MasterIT - Scripting Es3

Python

Sandro.Angius@lnf.infn.it

20/11/2002

easter.py

- Calcolo della data del giorno di Pasqua per gli anni compresi tra il 1582 e il 4200
- Viene utilizzato un algoritmo sviluppato Aloysius Lillius e da Christopher Clavius

```
#!/usr/bin/python

## This program calculates the date of Easter.
## Easter is the "first Sunday following the first full
## moon which occurs on or after March 21st".
## The algorithm is taken from Knuth's "Fundamental Algorithms",
## where in turn there is acknowledgement to the Neapolitan
## Aloysius Lillius and the German Jesuit mathematician
## Christopher Clavius.
## It applies for any year from 1582 to 4200.

import sys

def DateOfEaster(year):

    if year < 1582:
        return -1
    if year > 4200:
        return -2

    gold  = year % 19  + 1             # Golden Number
    cent  = year / 100 + 1             # Century
    offset= 3 * cent / 4 - 12          # Offset for Century but not leap year
    msynch= (8 * cent + 5) / 25 - 5    # Synch to Moon Orbit
    sun   = 5 * year / 4 - offset - 10 # Sundays...
    # "epact", i.e. full moon
    epact = (11 * gold + 20 + msynch - offset) % 30

    if epact < 0:
        epact = epact + 30
    if ((epact == 25) & (gold > 11)) | (epact == 24):
        epact = epact + 1
```

Easter.py - Pag. 1/3

```
full = 44 - epact
    if full < 21:
        full = full + 30
    full = full + 7 - (sun + full) % 7
    if full > 31:
        month = 4
        mday  = full - 31
    else:
        month = 3
        mday  = full

    return [month, mday]

def PrintUsage():
    print "Usage:", sys.argv[0], "<First_Year> [Last_Year]"
    sys.exit()

def CheckArg(numb):
    try:
        tmp = int(sys.argv[numb])
    except ValueError:
        print "\nThe year must be an integer number!!!\n"
        PrintUsage()
    except IndexError:
        return -1
    return tmp
```

Easter.py - Pag. 2/3

```
# Main

if (len(sys.argv) < 2) | (len(sys.argv) > 3):
    PrintUsage()

fyear = CheckArg(1)
lyear = CheckArg(2)

if lyear < fyear:
    lyear = fyear

Months = ["March", "April"]

for year in range(fyear, lyear + 1):

    Easter = DateOfEaster(year)

    if Easter == -1:
        print "Year must be greater than 1581, %d isn't !!!" % year
        sys.exit()
    elif Easter == -2:
        print "Year must be less than 4201, %d isn't !!!" % year
        sys.exit()
    else:
        print "%2d %s %d" %(Easter[1], Months[Easter[0] - 3], year)
```

Easter.py - Pag. 3/3

prime.py

– Scomposizione in fattori primi di un numero naturale.

```
#!/usr/local/bin/python

import sys
import math

def PrimesOfNum(numb, tlist=[]):
    plist = tlist[0:]
    maxtry = int(math.sqrt(numb)) + 1
    cnt    = 3L
    if numb % 2:
        while cnt < maxtry:
            if numb % cnt:
                cnt = cnt + 2
            else:
                plist.append(cnt)
                return PrimesOfNum(numb / cnt, plist)
        plist.append(numb)
        return plist
    else:
        plist.append(2L)
        if numb == 2:
            return plist
        else:
            return PrimesOfNum(numb / 2, plist)
```

Prime.py - Pag. 1/3

```
def FactorsOfNum(numb):
    resl = []
    lofp = PrimesOfNum(numb)
    lele = lpow = 0
    for ele in lofp:
        if ele <> lele:
            if lele:
                resl.append([ lele, lpow ])
            lele = ele
            lpow = 1L
        else:
            lpow = lpow + 1
    if lele:
        resl.append([ lele, lpow ])
    return resl

def PrintFactors(factl):
    sres = ''
    sep  = ''
    for fact in factl:
        base = str(fact[0])[:-1]
        if fact[1]>1:
            pow = '^' + str(fact[1])[:-1]
        else:
            pow = ''
        sres = sres + sep + base + pow
        sep  = ' * '
    return sres
```

Prime.py - Pag. 2/3

```
def PrintUsage():
    print "Usage:", sys.argv[0], "<Integer_Number> ..."
 sys.exit()

def CheckArg(numb):
    try:
        tmp = long(sys.argv[numb])
    except ValueError:
        print "\nPlease only numbers...\n"
        PrintUsage()
    if tmp < 1:
        print "\nPlease only numbers > 0 ...\n"
        PrintUsage()
    return tmp

# Main

if (len(sys.argv) < 2):
    PrintUsage()

for num in range(1,len(sys.argv)):
    oknum    = CheckArg(num)
    factors  = FactorsOfNum(oknum)
    print "%s = %s" % (str(oknum)[:-1], PrintFactors(factors))
```

Prime.py - Pag. 3/3

calturni.py

- Scrivere un programma che permetta la gestione di lavorazioni in turno su base settimanale.
- Input: per ogni giorno della settimana i nomi dei *"turnisti"*, mese, anno.
- Output: file di riepilogo con totale turni nel mese e nella settimana, totale turni per persona; file "scheda lavorazione" per ogni persona turnista.

Bibliografia

- http://www.python.org/doc/
- Knuth's “Fundamental Algorithms”
- Copia degli scripts si trova in:
  /afs/lnf.infn.it/project/master.it/doc/Scripting/EsScripts/