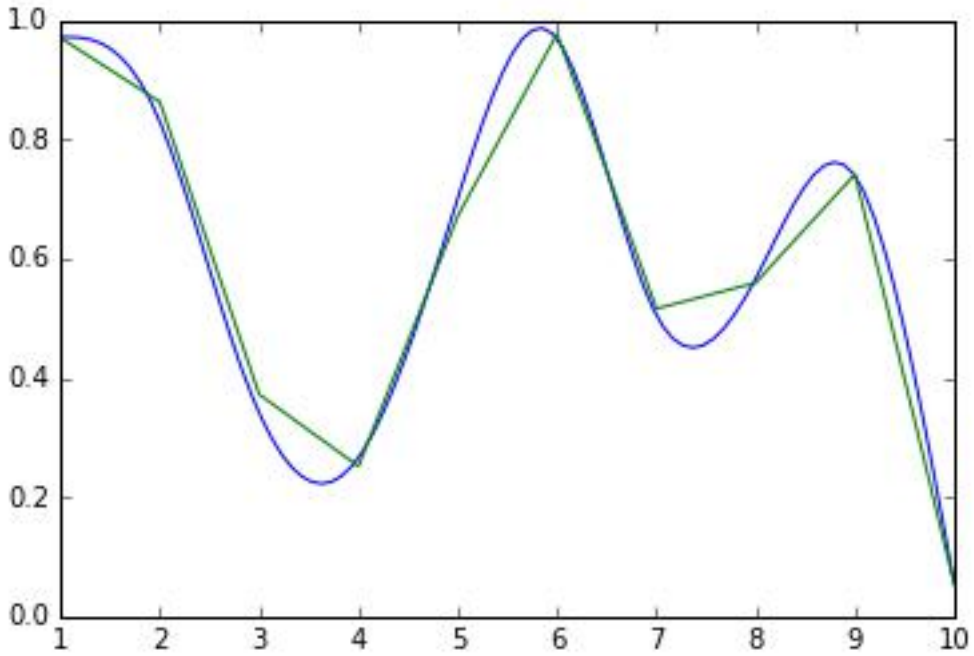
KRZYSZTOF SOKÓŁ-SZOŁTYSEK
PROGRAM 6
GRUPA PONIEDZIAŁKOWA

Program interpoluje wartości funkcji(losując pierwsze 10) metoda splajnów kubicznych. Na początku zostają wypisane wylosowane liczby na podstawie których będzie odbywał się proces interpolacji. Warto zwrócić uwagę na postać macierzy która jest potrzebna do obliczeń - w przypadku równoodległych węzłow interpolacji przybiera zawsze tą samą postać przedstawioną poniżej. Program rysuje także wykres na którym widzimy punkty wylosowane(zielona linia) oraz interpolację(niebieska).

PRZYKŁADOWY OUTPUT

Wylosowane liczby :
[0.97001143 0.86261256 0.37211152 0.25172791 0.67127012
0.976899 0.51594927 0.55931996 0.74125188 0.05439145]
[[4 1 0 0 0 0 0 0]
 [1 4 1 0 0 0 0 0]
 [0 1 4 1 0 0 0 0]
 [0 0 1 4 1 0 0 0]
 [0 0 0 1 4 1 0 0]
 [0 0 0 0 1 4 1 0]
 [0 0 0 0 0 1 4 1]
 [0 0 0 0 0 0 1 4]]



LISTING

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Sat Dec 24 16:00:18 2016

@author: dm
"""

import numpy as np
import matplotlib as plt

N = 10
Y = np.random.random(N)
print('Wylosowane liczby : ', Y)
X = list(range(N))
#print(X)
T = np.copy(X)
H = [T[i+1] - T[i] for i in range(N-1)]
#print(H)
B = [(1 / H[i]) * (Y[i+1] - Y[i]) for i in range(N-1)]

V = [2 * (H[i-1] + H[i]) for i in range(1, N-1)]

U = [6 * (B[i] - B[i-1]) for i in range(1, N-1)]

Vtri = [V[i] for i in V]
Htri = [H[i+1] for i in range(7)]
Tri = np.diag(Vtri) + np.diag(Htri, 1) + np.diag(Htri, -1)
print(Tri)

ZTri = np.linalg.solve(Tri, U)

Z = np.insert(ZTri,0,0)
Z = np.append(Z, 0)

num = 0
wyniki = []
#wyniki2 = []
#wyniki3 = []
#wyniki4 = []
```

```
def S(i, num):
    return (Z[i+1])/(6 * H[i]) * np.power((num - T[i]), 3 ) + (Z[i])/(6 * H[i]) *
np.power((T[i+1] - num), 3) + ((Y[i+1])/(H[i]) - (Z[i+1])/(6) * H[i]) * (num - T[i])
+ ((Y[i])/(H[i]) - (H[i])/(6) * (Z[i])) * (T[i+1] - num )

#S = [(Z[i+1])/(6 * H[i]) * np.power((num - T[i]), 3 ) + (Z[i])/(6 * H[i]) *
np.power((T[i+1] - num), 3) + ((Y[i+1])/(H[i]) - (Z[i+1])/(6) * H[i]) * (num - T[i])
+ ((Y[i])/(H[i]) - (H[i])/(6) * (Z[i])) * (T[i+1] - num ) for i in range(N-1)]



for i in range(0, 9):
     for j in range(1, 11):
          wyniki.append(S(i, i + j*0.1))
          #print(i,i + j*0.1)




          #np.linspace(i,i+1,10))
#for i in range(0, 9):
#      for j in range(1, 6):
#           wyniki2.append(S(i, i + j*0.2))
#
#
#
#
#for i in range(0, 9):
#      for j in range(1, 3):
#           wyniki3.append(S(i, i + j*0.5))
#
#
#for i in range(0, 9):
#      for j in range(1, 2):
#           wyniki4.append(S(i, i + j))

#print(wyniki)
lgth = len(wyniki)
#print(lgth)
#lgth2 =len(wyniki2)
#print(lgth2)
#lgth3 = len(wyniki3)
#print(lgth3)
#lgth4 = len(wyniki4)
```

```
#print(lgth4)

num = np.linspace(1,10,lgth)
#num2 = np.linspace(1,10,lgth2)
#num3 = np.linspace(1,10,lgth3)
#num4 = np.linspace(1,10,lgth4)
plt.pyplot.plot(num, wyniki)
#plt.pyplot.plot(num2, wyniki2)
#plt.pyplot.plot(num3, wyniki3)
#plt.pyplot.plot(num4, wyniki4)
plt.pyplot.plot(np.linspace(1,10,10), Y, linestyle='-')




plt.pyplot.show()
```