμενο!")
f.close()
```

```
# Άνοιγμα και διάβασμα του αρχείου μετά την προσθήκη κειμένου:
```

```
f = open("myfile.txt", "r")
print(f.read())
```

```
# Άνοιγμα του αρχείου και επανωγράψιμο (το υπάρχον κείμενο διαγράφεται):
```

```
f = open("myfile.txt", "w")
f.write("Ωπα! Έχουμε διαγράψει το περιεχόμενο!")
f.close()
```

```
# Άνοιγμα και ανάγνωση του αρχείου μετά το επανωγράψιμο:
```

```
f = open("myfile.txt", "r")
print(f.read())
```

Δημιουργία αρχείου:

Δημιουργία του αρχείου "myfile.txt":

```
f = open("myfile.txt", "x")
```

Το αποτέλεσμα είναι να δημιουργηθεί ένα κενό αρχείο!

Δημιουργία ενός αρχείου που δεν υπάρχει:

```
f = open("myfile.txt", "w")
```

9.0.3 Χρήση ενός for loop για ανάγνωση κειμένων

Είναι αναμενόμενο να μην καλούμε την `readline()` για να διαβάσει ένα κείμενο γραμμή —γραμμή.

Για να δούμε πώς μπορούμε να διαβάζουμε ένα αρχείο χρησιμοποιώντας ένα for loop:

```
f = open ('myfile.txt', 'r')  
  
for line in f:  
  
    print (line, end = '')  
  
f.close()
```

9.0.4 Χρήση της `read()` για ανάγνωση αρχείων με το μέγεθος του buffer

Μερικές φορές, για να μην δεσμεύουμε πόρους της μνήμης, μπορεί να θέλουμε να διαβάσουμε ένα αρχείο με το μέγεθος ενός buffer. Για να το κάνουμε αυτό, χρησιμοποιούμε τη συνάρτηση `read()`, κι όχι τη `readline()`, με την οποία μπορούμε να καθορίσουμε το μέγεθος του buffer.

Παράδειγμα:

```
inputFile = open ('myfile.txt', 'r')
```

```
outputFile = open ('myoutputfile.txt', 'w')
```

```
msg = inputFile.read(10)
```

```
while len(msg):
```

```
outputFile.write(msg)
```

```
msg = inputFile.read(10)
```

```
inputFile.close()
```

```
outputFile.close()
```

Πρώτα, ανοίγουμε δύο αρχεία. Το **inputFile.txt** και το **outputFile.txt** για ανάγνωση και γραφή αντίστοιχα.

Κατόπιν, χρησιμοποιούμε τη δήλωση **msg = inputFile.read(10)** και ένα **while loop** για να διαβάσει το αρχείο ανά 10 bytes τη φορά. Αυτό δηλώνουμε με την τιμή «10» στις παρενθέσεις της **read()**.

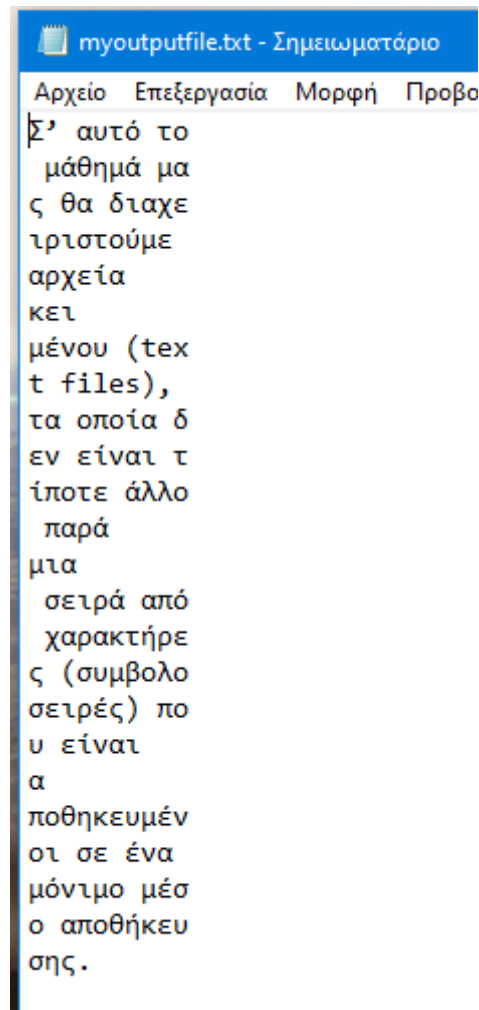
Η συνθήκη **while, while len(msg)**: ελέγχει το μήκος της μεταβλητής **msg**. όσο αυτό το μήκος δεν είναι ίσο με το μηδέν, ο βρόχος συνεχίζει να τρέχει.

Εντός του βρόχου **while**, η δήλωση **outputFile.write(msg)** γράφει το μήνυμα στο αρχείο εξόδου. Αφού γράψουμε το μήνυμα, η δήλωση **msg = inputFile.read(10)** διαβάζει τα επόμενα 10 byte και συνεχίζει να το κάνει μέχρι να διαβαστεί όλο το αρχείο. Όταν συμβεί αυτό, το πρόγραμμα κλείνει και τα δύο αρχεία.

Όταν εκτελούμε το πρόγραμμα, θα δημιουργηθεί ένα νέο αρχείο **myoutputfile.txt**. Όταν ανοίξουμε το αρχείο, θα παρατηρήσουμε ότι έχει το ίδιο περιεχόμενο με το αρχείο εισόδου μας **myfile.txt**. Για να δούμε αν διαβάζονται μόνο 10 byte κάθε φορά, μπορούμε να

αλλάξουμε τη γραμμή **`outputFile.write(msg)`** στο πρόγραμμα σε **`outputFile.write(msg + '\n')`**.

Τώρα, αν εκτελέσουμε ξανά το πρόγραμμα, το **`myoutputfile.txt`** περιέχει γραμμές με το πολύ 10 χαρακτήρες. Ας δούμε το αρχειάκι μας τώρα:



```
myoutputfile.txt - Σημειωματάριο
Αρχείο  Επεξεργασία  Μορφή  Προβο
Σ' αυτό το
μάθημά μα
ς θα διαχε
ιριστούμε
αρχεία
κει
μένου (tex
t files),
τα οποία δ
εν είναι τ
ίποτε άλλο
παρά
μια
σειρά από
χαρακτήρε
ς (συμβολο
σειρές) πο
υ είναι
α
ποθηκευμέν
οι σε ένα
μόνιμο μέσ
ο αποθήκευ
σης.
```

9.0.5 Προσθήκη δεδομένων σε ήδη υπάρχον αρχείο

Μπορούμε να ανοίξουμε ένα ήδη υπάρχον αρχείο και να προσθέσουμε δεδομένα στο τέλος του, χρησιμοποιώντας το mode 'a':

Ας δημιουργήσουμε ένα αρχειάκι με τον παρακάτω κώδικα:

```
f = open('C:\\Users\\NK\\Desktop\\myfile.txt', 'a')
```

```
f.write('Τελευταία γραμμή')
```

```
f.close()
```

```
f = open('C:\\Users\\NK\\Desktop\\myfile.txt', 'r')
```

```
text = f.read()
```

```
print(text)
```

```
f.close()
```

το οποίο προσθέτει λίγο κείμενο («Τελευταία γραμμή») στο τέλος του ήδη υπάρχοντος, κι ας το τρέξουμε:

```
>>>
```

```
== RESTART:
```

```
C:/Users/NK/AppData/Local/Programs/Python/Python311/read_w  
rite.py ==
```

*Σ' αυτό το μάθημά μας θα διαχειριστούμε αρχεία
κειμένου (text files), τα οποία δεν είναι τίποτε άλλο παρά*

μια σειρά από χαρακτήρες (συμβολοσειρές) που είναι

*αποθηκευμένοι σε ένα μόνιμο μέσο αποθήκευσης. Τελευταία
γραμμή*

9.0.6 Άνοιγμα, ανάγνωση κι εγγραφή δυναμικών αρχείων

Τα δυναμικά αρχεία αναφέρονται σε οποιοδήποτε αρχείο που δεν περιέχει κείμενο, όπως εικόνα ή βίντεο. Για να δουλέψουμε με δυναμικά αρχεία, χρησιμοποιούμε απλώς τη λειτουργία 'rb' ή 'wb'. Αντιγράφουμε ένα αρχείο jpg στην επιφάνεια εργασίας σας και το μετονομάζουμε σε *myimage.jpg*. Τώρα αλλάζουμε στο παραπάνω πρόγραμμα τις δύο πρώτες γραμμές:

```
inputFile = open ('myfile.txt', 'r')
```

```
outputFile = open ('myoutputfile.txt', 'w')
```

σε:

```
inputFile = open ('myimage.jpg', 'rb')
```

```
outputFile = open ('myoutputimage.jpg', 'wb')
```

Επίσης αλλάζουμε τη δήλωση *outputFile.write(msg + '\n')*

σε *outputFile.write(msg)*.

Εκτελώντας το νέο πρόγραμμα έχουμε ένα επιπλέον αρχείο εικόνας με όνομα *myoutputimage.jpg* στην επιφάνεια εργασίας μας. Όταν ανοίξουμε το αρχείο εικόνας, θα πρέπει να είναι ακριβώς το ίδιο με το *myimage.jpg*.



myoutputimage.jpg

9.0.7 Ασκήσεις

Άσκηση 1: Δημιουργία και εγγραφή σε αρχείο

Δημιουργήστε ένα νέο αρχείο με το όνομα "my_file.txt" και γράψτε το παρακάτω κείμενο σε αυτό: "Γειά σου κόσμε! Αυτό είναι το πρώτο αρχειάκι που δημιούργησα στην Python."

Άσκηση 2: Ανάγνωση από ένα αρχείο

Ανοίξτε το αρχείο "my_file.txt" που δημιουργήθηκε στην Άσκηση 1 και διαβάστε το περιεχόμενό του. Εκτυπώστε το περιεχόμενο στην κονσόλα.

Άσκηση 3: Προσάρτηση σε ένα αρχείο

Ανοίξτε το αρχείο "my_file.txt" και προσθέστε το παρακάτω κείμενο στο τέλος του αρχείου: "Αυτό είναι κείμενο που προστέθηκε στο αρχείο."

Άσκηση 4: Ανάγνωση ενός αρχείου γραμμή προς γραμμή

Δημιουργήστε ένα αρχείο με το όνομα "my_poem.txt" και γράψτε το παρακάτω ποίημα σε αυτό:

Roses are red,
Violets are blue,
Sugar is sweet,
And so are you.

Ανοίξτε το αρχείο "my_poem.txt" και διαβάστε το περιεχόμενό του γραμμή προς γραμμή. Εκτυπώστε κάθε γραμμή στην κονσόλα.

Άσκηση 5: Δημιουργία αρχείου αν δεν υπάρχει

Ανοίξτε το αρχείο "my_notes.txt" αν υπάρχει ή δημιουργήστε ένα νέο αρχείο με αυτό το όνομα αν δεν υπάρχει. Γράψτε το παρακάτω κείμενο στο αρχείο: "Αυτές είναι κάποιες σημειώσεις μου για την Python."

Άσκηση 6: Προσάρτηση σε ένα αρχείο αν δεν υπάρχει

Προσθέστε το παρακάτω κείμενο στο αρχείο "my_notes.txt" αν υπάρχει ή δημιουργήστε ένα νέο αρχείο με αυτό το όνομα αν δεν υπάρχει: "Έχουμε κάποιες επιπρόσθετες σημειώσεις για την Python."

Άσκηση 7: Αντιγραφή περιεχομένου από ένα αρχείο σε ένα άλλο

Δημιουργήστε ένα αρχείο με το όνομα "source.txt" και γράψτε το παρακάτω κείμενο σε αυτό: "Αυτό είναι το περιεχόμενο του πηγαίου αρχείου."

Δημιουργήστε ένα κενό αρχείο με το όνομα "destination.txt".

Ανοίξτε τα δύο αρχεία και αντιγράψτε το περιεχόμενο του "source.txt" στο "destination.txt".