МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное бюджетное образовательное учреждение
высшего профессионально образования
“Московский авиационный институт (национальный исследовательский университет)”

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И ПРИКЛАДНОЙ МАТЕМАТИКИ

Направление подготовки Фундаментальная информатика и информационные технологии

Курсовой проект №1

по курсу:
Дискретная математика

на тему:
Построение BC-дерева на основе глубинных номеров вершин графа

Работу выполнил
студент 1 курса
бакалавриата
очного отделения
Группы М8О-110Б
Чернова Ольга Николаевна

Научный руководитель
_____
Алексеев Н. С.

Подпись студента, число

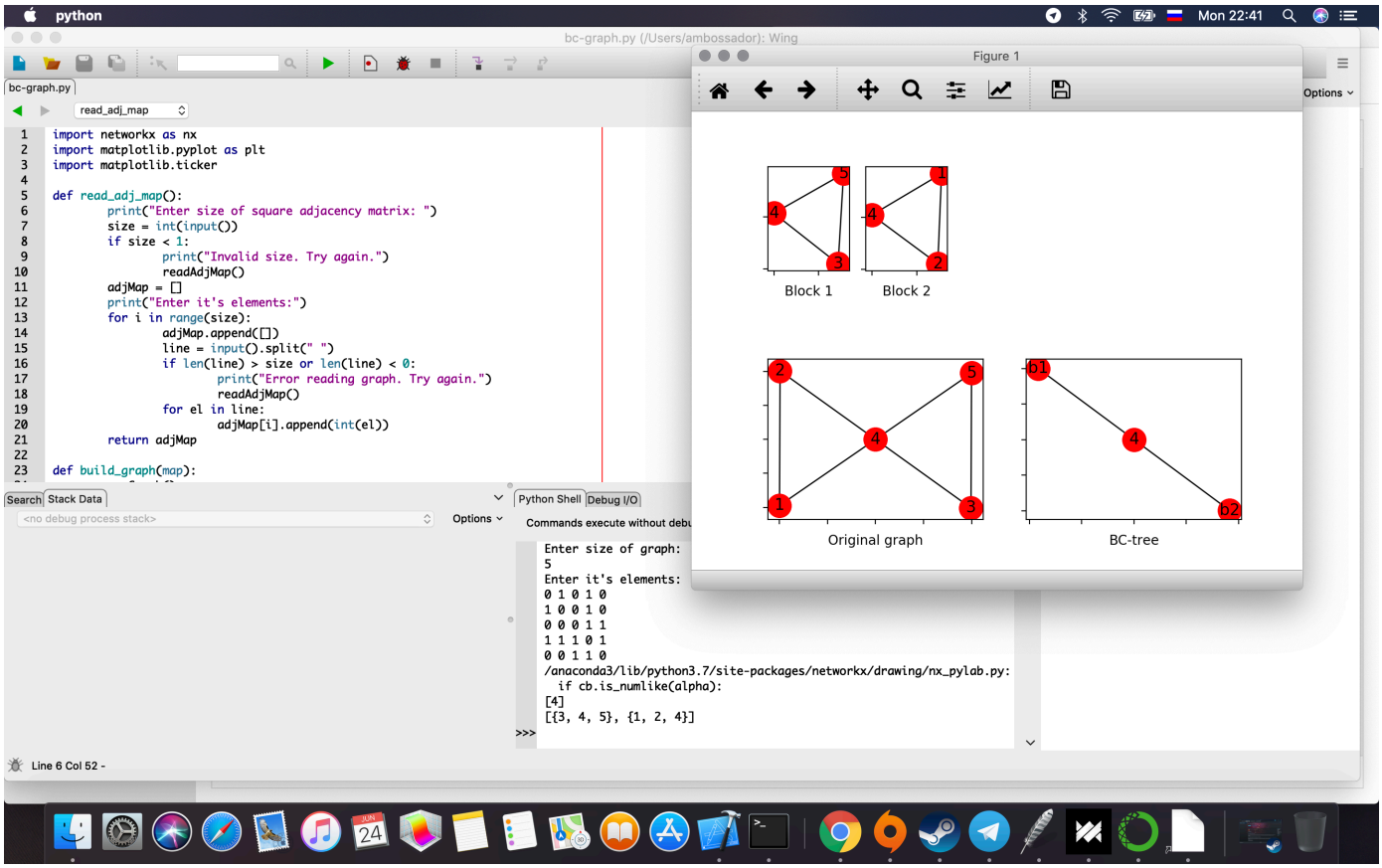
Москва 2018 год

bc-graph.py

```
import networkx as nx
import matplotlib.pyplot as plt
import matplotlib.ticker

def read_adj_map():
     print "Enter size of square adjacency matrix: "
     size = int(input())
     if size < 1:
          print "Invalid size. Try again."
          readAdjMap()
     adjMap = []
     print "Enter it's elements:"
     for i in range(size):
          adjMap.append([])
          line = input().split(" ")
          if len(line) > size or len(line) < 0:
               print "Error reading graph. Try again."
               readAdjMap()
          for el in line:
               adjMap[i].append(int(el))
     return adjMap

def build_graph(map):
    g = nx.Graph()
    for idx1, vert in enumerate(map):
        for idx2, edge in enumerate(vert):
            if edge > 0:
                g.add_edge(idx1+1, idx2+1)
    return

map = read_adj_map()

g = build_graph(map)

formatter = matplotlib.ticker.NullFormatter()

axes = plt.subplot(223)
axes.xaxis.set_major_formatter (formatter)
axes.yaxis.set_major_formatter (formatter)
axes.set_xlabel("Original graph")
nx.draw_networkx(g)


art_points = list(nx.articulation_points(g))
bicomponents = list(nx.biconnected_components(g))
print(art_points)
print(bicomponents)

axes = plt.subplot(224)
```

```
axes.xaxis.set_major_formatter (formatter)
axes.yaxis.set_major_formatter (formatter)
bc_tree = nx.Graph()
for block in bicomponents:
    for art in art_points:
        if art in block:
            s = 'b' + str(bicomponents.index(block) + 1)
            bc_tree.add_edge(s, art)
axes.set_xlabel("BC-tree")
nx.draw_networkx(bc_tree)

num = 351
for block in bicomponents:
    tempg = g.subgraph(list(block))
    axes = plt.subplot(num)
    axes.xaxis.set_major_formatter (formatter)
    axes.yaxis.set_major_formatter (formatter)
    axes.set_xlabel("Block " + str(num % 10))
    num += 1
    nx.draw_networkx(tempg)

plt.show()
```

Общий вид:



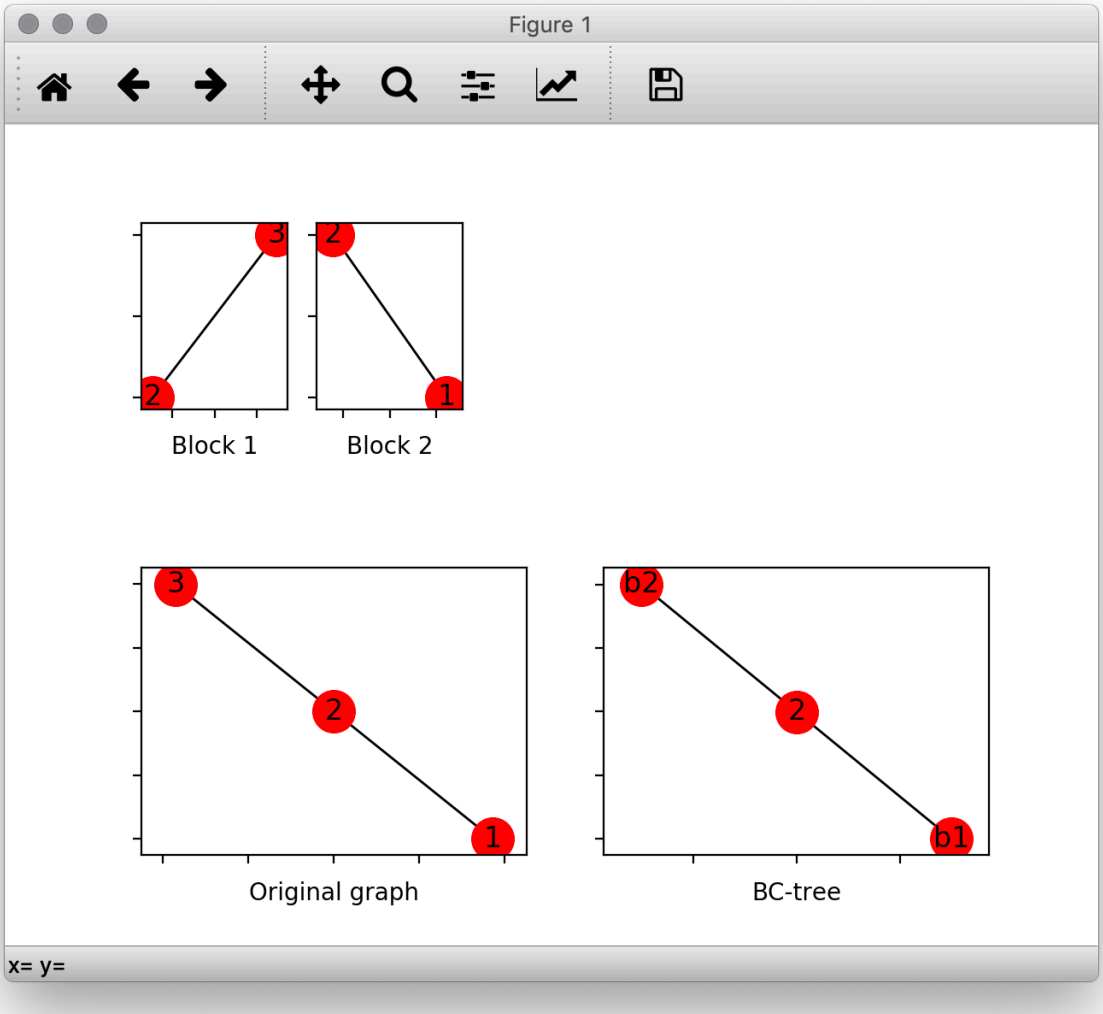
Москва 2018 год

Примеры работы программы:

Входные данные:

```
Enter size of square adjacency matrix:
3
Enter it's elements:
0 1 0
1 0 1
0 1 0
```

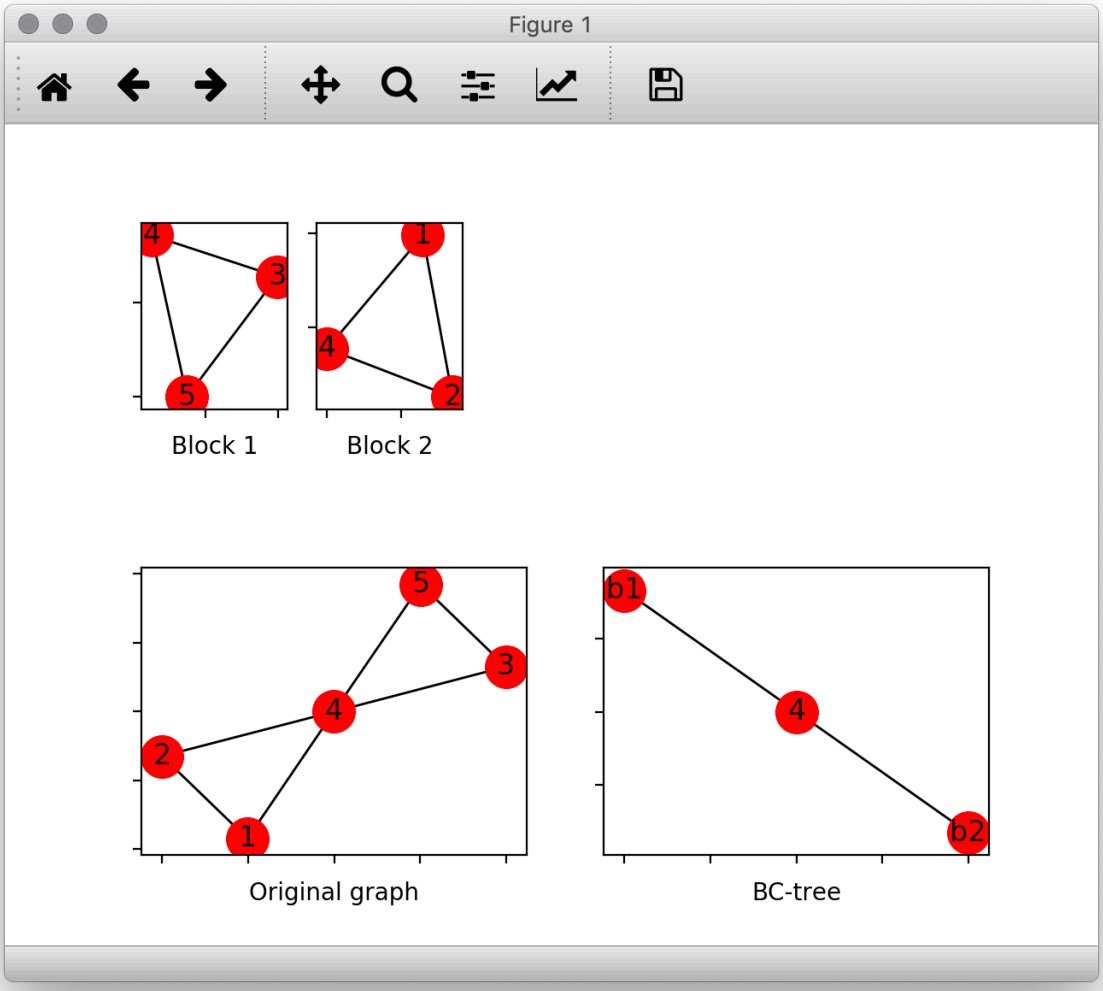
Выходные данные:



Входные данные:

```
Enter size of square adjacency matrix:
5
Enter it's elements:
0 1 0 1 0
1 0 0 1 0
0 0 0 1 1
1 1 1 0 1
0 0 1 1 0
```

Выходные данные:



Входные данные:

```
Enter size of square adjacency matrix:
5
Enter it's elements:
0 0 1 0 0
0 1 0 1 1
0 1 0 1 1
0 1 1 0 1
0 0 1 1 0
```

Выходные данные: