Autor: Paweł Salwa

W tym zadaniu tylko zaimplementowałem metodę potęgową dla znalezienia najmniejszej wartości własnej.
Korzystam z algorytmu thomasa, ponieważ mamy trójdiagonalną macierz.

Output:
Odwrotna metoda potegowa-
najmniejszy eignvalue wynosi: 0.00040076683257458435

Kod w pythonie (Salwa9.py):

```
# -*- coding: utf-8 -*-
"""
Created on Sun Feb 05 17:54:55 2017

@author: salwus
"""
from math import cosh,sqrt

N=1000

def vectorInit():
	global N

	h=20./(N-1)
	a=1./(h**2)
	b=2./(h**2)

	A=[]#diagonale
	B=[]
	C=[]

	A.append(0.)#indeks za macierza

	for i in range(0,N):
		x = (i-N/2)*h
		x = cosh(x)
		x = x**2
		if(i > 0):
			A.append(-a)
		B.append(b + 6/x)
		if(i< N-1):
			C.append(-a)

	C.append(0.)#indeks za macierza
	return A,B,C
#============================================================
def thomas(A,B,C,D):
```

```
	global N

	N -= 1

	C[0] /= B[0]
	D[0] /= B[0]

	for i in range(1,N):
		C[i] /= B[i] - A[i]*C[i-1]
		D[i] = (D[i] - A[i]*D[i-1]) / (B[i] - A[i]*C[i-1])

	D[N] = (D[N] - A[N]*D[N-1]) / (B[N] - A[N]*C[N-1])


	i = N - 1
	while i >= 0:
		D[i] -= C[i] * D[i+1]
		i -= 1
	return A,B,C
#=============================================================

def omp():
	global N
	A,B,C = vectorInit()
	Z = [1.,2.,3.,4.,5.,6.,7.]

	Lambda=0
	temp2=[]
	Evector=[]
	X=[]
	X_prev=[]
	Z=[]
	for i in range(0,N):
		Evector.append(0.)
		temp2.append(0.)
		X.append(1.)
		X_prev.append(1.)
		Z.append(0.)


	while(True):
		A,B,C = thomas(A,B,C,X)
		temp=0
		for i in range(0,N):
			temp += Evector[i]*X[i]
			temp2[i] = Evector[i]

		for i in range(0,N):
			temp2[i] *= temp

		for i in range(0,N):
			X[i] -= temp2[i]
```

```
            for i in range(0,N):
                Z[i] = X[i]

            X_norm = 0

            for i in range(0,N):
                X_norm += X[i]*X[i]
            X_norm = sqrt(X_norm)

            for i in range(0,N):
                X[i] /= X_norm
            if (abs( Lambda- X_norm ) < 0.00000001):
                break

            for i in range(0,N):
                X_prev[i] = X[i]
            Lambda = X_norm
        X_norm = 0
        for i in range(0,N):
            X_norm += Z[i]*X_prev[i]
        Eigenvalue = X_norm

        for i in range(0,N):
            Evector[i] = X[i]
            X_prev[i] = 1
            X[i] = 1

        print "Odwrotna metoda potegowa-\nnajmniejszy eignvalue wynosi: " + repr(Eigenvalue)


omp()
```