File Handling & I/O

ΓΛΩΣΣΙΚΉ ΤΕΧΝΟΛΟΓΊΑ

.NET Framework

- System.Out namespace
  - Βασική υποστήριξη για διαχείριση αρχείων και φακέλων
  - Ανάγνωση και εγγραφή σε file και data streams
- Κλήση εκτελέσιμου από κώδικα

System.IO.File

- Παρέχει μεθόδους για την δημιουργία, αντιγραφή, μεταφορά και άνοιγμα αρχείων.
- Σημαντικά μέλη:
  - Exists(string path): με δεδομένο το path, επιστρέφει αν υπάρχει το αρχείο ή όχι.
  - Open(string path): ανοίγει ένα αρχείο.
  - Create(string path): δημιουργεί αρχείο.
  - CreateText(string path): δημιουργεί αρχείο κειμένου.
  - Delete(string path): διαγράφει αρχείο.
  - Move(string source, string dest): μεταφέρει αρχείο.
  - Copy(string source, string dest): αντιγράφει αρχείο.

System.IO.Directory

- Παρέχει μεθόδους για τη δημιουργία, διαγραφή, μεταφορά ή απαρίθμηση των περιεχομένων μιας directory.
- Σημαντικά μέλη:
  - Exists(string path): έλεγχος ύπαρξης του directory
  - GetFiles(string path): επιστρέφει όλα τα files του directory.
  - GetFiles(string path, string pattern) : επιστρέφει τα files που ταιριάζουν στο συγκεκριμένο pattern.
  - GetDirectories(string path): επιστρέφει όλα τα subdirectories.
  - CreateDirectory(string path): δημιουργία directory
  - Delete, Move κλπ

File & Directory listing

```
static void MoveContents()
{
    string[] folders;
    string[] files;
    string output_folder;

    if (System.IO.Directory.Exists("C:\\output_folder\\") == false)
        System.IO.Directory.CreateDirectory("C:\\output_folder\\");

    if (System.IO.Directory.Exists("C:\\input_folder\\") == false)
    {
        Console.WriteLine("Input folder doesn't exist. Operation will be cancelled.");
        return;
    }

    files = System.IO.Directory.GetFiles("C:\\input_folder\\");
    foreach (string file in files)
    {
        File.Move(file, "C:\\output_folder\\" + file.Substring(file.LastIndexOf('\\') + 1));
    }

    folders = System.IO.Directory.GetDirectories("C:\\input_folder\\");
    foreach (string folder in folders)
    {
        Directory.Move(folder, "C:\\output_folder\\" + folder.Substring(folder.LastIndexOf('\\') + 1));
    }

}
```

File & Directory listing

```
static void MoveTextFiles()
{
    string[] files;

    if (System.IO.Directory.Exists("C:\\output_folder\\") == false)
        System.IO.Directory.CreateDirectory("C:\\output_folder\\");

    if (System.IO.Directory.Exists("C:\\input_folder\\") == false)
    {
        Console.WriteLine("Input folder doesn't exist. Operation will be cancelled.");
        return;
    }

    files = System.IO.Directory.GetFiles("C:\\input_folder\\", "*.txt");
    foreach (string file in files)
    {
        File.Move(file, "C:\\output_folder\\" + file.Substring(file.LastIndexOf('\\') + 1));
    }

}
```

Common exceptions

- FileNotFoundException: Προσπάθεια για προσπέλαση αρχείου που δεν υπάρχει.
- DirectoryNotFoundException: Προσπάθεια για προσπέλαση directory που δεν υπάρχει
- PathTooLongException: Χρήση path που είναι πολύ μεγάλο για να υποστηριχτεί από το σύστημα.
- IOException : Γενικό I/O error

System.IO.StreamReader

- Κλάση για την ανάγνωση stream από αρχείο.
- Δημιουργία:
  StreamReader sr = new StreamReader(path);
- Σημαντικά μέλη:
  - Close(): Πάντα τα κλείνουμε!!!
  - ReadLine(): Διαβάζει και επιστρέφει μία γραμμη.
  - EndOfStream: Flag για το αν έχουμε φτάσει στο τέλος του αρχείου.
  - ReadToEnd() : Διαβάζει μέχρι το τέλος του αρχείου.
  - Peek(): Επιστρέφει τον επόμενο χαρακτήρα χωρίς να τον καταναλώσει.

System.IO.StreamWriter

- Κλάση για την εγγραφή σε αρχείο.
- Δημιουργία:
  StreamWriter sw = new StreamWriter(path);
- Σημαντικά μέλη:
  - Close(): Είπαμε, πάντα τα κλείνουμε!!! Αν δεν κληθεί, δεν γράφεται τίποτα.
  - WriteLine(string line): Γράφει το περιεχόμενο και new line.
  - Write(string content): Γράφει το περιεχόμενο.

File I/O example

```
static void CopyFileContents(string input_path, string output_path)
{
    StreamReader sr = new StreamReader(input_path);
    //ή StreamReader sr = File.OpenText(input_path);
    StreamWriter sw = new StreamWriter(output_path);
    //ή StreamWriter sw = File.OpenText(output_path);
    string line;
    int i = 1;

    while (sr.EndOfStream == false)
    {
        line = sr.ReadLine();
        sw.Write(i + ":\t");
        sw.WriteLine(line);
        i++;
    }

    sw.Close();
    sr.Close();
}

static void AppendToFile(string input_path, string output_path)
{
    StreamWriter sw = new StreamWriter(output_path, true);
    //ή StreamWriter sw = File.AppendText(output_path);

    sw.Write("Hello World!");
    sw.Close();
}
```

Process call

- Όταν έχουμε ένα εκτελέσιμο το οποίο θέλουμε να καλέσουμε από κώδικα και να πάρουμε την έξοδό του, χρησιμοποιούμε την κλάση Process.
- Σημαντικά μέλη:
  - StartInfo: Ιδιότητα για τα preferences της εκτέλεσης.
  - Start(): Μέθοδος για να ξεκινήσει το process να εκτελείται.
  - WaitForExit(): Μέθοδος έτσι ώστε η τρέχουσα διεργασία να περιμένει την κληθείσα να τερματίσει πριν συνεχίσει.

Process call - example

```
static void CallTagger(string infilepath, string outfilepath)
{
    string tagCommand;
    tagCommand = "LEXICON.BROWN ";
    tagCommand += "\"" + infilepath + "\"";
    tagCommand += " BIGRAMS LEXICALRULEFILE.BROWN CONTEXTUALRULEFILE.BROWN";

    Process p=new Process();
    p.StartInfo.Arguments=tagCommand;
    p.StartInfo.CreateNoWindow = true;
    p.StartInfo.FileName="tagger.exe";
    p.StartInfo.UseShellExecute=false;
    p.StartInfo.RedirectStandardOutput=true;
    p.StartInfo.WindowStyle = ProcessWindowStyle.Hidden;
    p.Start();
    string output = p.StandardOutput.ReadToEnd();
    StreamWriter sw = new StreamWriter(outfilepath);
    sw.Write(output);
    sw.Close();
}
```

PYTHON

To module os I

- import os
  - Module για την διαχείρηση αρχείων και φακέλων.
  - Είναι cross-platform!!
- os.path.join(path1[, path2[, ...]])
  - Συνένωση μονοπατιών.
  - π.χ: os.path.join("c:\\music\\ap", "mahadeva.mp3")
  - Παρατηρείστε την έλλειψη “\”. Το αποτέλεσμα είναι σωστό!! Προστίθεται από μόνο του!!
- os.join.expanduser(~)
  - Μεταφορά στον “home” folder ανάλογα με το ΛΣ.
    - HOME (linux)
    - My Documents (Windows)

Το module os II

- os.path.split(path)
  - Επιστρέφει ένα tuple της μορφής (head, tail) όπου head το path μέχρι το τελευταίο "/" και tail ότι το ακολουθεί.
  - π.χ.: (filepath, filename) = os.path.split("c:\\music\\ap\\mahadeva.mp3")
    - filepath = 'c:\\music\\ap'
    - filename = 'mahadeva.mp3'

Το module os III

- os.path.splitext(path)
  - Επιστρέφει ένα tuple της μορφής (root, ext) όπου root το filename και ext η κατάληξη.
  - π.χ.: (shortname, extension) = os.path.splitext(filename)
  - shortname = 'mahadeva'
  - extension = '.mp3'

Το module os IV

- os.path.isfile(path)
  - Επιστρέφει boolean τιμή ανάλογα με το όρισμα.
- os.path.isdir(path)
  - Επιστρέφει boolean τιμή ανάλογα με το όρισμα.
- os.listdir(path)
  - Επιστρέφει μία λίστα με τα περιεχόμενα του φακέλου που δίνεται ως όρισμα.

Το module os V

- os.system(command)
  - Εκτέλεση εξωτερικής διεργασίας. Η παράμετρος command πρόκειται για ένα string που καλεί στην ουσία ένα εξωτερικό πρόγραμμα.
  - π.χ.: os.system("ls")
  - Ως επιστρεφόμενη τιμή, λαμβάνεται το exit status της διεργασίας.
  - Η υλοποίηση της παραπάνω συνάρτησης γίνεται μέσω της αντίστοιχης συνάρτησης system() της C και ακολουθείται γενικά το πρότυπο POSIX.

To module glob I

- import glob
  - Ανάκτηση full paths με χρήση wildcart.
- Παραδείγματα:
  - glob.glob('c:\\music_singles*.mp3')
  - ['c:\\music_singles\\a_time_long_forgotten_con.mp3',
  - 'c:\\music_singles\\hellraiser.mp3',
  - 'c:\\music_singles\\kairo.mp3',
  - 'c:\\music_singles\\long_way_home1.mp3',
  - 'c:\\music_singles\\sidewinder.mp3',
  - 'c:\\music_singles\\spinning.mp3']

Το module glob II

- glob.glob('c:\\music_singles\\s*.mp3')
  - ['sidewinder.mp3', 'spinning.mp3']
- glob.glob('c:\\music**.mp3')
  - Θα επιστρέψει μία λίστα με όλα τα mp3s που περιέχονται σε όλους τους υποφακέλους του φακέλου music!!

Άνοιγμα αρχείων

- open(filename, mode)
  - Το πρώτο όρισμα είναι τύπου string και περιέχει το όνομα του αρχείου (ή και το path κάτω από το οποίο αυτό υπάρχει).
  - Το δεύτερο όρισμα είναι επίσης τύπου string και υποδηλώνει τον τρόπο με τον οποίο θα χρησιμοποιηθεί το αρχείο.
  - 'r' (read - default)
  - 'w' (write)
  - 'a' (append)
  - 'r+' (both read/write)

Ανάγνωση περιεχομένου Ι

- f.read(size)
  - Η παράμετρος size είναι προεραιτική και υποδηλώνει πόσα bytes θα διαβαστούν από το αρχείο f (file object).
  - Είναι ευθύνη του προγραμματιστή να καθορίσει την τιμή της size. Αν δεν δοθεί τιμή, η read διαβάζει ολόκληρο το περιεχόμενο!!
  - Μεγάλη προσοχή στην χρήση της!! Η μνήμη δεν είναι άπειρη…

Ανάγνωση περιεχομένου II

- f.readline()
  - Ανάγνωση μίας γραμμής από το αρχείο. Ως delimeter χρησιμοποιείται ο χαρακτήρας νέας γραμμής ('\n').
  - Το επιστρεφόμενο string περιέχει τον χαρακτήρα νέας γραμμής εκτός και αν πρόκειται για το τέλος του αρχείου, το οποίο δεν τελειώνει με νέα γραμμή!!
  - Προσοχή στην επιστρεφόμενη τιμή!! Ένα κενό string είναι το τέλος του αρχείου ενώ ένα string της μορφής '\n' είναι μία κενή γραμμή!!

Ανάγνωση περιεχομένου III

- f.readlines(sizehint)
  - Επιστρέφει μία λίστα, όπου κάθε στοιχείο της είναι μία γραμμή του αρχείου.
  - Η παράμετρος sizehint είναι προεραιτική και συνίσταται για πραγματικά μεγάλα αρχεία, με σκοπό την καλύτερη διαχείρηση μνήμης. Αναφέρεται σε bytes, αλλά η συνάρτηση θα επιστρέψει μόνο ολόκληρες γραμμές.
  - Η καλύτερη προσέγγιση!!

Ανάγνωση περιεχομένου IV

- for line in f: print line
  - Απλοϊκή προσέγγιση
  - Προσπέλαση ανά γραμμή χωρίς την χρήση συνάρτησης.
  - Διαφορετικό buffering στην μνήμη!! Δεν πρέπει να χρησιμοποιείται σε συνδυασμό με τις προηγούμενες μεθόδους.

Εγγραφή

- f.write(string)
  - Τόσο απλά!!
  - Για εγγραφή άλλου τύπου δεδομένων, π.χ. int, πρέπει να γίνει πρώτα μετατροπή σε string, π.χ.:
  - a = 5
  - s = str(a)
  - f.write(s)
  - Hint: Προσοχή κατά την αντίστοιχη ανάκτηση και χρήση αριθμών από αρχεία. Πρέπει να γίνει μετατροπή σε int πριν την χρήση τους σε μαθηματικές πράξεις..

Προσπέλαση περιεχομένου

- f.tell()
  - Επιστρέφει έναν int που δείχνει σε ποιο byte (ξεκινώντας απ'την αρχή του αρχείου) βρίσκεται η προσπέλαση.
- f.seek(offset, from_what)
  - Ρητή αλλαγή στην θέση του δείκτη.
  - Η παράμετρος offset υποδηλώνει τα bytes που προστίθενται στην from_what για την μετακίνησή του κάθε φορά.
  - Η παράμετρος from_what υποδηλώνει την θέση του δείκτη.
    - 0 (απ'την αρχή του αρχείου - default)
    - 1 (απ'την τρέχουσα θέση του μέσα στο αρχείο)
    - 2 (από το τέλος του αρχείου)

Κλείσιμο αρχείων

- f.close()
  - Τόσο απλά!!
  - Όταν τελειώσουμε με το αρχείο εκτελούμε την παραπάνω εντολή για την αποδέσμευση μνήμης και την αποφυγή περίεργων καταστάσεων…

Exceptions I

- Κατά το άνοιγμα/κλείσιμο ή κατά την ανάγνωση/ εγγραφή ενός αρχείου οτιδήποτε μπορεί να πάει στραβά.. (“Νόμος του Murphy για το I/O.”)
- Χρησιμοποιούμε χειρισμό εξαιρέσεων για την αποφυγή “βίαιου” τερματισμού της εκτέλεσης του προγράμματός μας.
- Ο τύπος exception που γίνεται “throw” σε αυτές τις περιπτώσεις είναι ο IOError.

Exceptions II

- Γενική μορφή try block:
  - try:
    - ……
  - except IOError:
    - pass
  - finally:
    - ……

Exceptions III

- Παράδειγμα:
  - try:
    - f = open(path,mode)
    - f.readlines()
    - ……
  - except IOError:
    - pass
  - finally:
    - f.close()