Учреждение образования

«Белорусский государственный университет информатики и радиоэлектроники»

Кафедра информатики

Отчёт

Лабораторная работа №5

По учебной дисциплине *Методы оптимизации и управления*

Вариант 1

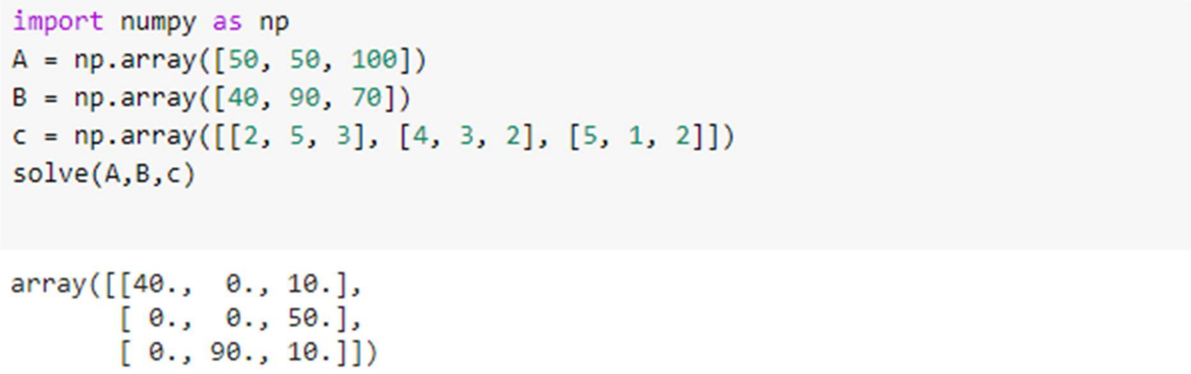
Выполнил:	Проверил:
студент группы №853504	доцент кафедры информатики
Кузьма В.В.	Дугинов О.И.

Минск 2021

ЗАДАНИЕ

Решить матричную транспортную задачу методом потенциалов.
Вариант 16 % 15 = 1.

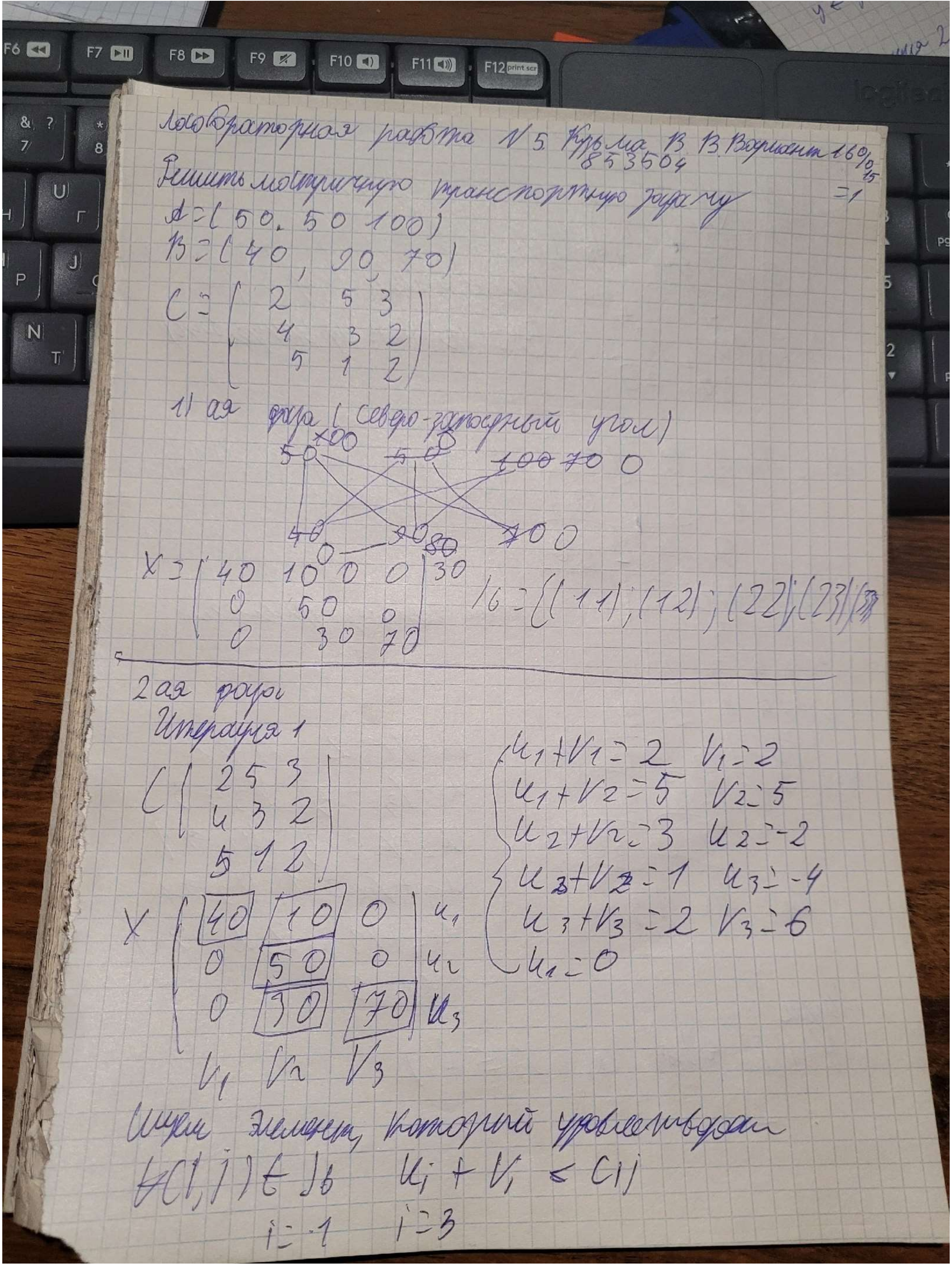
Примеры работы



```
import numpy as np
A = np.array([50, 50, 100])
B = np.array([40, 90, 70])
c = np.array([[2, 5, 3], [4, 3, 2], [5, 1, 2]])
solve(A,B,c)
```

```
array([[40.,  0., 10.],
       [ 0.,  0., 50.],
       [ 0., 90., 10.]])
```

ПОЭТАПНАЯ РЕАЛИЗАЦИЯ



Лабораторная работа № 5. Кузьма В. В. Вариант 16%15 = 1
853504

Решить матричную транспортную задачу

$A = (50, 50\ 100)$

$B = (40, 90, 70)$

$$C = \begin{pmatrix} 2 & 5 & 3 \\ 4 & 3 & 2 \\ 5 & 1 & 2 \end{pmatrix}$$

1) 1ая фаза (северо-западный угол)

50 ~~100~~
~~50~~ 0
~~100~~ ~~70~~ 0
~~40~~ 0
~~90~~ ~~80~~
~~70~~ 0 0

$$X = \begin{pmatrix} 40 & 10 & 0 & 0 \\ 0 & 50 & 0 & \\ 0 & 30 & 70 & \end{pmatrix} 30$$

$J_б = \{(1\,1), (1\,2); (2\,2); (2\,3)(3$

2ая фаза

Итерация 1

$$C \begin{pmatrix} 2 & 5 & 3 \\ 4 & 3 & 2 \\ 5 & 1 & 2 \end{pmatrix}$$

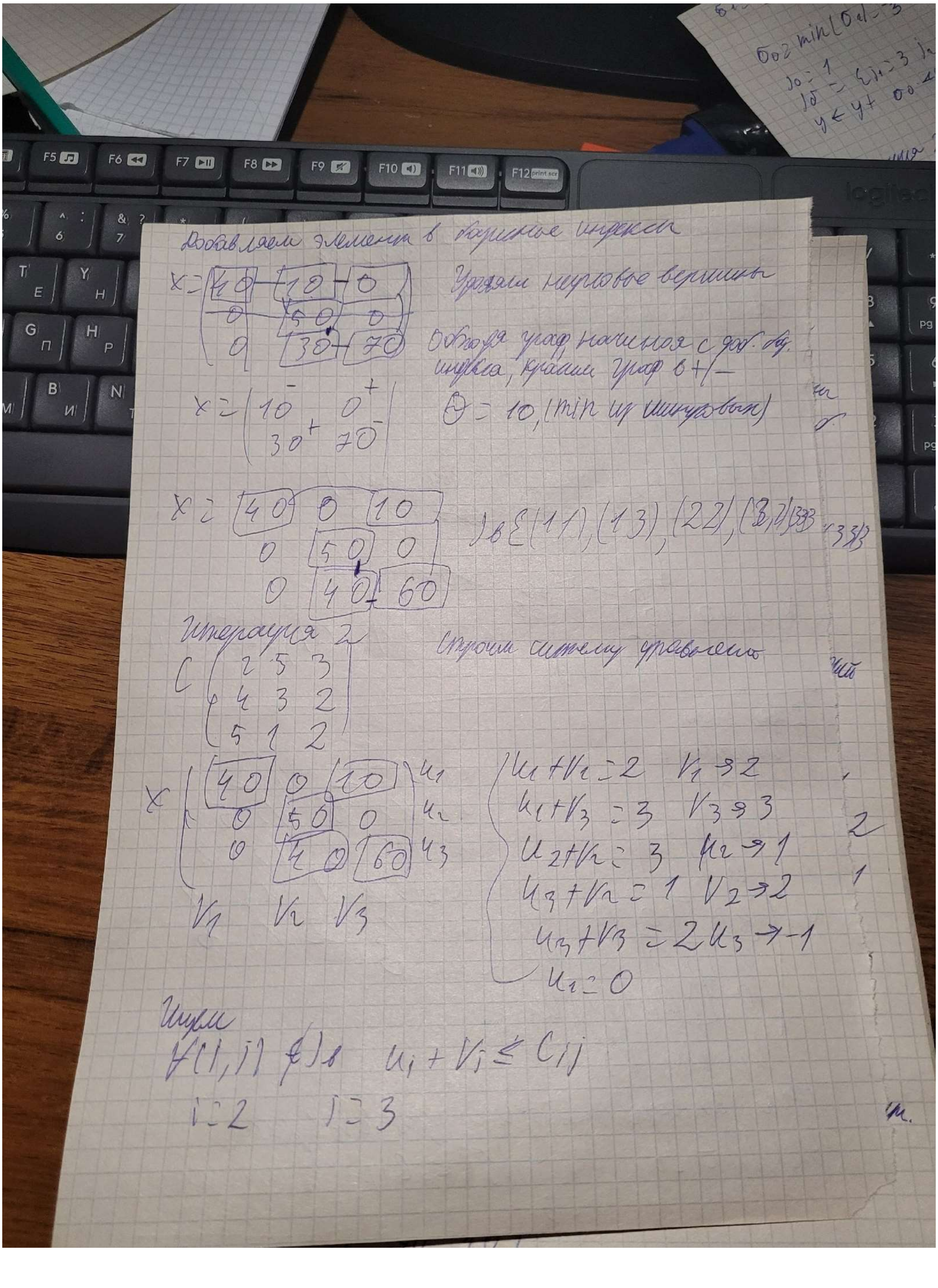
$$X \begin{array}{cccc} \boxed{40} & \boxed{10} & 0 & u_1 \\ 0 & \boxed{50} & 0 & u_2 \\ 0 & \boxed{30} & \boxed{70} & u_3 \\ v_1 & v_2 & v_3 & \end{array}$$

$$\begin{cases} u_1 + v_1 = 2 & v_1 = 2 \\ u_1 + v_2 = 5 & v_2 = 5 \\ u_2 + v_2 = 3 & u_2 = -2 \\ u_3 + v_2 = 1 & u_3 = -4 \\ u_3 + v_3 = 2 & v_3 = 6 \\ u_1 = 0 & \end{cases}$$

Ищем элемент, который удовлетворяет

$\forall (i, j) \in J_б \quad u_i + v_i \le c_{ij}$

$i = 1 \quad j = 3$



Добавляем элемент в базисные индексы

$$X = \begin{pmatrix} \boxed{40} & \boxed{10} & \boxed{0} \\ 0 & \boxed{50} & \boxed{0} \\ 0 & \boxed{30} & \boxed{70} \end{pmatrix}$$

Удаляем нулевые вершины

Обходя граф начиная с доб. эл. индекса, красим граф в +/−

$$X = \begin{pmatrix} 10^{-} & 0^{+} \\ 30^{+} & 70^{-} \end{pmatrix}$$

$\Theta = 10$, (min из минусовых)

$$X = \begin{pmatrix} \boxed{40} & 0 & \boxed{10} \\ 0 & \boxed{50} & 0 \\ 0 & \boxed{40} & \boxed{60} \end{pmatrix}$$

$J_б \in \{(1\,1), (1\,3), (2\,2), (3,2)(3\,3)\}$

Итерация 2

Строим систему уравнений

$$C \begin{pmatrix} 2 & 5 & 3 \\ 4 & 3 & 2 \\ 5 & 1 & 2 \end{pmatrix}$$

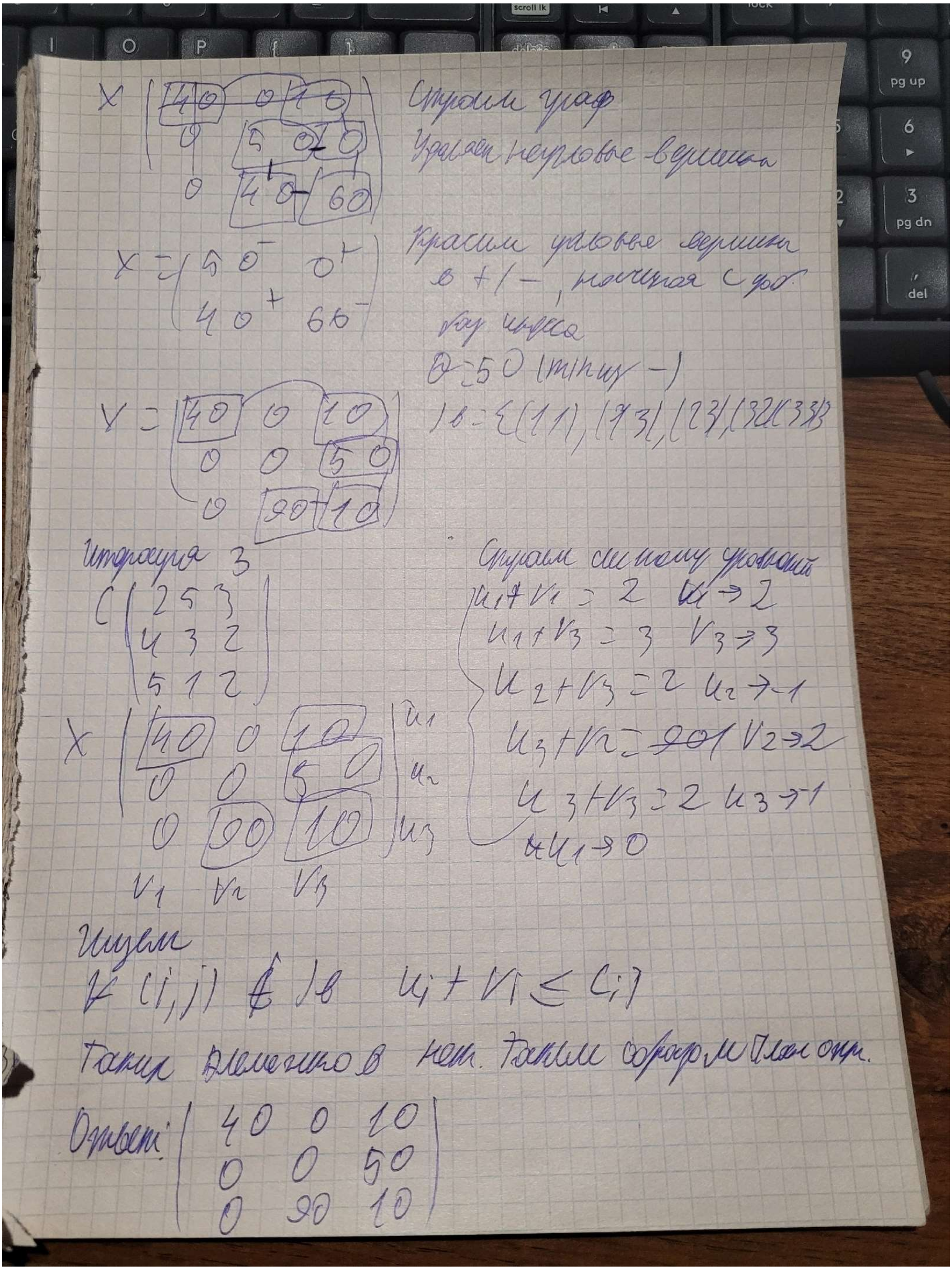
$$X \begin{array}{cccc} \boxed{40} & 0 & \boxed{10} & u_1 \\ 0 & \boxed{50} & 0 & u_2 \\ 0 & \boxed{40} & \boxed{60} & u_3 \\ V_1 & V_2 & V_3 & \end{array}$$

$$\begin{cases} u_1 + V_1 = 2 & V_1 \to 2 \\ u_1 + V_3 = 3 & V_3 \to 3 \\ u_2 + V_2 = 3 & u_2 \to 1 \\ u_3 + V_2 = 1 & V_2 \to 2 \\ u_3 + V_3 = 2 & u_3 \to -1 \\ u_1 = 0 & \end{cases}$$

Итер

$\forall (i,j) \notin J_б \quad u_i + V_j \leq C_{ij}$

$i = 2 \qquad j = 3$



$X \begin{pmatrix} \boxed{40} & 0 & \boxed{10} \\ 0 & \boxed{50} & \boxed{0} \\ 0 & \boxed{40} & \boxed{60} \end{pmatrix}$

Строим граф

Удаляем неугловые вершины

$X = \begin{pmatrix} 50^- & 0^+ \\ 40^+ & 60^- \end{pmatrix}$

Красим угловые вершины в +/−, начиная с угл. вер. индекса

$\theta = 50$ (min из −)

$J_б = \{(1\,1), (1\,3), (2\,3), (3\,2)(3\,3)\}$

$Y = \begin{pmatrix} \boxed{40} & 0 & \boxed{10} \\ 0 & 0 & \boxed{50} \\ 0 & \boxed{90} & \boxed{10} \end{pmatrix}$

Итерация 3

$C \begin{pmatrix} 2 & 5 & 3 \\ 4 & 3 & 2 \\ 5 & 1 & 2 \end{pmatrix}$

$X \begin{array}{c|c} \begin{pmatrix} \boxed{40} & 0 & \boxed{10} \\ 0 & 0 & \boxed{50} \\ 0 & \boxed{90} & \boxed{10} \end{pmatrix} & \begin{array}{c} u_1 \\ u_2 \\ u_3 \end{array} \\ \begin{array}{ccc} v_1 & v_2 & v_3 \end{array} & \end{array}$

Строим систему уравнений

$$\begin{cases} u_1 + v_1 = 2 & v_1 \to 2 \\ u_1 + v_3 = 3 & v_3 \to 3 \\ u_2 + v_3 = 2 & u_2 \to -1 \\ u_3 + v_2 = \cancel{90}\,1 & v_2 \to 2 \\ u_3 + v_3 = 2 & u_3 \to -1 \end{cases}$$

$u_1 \to 0$

Ищем

$\forall (i,j) \notin J_б \quad u_i + v_j \le c_{ij}$

Таких элементов нет. Таким образом План опт.

Ответ: $\begin{pmatrix} 40 & 0 & 10 \\ 0 & 0 & 50 \\ 0 & 90 & 10 \end{pmatrix}$

КОД ПРОГРАММЫ

```
def generator(n, m):
  for i in range(n):
    for j in range(m):
      yield i, j
def solve(A, B, c):
  n = len(A)
```

```
m = len(B)
X = np.zeros((n, m))
aa = np.copy(A)
bb = np.copy(B)
i, j = 0, 0
Jb = np.empty((0, 2), int)
while True:
  X[i][j] = min(aa[i], bb[j])
  aa[i] -= X[i][j]
  bb[j] -= X[i][j]
  Jb = np.append(Jb, np.array([[i, j]]), axis = 0)
  if aa[i] == 0:
    i += 1
  if bb[j] == 0:
    j += 1
  if i == n or j == m:
    break
while True:
  cb = []
  for i in Jb:
    cb.append(c[i[0]][i[1]])
  u = np.zeros(n)
  v = np.zeros(m)
  ram = np.zeros((2, max(n, m)))
  eq = [[f'u{i[0]}', f'v{i[1]}'] for i in Jb]
  queue = ['u0']
  ram[0][0] = 1
  while len(queue) > 0:
    now = queue[0]
    queue.pop(0)
    k = 0 if now[0] == 'u' else 1
    index = int(now[1])
    numb = u[index] if k == 0 else v[index]
    for i in range(len(Jb)):
      if eq[i][k] != now:
        continue
      if ram[(k + 1) % 2][int(eq[i][(k + 1) % 2][1])] == 0:
        queue.append(eq[i][(k + 1) % 2])
      ram[(k + 1) % 2][int(eq[i][(k + 1) % 2][1])] = 1
      new_numb = cb[i] - numb
      if k == 0:
        v[int(eq[i][(k + 1) % 2][1])] = new_numb
      else:
        u[int(eq[i][(k + 1) % 2][1])] = new_numb
  ri, rj = -5, -5
  for i, j in generator(n,m):
    k = np.array([i, j])
    if not any(np.equal(Jb,k).all(1)):
      if u[i]+v[j] >c[i][j]:
        ri, rj = i, j
        break
  if ri == -5 and rj == -5:
    return X
  Jb = np.append(Jb, [[ri, rj]], axis = 0)
```

```
    q = np.copy(Jb)
    while True:
      flx, fly = 1, 1
      for i in range(n):
        count = 0
        for k in q:
          if k[0] == i:
            count += 1
        if count < 2:
          j = 0
          while j < len(q):
            if q[j][0] == i:
              q = np.delete(q, j, axis = 0)
              flx = 0
            else:
              j += 1
      for i in range(n):
        count = 0
        for k in q:
          if k[1] == i:
            count += 1
        if count < 2:
          j = 0
          while j < len(q):
            if q[j][1] == i:
              q = np.delete(q, j, axis = 0)
              fly = 0
            else:
              j += 1
      if flx == 1 and fly == 1:
        break
    graph = np.full((len(q), len(q)), 0)
    for i in range(len(graph)):
      for j in range(len(graph)):
        if i == j:
          graph[i][j] = 0
          continue
        for k in range(len(graph)):
          if k == i or k == j:
            continue
          if q[i][0] == q[j][0] and q[k][1] in range(min(q[i][1], q[j][1]
), max(q[j][1], q[i][1]) + 1):
            graph[i][j] = 1
          elif q[i][1] == q[j][1] and q[k][0] in range(min(q[i][0],q[j][0
]), max(q[i][0], q[i][0]) + 1):
            graph[i][j] = 1
    queue = [len(q) - 1]
    pm = np.array([2 for _ in range(len(q))])
    pm[-1] = 1
    while len(queue) > 0:
      x = queue[0]
      queue.pop(0)
      for j in range(len(q)):
        if graph[x][j] == 1 and pm[j] == 2:
```

```
      queue.append(j)
      pm[j] = (pm[x] + 1) % 2
tetta = 999 ** 999
di, dj = 9999, 9999
for i in range(len(q)):
  if pm[i] == 0 and X[q[i][0]][q[i][1]] < tetta:
    tetta = X[q[i][0]][q[i][1]]
    ti = q[i][0]
    tj = q[i][1]
for i in range(len(q)):
  if pm[i] == 1:
    X[q[i][0]][q[i][1]] += tetta
  else:
    X[q[i][0]][q[i][1]] -= tetta
for i in range(len(Jb)):
  if Jb[i][0] == ti and Jb[i][1] == tj:
    Jb = np.delete(Jb, i, 0)
    break
```