MediaWiki:IsbnBot.py


```
#coding=utf-8
import urllib, urllib2
import sys, time
import re
import cookielib
import xml.dom.minidom
from xml.dom.minidom import parse, parseString
import MySQLdb

##############
#            #
#  nastaveni #
#            #
##############
user_name = ''
password = ''
mysqltable = ''
mysqlserver = 'localhost'
mysqluser = ''
mysqlpwd = ''
mysqldb = ''
testurl = 'http://test-wiki.lf1.cuni.cz/api.php'
url = 'https://www.wikiskripta.eu/api.php'
cj = cookielib.CookieJar()

isbn = []
isbnCollector = []
incorrect = {}
unique = []
endash = []
citaceList = []
differ = 0


namespaces = {
    #-2: "Média",
    #-1: "Speciální",
    1: "Diskuse",
    2: "Uživatel",
    3: "Uživatel diskuse",
    4: "WikiSkripta",
    5: "WikiSkripta diskuse",
    6: "Soubor",
    7: "Soubor diskuse",
    8: "MediaWiki",
    9: "MediaWiki diskuse",
    10: "Šablona",
    11: "Šablona diskuse",
    12: "Nápověda",
    13: "Nápověda diskuse",
    14: "Kategorie",
    15: "Kategorie diskuse",
    100: "Portál",
    101: "Portál diskuse",
    102: "Fórum",
    103: "Fórum diskuse"
}

##############
#            #
#  funkce    #
#            #
##############
class LoginError(Exception):
    """Výjimka, jež bude vyvolána nepovedeným přihlášením k api serveru."""
    pass
class NoToken(Exception):
    """Výjimka, jež bude vyvolána, chybí-li token nutný k editaci."""
    pass
class EditError(Exception):
    """Výjimka, jež bude vyvolána nepovedenou editací."""
    pass




def fromUnicode(unicodeString):
    returnString = ""
    citliveZnaky = { 382: 'ž', 269: 'č', 283: 'ě', 237: 'í', 352: 'Š', 225: 'á', 345: 'ř', 353: 'š', 253: 'ý', 367: 'ů', 233: 'é', 381: 'Ž',
268: 'Č', 218: 'Ú', 250: 'ú', 357: 'ť', 271: 'ď', 328: 'ň', 243: 'ó', 8230: '…', 8222: '„', 8220: '“', 8722: '−', 318: 'ľ', 270: 'Ď', 244:
'ô', 154: '', 8211: '–', 327: 'Ň', 205: 'Í', 183: '·', 215: '×', 344: 'Ř', 9742: '☎', 9997: '✍', 322: 'ł', 232: 'è', 221: 'Ý', 8212: '—', 160:
' ', 167: '§', 61474: '', 252: 'ü', 177: '±', 945: 'α', 228: 'ä', 960: 'π', 246: 'ö', 946: 'β', 176: '°', 346: 'Ś', 282: 'Ě', 193: 'Á', 352:
'Š', 366: 'Ů', 180: '´', 8217: '’', 231: 'ç', 224: 'à', 201: 'É', 314: 'Ĺ' }
    czKeys = citliveZnaky.keys()
    for char in unicodeString:
        if (ord(char) in czKeys):
            returnString = returnString + citliveZnaky[ord(char)]
        else:
            returnString = returnString + str(char)
    return returnString

def toUnicode(string):
    """Přepíše do utf-8"""
    try:
        return unicode(string)
    except UnicodeDecodeError:
        returnString = unicode(string.encode("string_escape"))
```

```
        citliveZnaky = {
            u"\\xc3\\xa1": u"á",
            u"\\xc4\\x8d": u"č",
            u"\\xc4\\x8f": u"ď",
            u"\\xc3\\xa9": u"é",
            u"\\xc4\\x9b": u"ě",
            u"\\xc3\\xad": u"í",
            u"\\xc5\\x88": u"ň",
            u"\\xc3\\xb3": u"ó",
            u"\\xc5\\x99": u"ř",
            u"\\xc5\\xa1": u"š",
            u"\\xc5\\xa5": u"ť",
            u"\\xc3\\xba": u"ú",
            u"\\xc5\\xaf": u"ů",
            u"\\xc3\\xbd": u"ý",
            u"\\xc5\\xbe": u"ž"
        }
        for znak, nahrada in citliveZnaky.iteritems():
            returnString = returnString.replace(znak, nahrada)
        return returnString

def get_the_POST_page(values, apiserver = url):
    """Pošle požadavek html serveru 'apiserver' (implicitně wikiskripta) metodou POST s parametry 'values' typu dictionary a prečte si
odpověd, kterou vrátí. Pokud je třeba metodou POST poslat proměnné, které nemají hodnoty, využije se 'blank_values_list' typu list."""
    global cj
    try:
        data = urllib.urlencode(values)
        request = urllib2.Request(apiserver, data)
        opener = urllib2.build_opener(urllib2.HTTPCookieProcessor(cj))
        response = opener.open(request)
        #response = urllib2.urlopen(request)
        the_page = response.read()
        return the_page

    except urllib2.HTTPError as httperror:
        print "Chyba typu urllib2.HTTPError při požadavku na API o " + str(values) + "."
        print httperror
        return "<xml><edit result='error' /><error type='urllib2.HTTPError' /></xml>"
    except UnicodeError as ue:
        print "Chyba typu UnicodeError při požadavku na API o " + str(values) + "."
        print ue
        return "<xml><edit result='error' /><error type='UnicodeError' /></xml>"
    except urllib2.URLError as urlerror:
        print "Chyba typu urllib2.URLError při požadavku na API o " + str(values) + "."
        print urlerror
        return "<xml><edit result='error' /><error type='urllib2.URLError' /></xml>"

def get_time(n):
    """Prevede cas (vteriny od zacatku epochy) do formatu hh:mm:ss yy/mm/dd. Pokud je argumentem n konstanta None, vraci tak aktualni cas."""
    now = time.localtime(n)
    actual_time = str(now[3]) + ':' + str(now[4]) + ':' + str(now[5]) + ' ' + str(now[0]) + '/' + str(now[1]) + '/' + str(now[2])
    return actual_time
def get_human_readable_time():
    """Převede čas (vteřiny od začátku epochy) do formátu dd.mm.yyyy, hh:mm."""
    now = time.localtime()
    actual_time = str(now[2]) + "." + str(now[1]) + "." + str(now[0]) + ", " + str(now[3]) + ":" + str(now[4])
    return actual_time

def get_page_content(title):
    """Získá text wiki článku."""
    try:
        values = {"action": "query", "prop": "revisions", "rvprop": "content", "titles": title, "format": "xml"}
        the_page = get_the_POST_page(values)
    except UnicodeDecodeError as ude:
        try:
            values = {"action": "query", "prop": "revisions", "rvprop": "content", "titles": title.encode("utf-8"), "format": "xml"}
            the_page = get_the_POST_page(values)
        except UnicodeDecodeError as ude2:
            try:
                values = {"action": "query", "prop": "revisions", "rvprop": "content", "titles": title.decode("latin-2"), "format": "xml"}
                the_page = get_the_POST_page(values)
            except UnicodeEncodeError as uee:
                print "UnicodeEncodeError on get_page_content, title: "
                print title

    try:
        dom = parseString(the_page)
        if (dom.getElementsByTagName("rev") and dom.getElementsByTagName("rev")[0].childNodes):
            content = dom.getElementsByTagName("rev")[0].childNodes[0].nodeValue
            return content
        else:
            return ""
    except Exception as excp:
            print "Chyba při získávání textu článku '" + title + "'."
            print excp
            return ""
def login(login_name = user_name, pswd = password):
    """Přihlásí uživatele login_name (implicitně user_name) k api serveru. Pokud není přihlášení úspěšné, vyvolá výjimku LoginError."""
    values = {'action': 'login', 'lgname': login_name.encode("utf-8"), 'lgpassword': pswd.encode("utf-8"), 'format': 'xml'}
    the_page = get_the_POST_page(values)
    try:
        dom = parseString(the_page)
        login = dom.getElementsByTagName("login")[0]
        if (login.getAttribute("result") == "Success"):
            print "Přihlášení proběhlo úspěšně."
        elif (login.getAttribute("result") == "NeedToken"):
            values = {'action': 'login', 'lgname': login_name.encode("utf-8"), 'lgpassword': pswd.encode("utf-8"), 'lgtoken':
login.getAttribute("token"), 'format': 'xml'}
            the_page = get_the_POST_page(values)
            dom = parseString(the_page)
            login = dom.getElementsByTagName("login")[0]
            if (login.getAttribute("result") == "Success"):
                print "Přihlášení proběhlo úspěšně."
            else:
                print "Nebylo možné se přihlásit. Zkuste změnit user_name nebo password.\n"
                print the_page
                raise LoginError, "Nebylo možné se přihlásit."
        else:
            print "Nebylo možné se přihlásit. Zkuste změnit user_name nebo password.\n"
```

```
            print the_page
            raise LoginError, "Nebylo možné se přihlásit."
    except Exception as excp:
            print "Chyba typu " + str(type(excp)) + " při přihlašování."
            print excp
            raise LoginError, "Nebylo možné se přihlásit:"
def logout():
    """Odhlasi uzivatele od api serveru."""
    values = {'action': 'logout'}
    get_the_POST_page(values)
def list_all_pages(namespace_number, redirects = False):
    """Vrací seznam všech článků wikiskript v zadaném jmenném prostoru. Pokud jmenný prostor není zadán, hledá v hlavním jmenném prostoru."""
    namespaces = {
        #-2: "Média",
        #-1: "Speciální",
        1: "Diskuse",
        2: "Uživatel",
        3: "Uživatel diskuse",
        4: "WikiSkripta",
        5: "WikiSkripta diskuse",
        6: "Soubor",
        7: "Soubor diskuse",
        8: "MediaWiki",
        9: "MediaWiki diskuse",
        10: "Šablona",
        11: "Šablona diskuse",
        12: "Nápověda",
        13: "Nápověda diskuse",
        14: "Kategorie",
        15: "Kategorie diskuse",
        100: "Portál",
        101: "Portál diskuse",
        102: "Fórum",
        103: "Fórum diskuse"
    }

    if redirects:
        if namespace_number:
            print "Probíhá vyhledávání ve jmenném prostoru " + namespaces[namespace_number]
            values = {'action': 'query', 'list': 'allpages', 'apnamespace': namespace_number, 'aplimit': '500', 'apfilterredir': 'all'}
        else:
            print "Probíhá vyhledávání v hlavním jmenném prostoru."
            values = {'action': 'query', 'list': 'allpages', 'aplimit': '500', 'apfilterredir': 'all'}
    else:
        if namespace_number:
            print "Probíhá vyhledávání ve jmenném prostoru " + namespaces[namespace_number]
            values = {'action': 'query', 'list': 'allpages', 'apnamespace': namespace_number, 'aplimit': '500', 'apfilterredir':
'nonredirects'}
        else:
            print "Probíhá vyhledávání v hlavním jmenném prostoru."
            values = {'action': 'query', 'list': 'allpages', 'aplimit': '500', 'apfilterredir': 'nonredirects'}
    the_page = get_the_POST_page(values)
    get_continues = re.compile('(?<=apfrom=&quot;).*(?=quot;)', re.UNICODE)
    continues = get_continues.findall(the_page)
    pages = the_page
    while continues:
        if namespace_number:
            values = {'action': 'query', 'list': 'allpages', 'apnamespace': namespace_number, 'aplimit': '500', 'apfrom': continues[0],
'apfilterredir': 'nonredirects'}
        else:
            values = {'action': 'query', 'list': 'allpages', 'aplimit': '500', 'apfrom': continues[0], 'apfilterredir': 'nonredirects'}
        the_page = get_the_POST_page(values)
        continues = get_continues.findall(the_page)
        pages = pages + "\n\n" + the_page
    get_titles = re.compile('(?<=title=&quot;).*(?=&quot;)', re.UNICODE)
    titles = get_titles.findall(pages)
    return titles

def get_isbn(content, title):
    global incorrect
    global citaceList

    parametry = {
        'korporace': u"\| ?korporace ?= ?(.*?)\n? ?\|",
        'prijmeni1': u"\| ?příjmení1 ?= ?(.*?)\n? ?\|",
        'jmeno1': u"\| ?jméno1 ?= ?(.*?)\n? ?\|",
        'prijmeni2': u"\| ?příjmení2 ?= ?(.*?)\n? ?\|",
        'jmeno2': u"\| ?jméno2 ?= ?(.*?)\n? ?\|",
        'prijmeni3': u"\| ?příjmení3 ?= ?(.*?)\n? ?\|",
        'jmeno3': u"\| ?jméno3 ?= ?(.*?)\n? ?\|",
        'kolektiv': u"\| ?kolektiv ?= ?((ano|ne))\n? ?\|",
        'titul': u"\| ?titul ?= ?(.*?)\n? ?\|",
        'podnazev': u"\| ?podnázev ?= ?(.*?)\n? ?\|",
        'url': u"\| ?url ?= ?(.*?)\n? ?\|",
        'vydani': u"\| ?vydání ?= ?(.*?)\n? ?\|",
        'misto': u"\| ?místo ?= ?(.*?)\n? ?\|",
        'vydavatel': u"\| ?vydavatel ?= ?(.*?)\n? ?\|",
        'rok': "\| u?rok ?= ?([0-9]{4})\n? ?\|",
        'rozsah' : u"\| ?rozsah ?= ?([0-9]*)\n? ?\|",
        'edice': u"\| ?edice ?= ?(.*?)\n? ?\|",
        'svazek': u"\| ?svazek ?= ?(.*?)\n? ?\|"
    }

    citacniSablona = re.compile("{{Citace[^}]*?isbn[^}]*?}}", re.UNICODE)
    isbnVSablone = re.compile("\|\s?isbn\s?=\s*(?P<isbn_number>[-0123456789Xx]*).*", re.UNICODE)
    isbn10 = re.compile("[0123456789Xx]{10}|[-0123456789Xx]{13}", re.UNICODE)
    isbn13 = re.compile("[0123456789Xx]{13}|[-0123456789Xx]{17}", re.UNICODE)

    for sablona in citacniSablona.findall(content):
        vSablone = isbnVSablone.search(sablona)
        if (vSablone):
            thisISBN = vSablone.group('isbn_number')
            thisISBNCislo = jen_cislo(thisISBN)
            if (valid_isbn(thisISBNCislo)):
                citace = {}
                citace["isbn"] = thisISBN
                citace["niceISBN"] = thisISBNCislo
                for parametr in parametry.keys():
                    parametrRe = re.compile(parametry[parametr], re.UNICODE)
```

```
                    parametrSearch = parametrRe.search(sablona)
                    if (parametrSearch):
                        citace[parametr] = parametrSearch.group(1).strip()
                citaceList.append(citace)
            else:
                if (thisISBN != "-"):
                    incorrect[title] = thisISBN

def jen_cislo(isbn):
    """Z řetězce vybere jen čísla a znaky X nebo x."""
    returnValue = ''
    for char in isbn:
        asciiord = ord(char)
        if ((asciiord >= 48 and asciiord <= 57) or char == 'x' or char == 'X'):
            returnValue = returnValue + char
    return returnValue

def hodnota(char):
    """Hodnota čísla jako znaku. X nebo x dává hodnotu 10."""
    asciiord = ord(char)
    if (asciiord >= 48 and asciiord <= 57):
        return (asciiord - 48)
    elif (char == 'x' or char == 'X'):
        return 10
    else:
        return 0

def valid_isbn(isbnSeZnaky):
    """Zjistí, zdali isbn může existovat."""

    isbn = jen_cislo(isbnSeZnaky)
    if (len(isbn) == 10):
        #číslice z ISBN jako vektor
        vektorISBN = [hodnota(isbn[x]) for x in range(10)]
        #váhy jednotlivých číslic jako vektor
        vektorValidace = [ 10, 9, 8, 7, 6, 5, 4, 3, 2, 1 ]
        #vynásob vektory po prvcích
        vazenyVektor = [vektorISBN[x] * vektorValidace[x] for x in range(10)]
        #suma všech vážených cifer (skalární součin s jednotkovým vektorem)
        suma = sum(vazenyVektor)
        if ((suma % 11) == 0):
            return True
        else:
            return False
    elif (len(isbn) == 13):
        for char in isbn:
            if (char == 'x' or char == 'X'):
                return False

        #číslice z ISBN jako vektor
        vektorISBN = [hodnota(isbn[x]) for x in range(13)]
        #váhy jednotlivých číslic jako vektor
        vektorValidace = [ 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1, 3, 1]
        #vynásob vektory po prvcích
        vazenyVektor = [vektorISBN[x] * vektorValidace[x] for x in range(13)]
        suma = sum(vazenyVektor)
        if ((suma % 10) == 0):
            return True
        else:
            return False
    else:
        return False

#################
#               #
# telo programu #
#               #
#################
login()
all_pages = list_all_pages(None)
for page in all_pages:
    content = get_page_content(page)
    get_isbn(content, page)
logout()

counter = 0
citaceUniqueList = {}
for citace in citaceList:
    if (citace["niceISBN"] in citaceUniqueList.keys()):
        if (citace != citaceUniqueList[citace["niceISBN"]]):
            counter = counter + 1
            """
            print "Zaznam: "
            print citaceUniqueList[citace["niceISBN"]]
            print "Novy: "
            print citace
            if (raw_input("Nahradit záznam? (y/cokoliv)") == "y"):
                citaceUniqueList[citace["niceISBN"]] = citace
            """
    else:
        citaceUniqueList[citace["niceISBN"]] = citace

print "Celkem: " + str(len(citaceList))
print "Unikatni: " + str(len(citaceUniqueList))
print "Lisi se: " + str(counter)
f = open("output.txt", "w")
db = MySQLdb.connect(mysqlserver, mysqluser, mysqlpwd, mysqldb)
c = db.cursor()
f.write("INSERT INTO `" + mysqldb +"`.`" + mysqltable + "` (`isbnid`, `isbn`, `korporace`, `prijmeni1`, `jmeno1`, `prijmeni2`, `jmeno2`,
`prijmeni3`, `jmeno3`, `kolektiv`, `titul`, `podnazev`, `url`, `vydani`, `misto`, `vydavatel`, `rok`, `rozsah`, `edice`, `svazek`) VALUES\n")
for key in citaceUniqueList.keys():
    citace = citaceUniqueList[key]
    isbn = citace["isbn"]
    niceISBN = citace["niceISBN"]
    if ("korporace" in citace.keys()): korporace = citace["korporace"]
    else: korporace = ""
    if ("prijmeni1" in citace.keys()): prijmeni1 = citace["prijmeni1"]
    else: prijmeni1 = ""
    if ("jmeno1" in citace.keys()): jmeno1 = citace["jmeno1"]
```

```
    else: jmeno1 = ""
    if ("prijmeni2" in citace.keys()): prijmeni2 = citace["prijmeni2"]
    else: prijmeni2 = ""
    if ("jmeno2" in citace.keys()): jmeno2 = citace["jmeno2"]
    else: jmeno2 = ""
    if ("prijmeni3" in citace.keys()): prijmeni3 = citace["prijmeni3"]
    else: prijmeni3 = ""
    if ("jmeno3" in citace.keys()): jmeno3 = citace["jmeno3"]
    else: jmeno3 = ""
    if ("kolektiv" in citace.keys()):
        if (citace["kolektiv"] == "ano"):
            kolektiv = u"1"
        else:
            kolektiv = u"0"
    else: kolektiv = u"0"
    if ("titul" in citace.keys()): titul = citace["titul"]
    else: titul = ""
    if ("podnazev" in citace.keys()): podnazev = citace["podnazev"]
    else: podnazev = ""
    if ("url" in citace.keys()): url = citace["url"]
    else: url = ""
    if ("vydani" in citace.keys()): vydani = citace["vydani"]
    else: vydani = ""
    if ("misto" in citace.keys()): misto = citace["misto"]
    else: misto = ""
    if ("vydavatel" in citace.keys()): vydavatel = citace["vydavatel"]
    else: vydavatel = "vydavatel"
    if ("rok" in citace.keys()): rok = citace["rok"]
    else: rok = ""
    if ("rozsah" in citace.keys()): rozsah = citace["rozsah"]
    else: rozsah = ""
    if ("edice" in citace.keys()): edice = citace["edice"]
    else: edice = ""
    if ("svazek" in citace.keys()): svazek = citace["svazek"]
    else: svazek = ""

    parametry = [niceISBN, isbn, korporace, prijmeni1, jmeno1, prijmeni2, jmeno2, prijmeni3, jmeno3, kolektiv, titul, podnazev, url, vydani,
misto, vydavatel, rok, rozsah, edice, svazek]
    parametry = [parametr.replace("'","\\'") for parametr in parametry]
    parametry = [parametr.replace("\"","\\\"") for parametr in parametry]
    #parametry = [unicode(parametr, "utf-8") for parametr in parametry]
    outputline = "('" + "', '". join(parametry) + "'),\n"
    sql = "INSERT INTO `" + mysqldb + "`.`" + mysqltable + "` (`key`, `isbnid`, `isbn`, `korporace`, `prijmeni1`, `jmeno1`, `prijmeni2`,
`jmeno2`, `prijmeni3`, `jmeno3`, `kolektiv`, `titul`, `podnazev`, `url`, `vydani`, `misto`, `vydavatel`, `rok`, `rozsah`, `edice`, `svazek`)
VALUES\n"
    sql =  sql + "('', '" + "', '". join(parametry) + "');"
    f.write(outputline.encode("utf-8"))
    c.execute(sql.encode("utf-8"))

print "Unikatnich ISBN: " + str(len(unique)) + " z " + str(len(isbn) - len(incorrect))
print "Chybnych ISBN: " + str(len(incorrect))
for key in incorrect.keys():
    print "* [[" + fromUnicode(key) + "]]: " + fromUnicode(incorrect[key])
print "ENDASH:"
for dash in endash:
    print "* [[" + dash + "]]"
```