Olga Ibáñez Solé
NetID: oibanez
N° de matricule : 000426526
INFO-101: Groupe 1

# Projet 3: gestion d'agenda

```
import sys
import codecs

"""
This program takes a calendar file (ics format) that includes some
events, and writes another file respecting the format but containing
only those files that meet any of a variable number of requirements
"""

fileIN = sys.argv[1] # Name of the input calendar file
fileOUT = sys.argv[2] # Name of the output calendar file
requirements = sys.argv[3:] # Remaining string arguments express
                            # requirements

fd = codecs.open(fileIN, "r", "utf-8")
fn = codecs.open(fileOUT, "w", "utf-8")

"""
This program converts each event into a dictionary with one entry per
attribute. This allows an easy checking of the requirements. All info in the
dictionary will be in lowercase to avoid case mismatching.
"""
def meets(req, dic):
  """
Receives a string expressing a requirement and a dictionary representing
an event. Returns True if the event meets the requirement and False
otherwise. It is assumed that the requirements consist of several atomic
subrequirements separated by " and ".
    """
    def test(subreq, dic):
       """
    Receives a string expressing an atomic subrequirement and a dictionary
    representing an event. Returns True if the event meets the
    subrequirement and False if it does not. It is assumed that the atomic
    subrequirement has the form Name:Value.
        """
```

Olga Ibáñez Solé
NetID: oibanez
N° de matricule : 000426526
INFO-101: Groupe 1

```
        subreq = subreq.lower().split(":")
        subreq = [subreq[0].strip(), subreq[1].strip()]
        result = False
        if subreq[0] in dic:
            if subreq[1] in dic[subreq[0]]:
                result = True
        return result

    req_list = req.split(" and ")
    i = 0
    while i < len(req_list) and test(req_list[i], dic):
        i += 1
    return(i == len(req_list))

# The header is processed and written in the output file

line = fd. readline()
head = ""
while line != "BEGIN:VEVENT\r\n" and line != "END:VCALENDAR\r\n":
    head += line
    line = fd.readline()
fn.write(head)

# Main loop: reads each event, checks if it meets any of the
# requirements and if so, writes it in the output file

while line != "END:VCALENDAR\r\n":
    event = line
    line = fd.readline()
    liste = []

    # Standard attributes (excluding "DESCRIPTION:") are processed and
    # included in liste in the form of pairs [Name, Value]:

    while "DESCRIPTION:" not in line:
        event += line
        liste.append(line.split(":"))
        line = fd.readline()
```

Olga Ibáñez Solé
NetID: oibanez
N° de matricule : 000426526
INFO-101: Groupe 1

```
    # Nonstandard attributes included in DESCRIPTION are
    # extracted and included similarly in liste:

    event += line
    newline = line.replace("DESCRIPTION:", "")
    liste_desc = newline.split("\\n")
    liste += [elem.split(": ") for elem in liste_desc]

    # Finally a dictionary whose keys are the names of the fields and whose
    # values are the information contained in those fields is created:

    dic = {elem[0].lower().strip():elem[1].lower().strip() for elem in
liste}

    while line != "END:VEVENT\r\n":
        line = fd.readline()
        event += line

    # The event is tested to see if it meets the requirements and if so
    # it is included in the output:

    i = 0
    while i < len(requirements) and meets(requirements[i], dic) == False:
        i += 1

    if i != len(requirements):
        fn.write(event)
    line = fd.readline()


# The tail of the calendar ("END:VCALENDAR") is written in the output:

fn.write(line)

fd.close()
fn.close()
```