1. Запрограммировать точную постановку для multidimensional multiway NPP с критерием оптимизации : минимизация максимальной разности по компоненте (по координате) между суммарными характеристиками групп (подмножеств)[сформулирована в классе] . Найти решение для случайно сгенерированных данных

```
In[∘]:= n = 8;(*количество векторов*)
NC = 4;(*размерность векторов*)
k = 3;(*количество групп*)

In[∘]:= initialData1 = RandomInteger[{0, 20}, {n, NC}]

Out[∘]= {{1, 0, 15, 5}, {20, 13, 2, 7}, {0, 5, 7, 13}, {17, 18, 5, 12},
 {6, 4, 17, 3}, {11, 20, 6, 7}, {10, 4, 1, 11}, {8, 7, 1, 3}}

In[∘]:= varsX = Array[x, {n, k}];
vars = Join[Flatten@varsX, {delta}];

In[∘]:= objFun = delta;

In[∘]:= c = Last@CoefficientArrays[objFun, vars];

In[∘]:= con1 = Total[varsX, {2}]; (*первое условие*)
rhs1 = ConstantArray[{1, 0}, n];

In[∘]:=
 (*= - '0', ≥ - '1', ≤ - '-1'*)
J1J2 = Subsets[Range[k], {2}]; (*варианты пар номеров групп, где j1<j2*)
listL = {};
listi = {};
For[l = 1, l ≤ NC, l++,
  For[j = 1, j ≤ Length[J1J2], j++,
   listi = {};
   For[i = 1, i ≤ n, i++, AppendTo[listi,
     Transpose[initialData1]〚l〛〚i〛 * (varsX〚i〛〚J1J2〚j〛〚1〛〛 - varsX〚i〛〚J1J2〚j〛〚2〛〛)]];
   AppendTo[listL, delta - Total@listi];
  ]
 ];

con2 = listL;(*второе условие*)
rhs2 = ConstantArray[{0, 1}, Length@listL];
```

In[]:=
```
listL = {};
listi = {};
For[l = 1, l ≤ NC, l++,
  For[j = 1, j ≤ Length[J1J2], j++,
   listi = {};
   For[i = 1, i ≤ n, i++, AppendTo[listi,
     Transpose[initialData1][[l]][[i]] * (varsX[[i]][[J1J2[[j]][[1]]]] - varsX[[i]][[J1J2[[j]][[2]]]])]];
   AppendTo[listL, delta + Total@listi];
  ]
 ];

con3 = listL;(*третье условие*)
rhs3 = ConstantArray[{0, 1}, Length@listL];
```

In[]:=
```
lu = Join[ConstantArray[{0, 1}, n * k], ConstantArray[{0, Total@Total@initialData1}, 1]];
domain = Join[ConstantArray[Integers, n * k], ConstantArray[Reals, 1]];
m = Last@CoefficientArrays[Join[con1, con2, con3], vars];
```

In[]:=
```
sol = LinearProgramming[c, m, Join[rhs1, rhs2, rhs3], lu, domain]
```

LinearProgramming: Warning: integer linear programming will use a machine−precision approximation of the inputs.

Out[]=
```
{0, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 12.}
```

In[]:=
```
partition = Pick[initialData1, #, 1] & /@
  Transpose[Partition[sol[[ ;; -2]], k]] (*полученное разбиение векторов на группы*)
```

Out[]=
```
{{{17, 18, 5, 12}, {6, 4, 17, 3}}, {{1, 0, 15, 5}, {11, 20, 6, 7}, {10, 4, 1, 11}},
 {{20, 13, 2, 7}, {0, 5, 7, 13}, {8, 7, 1, 3}}}
```

2. Сформулировать и запрограммировать задачу для multidimensional multiway NPP с критерием оптимизации : минимизации взвешенной суммы относительных отклонений суммарных характеристик групп от идеальных значений по координатам . Найти решение для случайно сгенерированных данных .

In[]:=
```
n = 8;(*количество векторов*)
NC = 4;(*размерность векторов*)
k = 3;(*количество групп*)
```

In[]:=
```
weights2 = Normalize[RandomReal[{1, 10}, NC], Total]
initialData2 = RandomInteger[{1, 20}, {n, NC}]
ideal2 = Total[N@initialData2] / k
```

Out[]=
```
{0.0846376, 0.158391, 0.48882, 0.268151}
```

Out[]=
```
{{10, 10, 11, 13}, {13, 13, 12, 17}, {3, 7, 13, 9}, {12, 1, 11, 8},
 {11, 2, 15, 9}, {17, 12, 8, 14}, {18, 16, 2, 5}, {8, 2, 20, 20}}
```

Out[]=
```
{30.6667, 21., 30.6667, 31.6667}
```

In[]:=
```
varsX = Array[x, {n, k}];
varsDelta = Array[delta, {NC, k}];
vars = Join[Flatten@varsX, Flatten@varsDelta];
```

```
In[]:= listFunk = {};
For[l = 1, l ≤ NC, l++,
 For[j = 1, j ≤ k, j++,
  AppendTo[listFunk, weights2[[l]] * delta[l, j]]]]

In[]:= objFun = Total@listFunk;

In[]:= c = Last@CoefficientArrays[objFun, vars];

In[]:= con1 = Total[varsX, {2}] ;(*первое условие*)
rhs1 = ConstantArray[{1, 0}, n];

In[]:=
(*= - '0', ≥ - '1', ≤ - '-1'*)
listY = {};
For[l = 1, l ≤ NC, l++,
 listl = {};
 For[j = 1, j ≤ k, j++,
  AppendTo[listl,
   delta[l, j] + Dot[Transpose[varsX][[j]], Transpose[initialData2][[l]] / ideal2[[l]]]];
 AppendTo[listY, listl]]

con2 = Flatten@listY;(*второе условие*)
rhs2 = ConstantArray[{1, 1}, NC * k];

In[]:= listY = {};
For[l = 1, l ≤ NC, l++,
 listl = {};
 For[j = 1, j ≤ k, j++,
  AppendTo[listl,
   -delta[l, j] + Dot[Transpose[varsX][[j]], Transpose[initialData2][[l]] / ideal2[[l]]]];
 AppendTo[listY, listl]]

con3 = Flatten@listY;(*третье условие*)
rhs3 = ConstantArray[{1, -1}, NC * k];

In[]:= lu  =  Join[ConstantArray[{0, 1}, n * k], ConstantArray[{0, 1}, NC * k]];
domain = Join[ConstantArray[Integers, n * k],  ConstantArray[Reals, NC * k]];
m = Last@CoefficientArrays[Join[con1, con2, con3], vars];

In[]:= sol = LinearProgramming[c, m, Join[rhs1, rhs2, rhs3], lu, domain]

Out[]= {1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1,
 1, 0, 0, 0.413043, 0.0434783, 0.369565, 0.428571, 0.047619, 0.47619,
 0.0108696, 0.0434783, 0.0543478, 0.0421053, 0.0210526, 0.0210526}
```

In[ ]:=
```
partition = Pick[initialData2, #, 1] & /@
  Transpose[Partition[sol[[;; -11]], k]] (*полученное разбиение векторов на группы*)
```

Out[ ]=
```
{{{10, 10, 11, 13}, {8, 2, 20, 20}}, {{3, 7, 13, 9}, {12, 1, 11, 8}, {17, 12, 8, 14}},
 {{13, 13, 12, 17}, {11, 2, 15, 9}, {18, 16, 2, 5}}}
```

In[ ]:=

3. Сформулировать и запрограммировать задачу для multidimensional multiway NPP с критерием оптимизации : минимизации взвешенной суммы относительных отклонений суммарных характеристик групп от идеальных значений по координатам для каждой группы . Найти решение для случайно сгенерированных данных .

In[ ]:=
```
n = 8; (*количество векторов*)
NC = 4; (*размерность векторов*)
k = 3; (*количество групп*)
(*исходные данные для задачи*)
weights3 = Normalize[RandomReal[{1, 10}, NC], Total]
initialData3 = RandomInteger[{1, 20}, {n, k, NC}]
ideals3 = Total[N@initialData3^T, {2}] / k
```

Out[ ]=
```
{0.127431, 0.158629, 0.312464, 0.401476}
```

Out[ ]=
```
{{{20, 16, 15, 1}, {20, 17, 10, 19}, {8, 7, 8, 6}},
 {{9, 9, 3, 9}, {12, 13, 1, 5}, {10, 19, 4, 19}},
 {{3, 10, 16, 5}, {4, 17, 13, 8}, {14, 19, 5, 16}},
 {{1, 10, 4, 3}, {5, 19, 6, 8}, {7, 4, 16, 3}}, {{11, 4, 2, 4}, {3, 7, 3, 19}, {12, 2, 5, 10}},
 {{3, 10, 15, 19}, {11, 3, 19, 17}, {3, 16, 10, 11}},
 {{2, 11, 11, 3}, {4, 3, 4, 15}, {7, 6, 17, 4}},
 {{4, 18, 11, 12}, {12, 1, 7, 1}, {20, 9, 15, 19}}}
```

Out[ ]=
```
{{17.6667, 29.3333, 25.6667, 18.6667},
 {23.6667, 26.6667, 21., 30.6667}, {27., 27.3333, 26.6667, 29.3333}}
```

In[ ]:=
```
varsX = Array[x, {n, k}];
varsDelta = Array[delta, {NC, k}];
vars = Join[Flatten@varsX, Flatten@varsDelta];
```

In[ ]:=
```
listFunk = {};
For[l = 1, l ≤ NC, l++,
 For[j = 1, j ≤ k, j++,
  AppendTo[listFunk, weights3[[l]] * delta[l, j]]]]
```

In[ ]:=
```
objFun = Total@listFunk;
```

In[ ]:=
```
c = Last@CoefficientArrays[objFun, vars];
```

In[ ]:=
```
con1 = Total[varsX, {2}] ; (*первое условие*)
rhs1 = ConstantArray[{1, 0}, n];
```

```
In[•]:= listY = {};
For[l = 1, l ≤ NC, l++,
 listl = {};
 For[j = 1, j ≤ k, j++,
  AppendTo[listl, delta[l, j] + Dot[Transpose[varsX][[j]],
      Transpose[Transpose[initialData3][[j]]][[l]] / ideals3[[j]][[l]]]];
 AppendTo[listY, listl]]

con2 = Flatten@listY;(*второе условие*)
rhs2 = ConstantArray[{1, 1}, NC * k];

In[•]:= listY = {};
For[l = 1, l ≤ NC, l++,
 listl = {};
 For[j = 1, j ≤ k, j++,
  AppendTo[listl, -delta[l, j] + Dot[Transpose[varsX][[j]],
      Transpose[Transpose[initialData3][[j]]][[l]] / ideals3[[j]][[l]]]];
 AppendTo[listY, listl]]

con3 = Flatten@listY;(*третье условие*)
rhs3 = ConstantArray[{1, -1}, NC * k];

In[•]:= lu = Join[ConstantArray[{0, 1}, n * k], ConstantArray[{0, 1}, NC * k]];
domain = Join[ConstantArray[Integers, n * k], ConstantArray[Reals, NC * k]];
m = Last@CoefficientArrays[Join[con1, con2, con3], vars];

In[•]:= sol = LinearProgramming[c, m, Join[rhs1, rhs2, rhs3], lu, domain]

Out[•]= {0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0,
  0, 0, 1, 0.301887, 0.0140845, 0.148148, 0.159091, 0.275, 0.0853659,
  0.220779, 0.0952381, 0.0625, 0.0178571, 0.119565, 0.0227273}

In[•]:= partition = Pick[initialData3, #, 1] & /@ Transpose[Partition[sol[[ ;; -11]], k]]

Out[•]= {{{{9, 9, 3, 9}, {12, 13, 1, 5}, {10, 19, 4, 19}},
   {{1, 10, 4, 3}, {5, 19, 6, 8}, {7, 4, 16, 3}}, {{11, 4, 2, 4}, {3, 7, 3, 19}, {12, 2, 5, 10}},
   {{2, 11, 11, 3}, {4, 3, 4, 15}, {7, 6, 17, 4}}},
  {{{20, 16, 15, 1}, {20, 17, 10, 19}, {8, 7, 8, 6}},
   {{3, 10, 16, 5}, {4, 17, 13, 8}, {14, 19, 5, 16}}},
  {{{3, 10, 15, 19}, {11, 3, 19, 17}, {3, 16, 10, 11}},
   {{4, 18, 11, 12}, {12, 1, 7, 1}, {20, 9, 15, 19}}}}
```

In[ ]:=
```
result = {};
For[i = 1, i ≤ Length[partition], i++,
 group = {};
 For[j = 1, j ≤ Length[partition⟦i⟧], j++, AppendTo[group, partition⟦i, j, i⟧]];
 AppendTo[result, group]
]
result
```

Out[ ]=
```
{{{9, 9, 3, 9}, {1, 10, 4, 3}, {11, 4, 2, 4}, {2, 11, 11, 3}},
 {{20, 17, 10, 19}, {4, 17, 13, 8}}, {{3, 16, 10, 11}, {20, 9, 15, 19}}}
```