1. Техническое задание

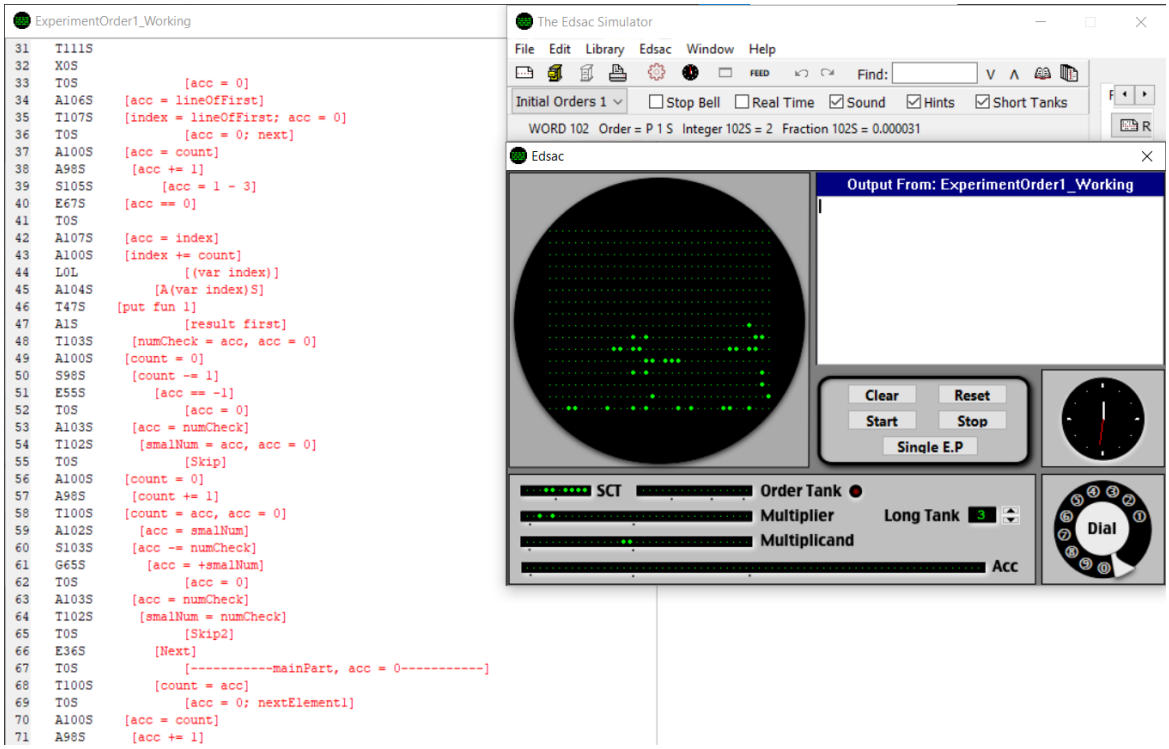
Реализовать нахождение наибольшего общего делителя (НОД) для массива чисел при помощи EDSAC.

2. Метод решения

Идея алгоритма заключается в том, что мы вычитаем из меньшее числа меньше и заменяем первое на их разность до тех пор, пока их разность не станет равна нулю. В таком случае уменьшаемое и вычитаемое как раз и будут искомым числом.

- A = 10; B= 2
    - A-B = 8
    - A-B = 6
    - A-B = 4
    - A-B = 2
    - A-B = 0
- Ответ = 2

3. Программа Orders 1



ExperimentOrder1_Working

```
31   T111S
32   X0S
33   T0S              [acc = 0]
34   A106S    [acc = lineOfFirst]
35   T107S    [index = lineOfFirst; acc = 0]
36   T0S              [acc = 0; next]
37   A100S    [acc = count]
38   A98S      [acc += 1]
39   S105S         [acc = 1 - 3]
40   E67S     [acc == 0]
41   T0S
42   A107S    [acc = index]
43   A100S    [index += count]
44   L0L              [(var index)]
45   A104S         [A(var index)S]
46   T47S    [put fun 1]
47   A1S              [result first]
48   T103S     [numCheck = acc, acc = 0]
49   A100S    [count = 0]
50   S98S      [count -= 1]
51   E55S         [acc == -1]
52   T0S              [acc = 0]
53   A103S     [acc = numCheck]
54   T102S      [smalNum = acc, acc = 0]
55   T0S              [Skip]
56   A100S    [count = 0]
57   A98S      [count += 1]
58   T100S    [count = acc, acc = 0]
59   A102S      [acc = smalNum]
60   S103S     [acc -= numCheck]
61   G65S        [acc = +smalNum]
62   T0S              [acc = 0]
63   A103S     [acc = numCheck]
64   T102S      [smalNum = numCheck]
65   T0S              [Skip2]
66   E36S        [Next]
67   T0S              [----------mainPart, acc = 0----------]
68   T100S       [count = acc]
69   T0S              [acc = 0; nextElement1]
70   A100S    [acc = count]
71   A98S      [acc += 1]
```

The Edsac Simulator

File Edit Library Edsac Window Help

Find:

Initial Orders 1 | Stop Bell | Real Time | Sound | Hints | Short Tanks

WORD 102 Order = P 1 S Integer 102S = 2 Fraction 102S = 0.000031

Edsac

Output From: ExperimentOrder1_Working

Clear Reset Start Stop Single E.P

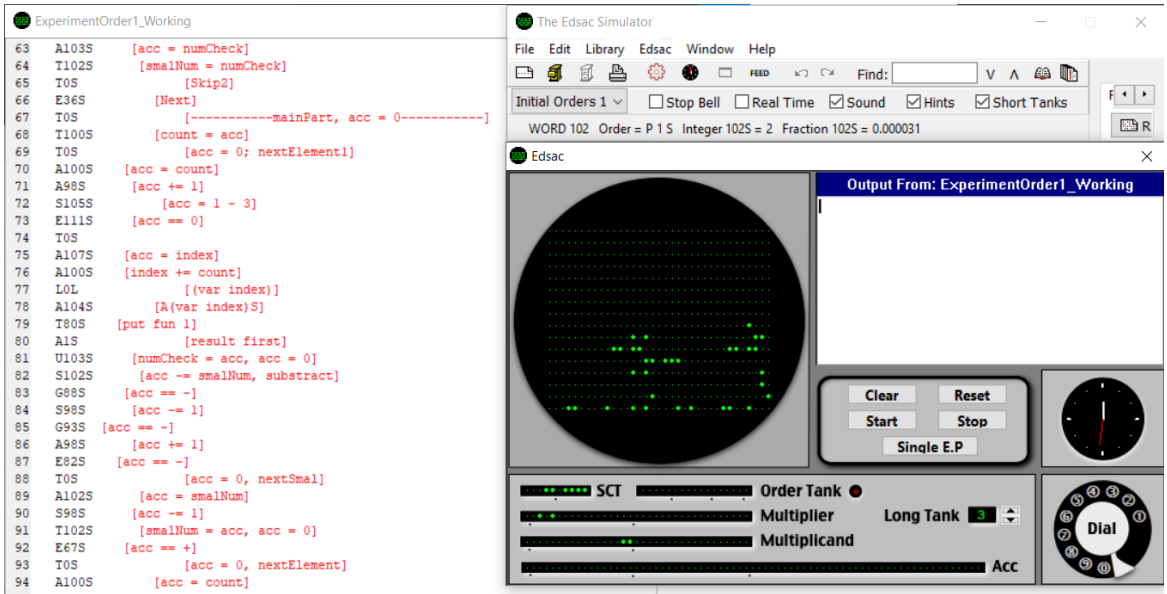
SCT Order Tank

Multiplier Long Tank 3

Multiplicand

Acc

Dial



ExperimentOrder1_Working

```
63   A103S     [acc = numCheck]
64   T102S      [smalNum = numCheck]
65   T0S              [Skip2]
66   E36S        [Next]
67   T0S              [----------mainPart, acc = 0----------]
68   T100S       [count = acc]
69   T0S              [acc = 0; nextElement1]
70   A100S    [acc = count]
71   A98S      [acc += 1]
72   S105S        [acc = 1 - 3]
73   E111S     [acc == 0]
74   T0S
75   A107S    [acc = index]
76   A100S    [index += count]
77   L0L              [(var index)]
78   A104S         [A(var index)S]
79   T80S    [put fun 1]
80   A1S              [result first]
81   U103S     [numCheck = acc, acc = 0]
82   S102S      [acc -= smalNum, substract]
83   G88S     [acc == -]
84   S98S      [acc -= 1]
85   G93S  [acc == -]
86   A98S      [acc += 1]
87   E82S    [acc == -]
88   T0S              [acc = 0, nextSmal]
89   A102S      [acc = smalNum]
90   S98S      [acc -= 1]
91   T102S      [smalNum = acc, acc = 0]
92   E67S     [acc == +]
93   T0S              [acc = 0, nextElement]
94   A100S        [acc = count]
95   A98S      [acc += 1]
96   T100S        [count = acc]
97   E69S [acc == +]
98   P0L          [val one = 1]
99   P0L          [var one = 1]
100  P0S          [var count = 0]
101  P0S          [var answer = 0]
102  P0S          [var smalNum = 0]
103  P0S          [var numCheck = 0]
104  A0S          [fun 1]
105  P1L             [size ;3]
106  P54S [listStart]
107  P0S              [var index]
108  [array] P3S [first]   [2*3 + 0 = 6]
109          P5S [second]  [2*3 + 1 = 7]
110          P4S [third]   [2*3 + 0 = 8]
111    [End]
```

The Edsac Simulator

File Edit Library Edsac Window Help

Find:

Initial Orders 1 | Stop Bell | Real Time | Sound | Hints | Short Tanks

WORD 102 Order = P 1 S Integer 102S = 2 Fraction 102S = 0.000031

Edsac

Output From: ExperimentOrder1_Working

Clear Reset Start Stop Single E.P

SCT Order Tank

Multiplier Long Tank 3

Multiplicand

Acc

Dial

[Реализовать нахождение наибольшего общего делителя (НОД) для массива чисел при помощи EDSAC; Ответ сохраняется в строке 102]

```
T111S
X0S
T0S             [acc = 0]
A106S  [acc = lineOfFirst]
T107S  [index = lineOfFirst; acc = 0]
T0S             [acc = 0; next]
A100S  [acc = count]
A98S    [acc += 1]
S105S           [acc = 1 - 3]
E67S    [acc == 0]
T0S
A107S  [acc = index]
A100S  [index += count]
L0L             [(var index)]
A104S           [A(var index)S]
T47S    [put fun 1]
A1S             [result first]
T103S  [numCheck = acc, acc = 0]
A100S  [count = 0]
S98S    [count -= 1]
E55S            [acc == -1]
T0S             [acc = 0]
A103S  [acc = numCheck]
T102S  [smalNum = acc, acc = 0]
T0S             [Skip]
A100S  [count = 0]
A98S    [count += 1]
T100S  [count = acc, acc = 0]
A102S           [acc = smalNum]
S103S  [acc -= numCheck]
G65S            [acc = +smalNum]
T0S             [acc = 0]
A103S  [acc = numCheck]
T102S  [smalNum = numCheck]
T0S             [Skip2]
E36S            [Next]
```

Начиная со строки 31 и заканчивая строкой 66, этот алгоритм представляет собой цикл, предназначенный для поиска наибольшего числа в списке. Первый элемент берется и

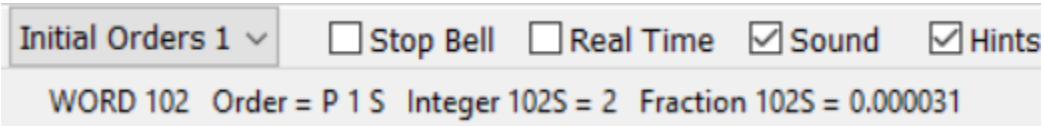
добавляется в качестве переменной, после чего эта часть кода всегда пропускается. Если результат вычитания в накопителе отрицательный, то текущее число в списке становится самым маленьким.

```
T0S             [----------mainPart, acc = 0----------]
T100S           [count = acc]
T0S             [acc = 0; nextElement1]
A100S  [acc = count]
A98S    [acc += 1]
S105S           [acc = 1 - 3]
E111S   [acc == 0]
T0S
A107S  [acc = index]
A100S  [index += count]
L0L             [(var index)]
A104S           [A(var index)S]
T80S    [put fun 1]
A1S             [result first]
U103S  [numCheck = acc, acc = 0]
S102S   [acc -= smalNum, substract]
G88S    [acc == -]
S98S    [acc -= 1]
G93S  [acc == -]
A98S    [acc += 1]
E82S    [acc == -]
T0S             [acc = 0, nextSmal]
A102S           [acc = smalNum]
S98S    [acc -= 1]
T102S  [smalNum = acc, acc = 0]
E67S    [acc == +]
T0S             [acc = 0, nextElement]
A100S           [acc = count]
A98S    [acc += 1]
T100S           [count = acc]
E69S [acc == +]
P0L     [val one = 1]
P0L     [var one = 1]
P0S     [var count = 0]
P0S     [var answer = 0]
P0S     [var smalNum = 0]
P0S     [var numCheck = 0]
A0S             [fun 1]
P1L             [size ;3]
```

```
P54S [listStart]
P0S                 [var index]
[array] P3S [first]   [2*3 + 0 = 6]
          P5S [second]  [2*3 + 1 = 7]
          P4S [third]   [2*3 + 0 = 8]
 [End]
```

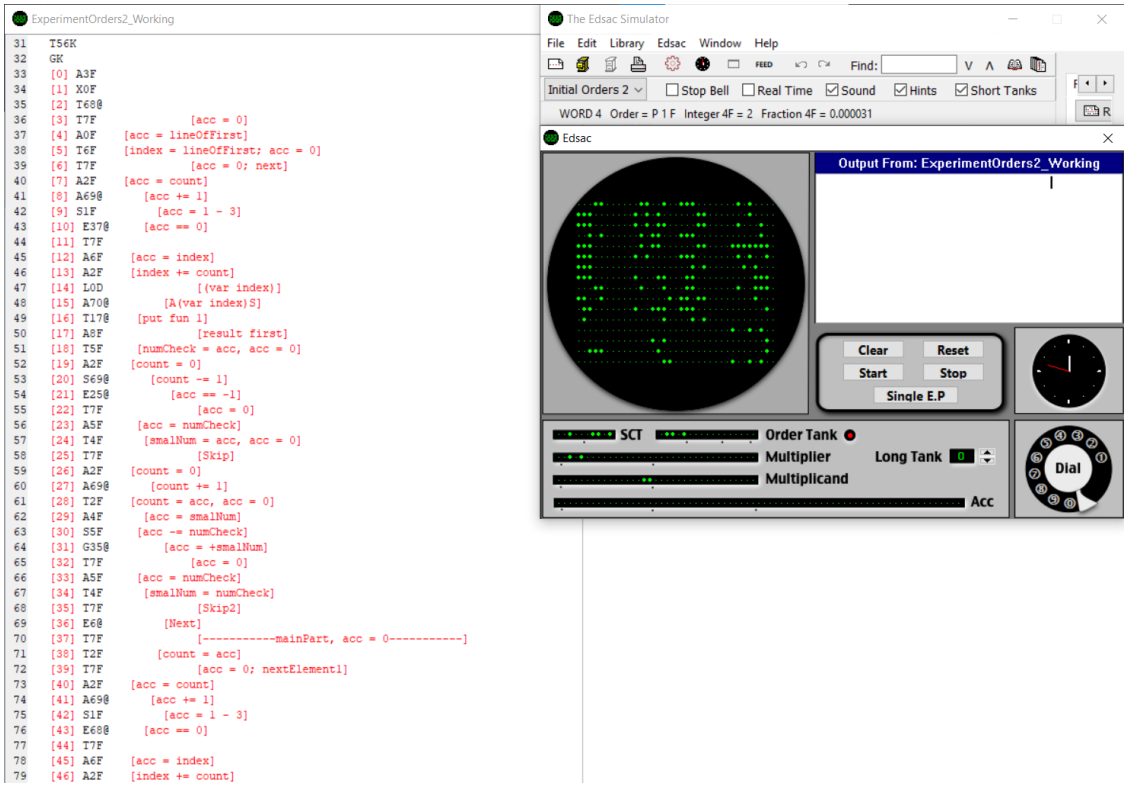
Это основная часть кода. После получения наименьшего значения оно всегда вычитается до тех пор, пока мы не получим ответ 0. Чтобы узнать, равен ли ответ нулю, он вычитает 1 в качестве проверки, так что, если после проверки он становится отрицательным, это означает, что ответ равен 0.

4. Работа программы Orders 1

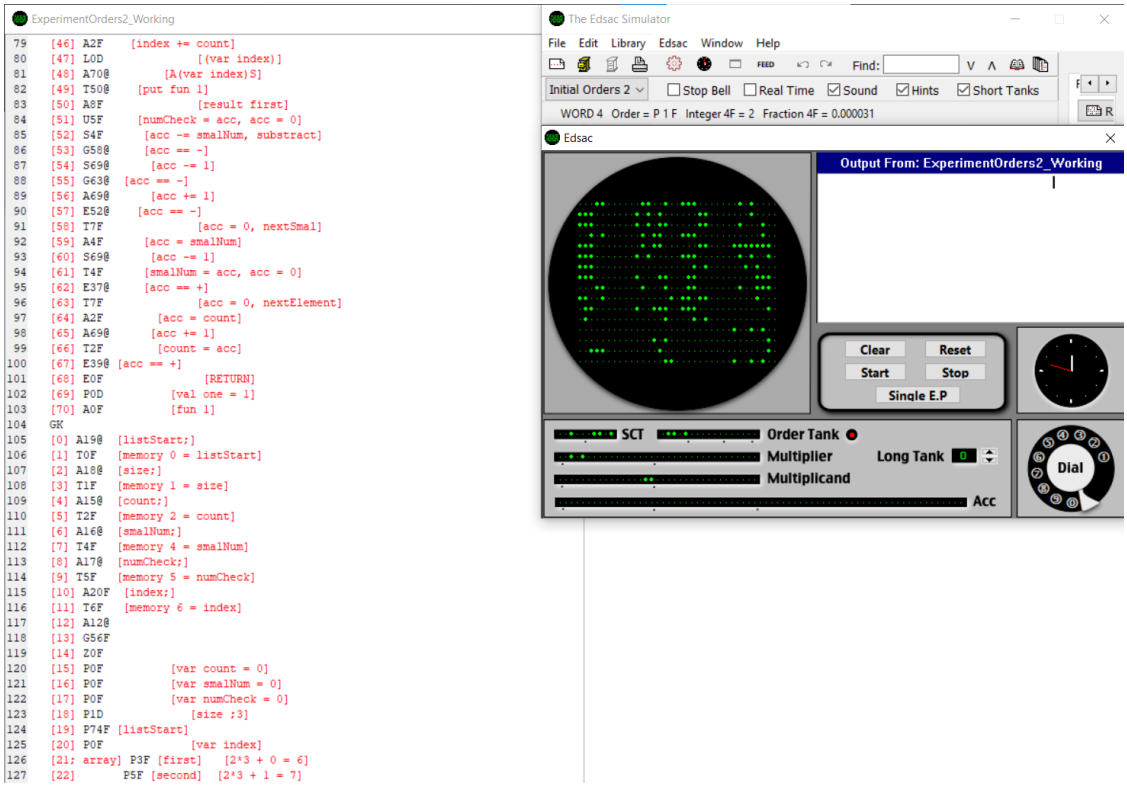


Ответ сохраняется в строке 102.

5. Программа Orders 2



```
ExperimentOrders2_Working
31   T56K
32   GK
33   [0] A3F
34   [1] X0F
35   [2] T68@
36   [3] T7F              [acc = 0]
37   [4] A0F    [acc = lineOfFirst]
38   [5] T6F    [index = lineOfFirst; acc = 0]
39   [6] T7F              [acc = 0; next]
40   [7] A2F    [acc = count]
41   [8] A69@      [acc += 1]
42   [9] S1F        [acc = 1 - 3]
43   [10] E37@     [acc == 0]
44   [11] T7F
45   [12] A6F     [acc = index]
46   [13] A2F     [index += count]
47   [14] L0D             [(var index)]
48   [15] A70@         [A(var index)S]
49   [16] T17@    [put fun 1]
50   [17] A8F             [result first]
51   [18] T5F     [numCheck = acc, acc = 0]
52   [19] A2F     [count = 0]
53   [20] S69@       [count -= 1]
54   [21] E25@          [acc == -1]
55   [22] T7F             [acc = 0]
56   [23] A5F     [acc = numCheck]
57   [24] T4F      [smalNum = acc, acc = 0]
58   [25] T7F             [Skip]
59   [26] A2F     [count = 0]
60   [27] A69@       [count += 1]
61   [28] T2F     [count = acc, acc = 0]
62   [29] A4F       [acc = smalNum]
63   [30] S5F     [acc -= numCheck]
64   [31] G35@         [acc = +smalNum]
65   [32] T7F             [acc = 0]
66   [33] A5F     [acc = numCheck]
67   [34] T4F      [smalNum = numCheck]
68   [35] T7F             [Skip2]
69   [36] E6@          [Next]
70   [37] T7F             [----------mainPart, acc = 0----------]
71   [38] T2F         [count = acc]
72   [39] T7F             [acc = 0; nextElement1]
73   [40] A2F    [acc = count]
74   [41] A69@       [acc += 1]
75   [42] S1F        [acc = 1 - 3]
76   [43] E68@     [acc == 0]
77   [44] T7F
78   [45] A6F     [acc = index]
79   [46] A2F     [index += count]
```

```
ExperimentOrders2_Working
79   [46] A2F     [index += count]
80   [47] L0D             [(var index)]
81   [48] A70@         [A(var index)S]
82   [49] T50@    [put fun 1]
83   [50] A8F             [result first]
84   [51] U5F     [numCheck = acc, acc = 0]
85   [52] S4F      [acc -= smalNum, substract]
86   [53] G58@     [acc == -]
87   [54] S69@       [acc -= 1]
88   [55] G63@  [acc == -]
89   [56] A69@       [acc += 1]
90   [57] E52@     [acc == -]
91   [58] T7F             [acc = 0, nextSmal]
92   [59] A4F      [acc = smalNum]
93   [60] S69@       [acc -= 1]
94   [61] T4F      [smalNum = acc, acc = 0]
95   [62] E37@     [acc == +]
96   [63] T7F             [acc = 0, nextElement]
97   [64] A2F        [acc = count]
98   [65] A69@       [acc += 1]
99   [66] T2F         [count = acc]
100  [67] E39@ [acc == +]
101  [68] E0F              [RETURN]
102  [69] P0D         [val one = 1]
103  [70] A0F         [fun 1]
104  GK
105  [0] A19@  [listStart;]
106  [1] T0F   [memory 0 = listStart]
107  [2] A18@  [size;]
108  [3] T1F   [memory 1 = size]
109  [4] A15@  [count;]
110  [5] T2F   [memory 2 = count]
111  [6] A16@  [smalNum;]
112  [7] T4F   [memory 4 = smalNum]
113  [8] A17@  [numCheck;]
114  [9] T5F   [memory 5 = numCheck]
115  [10] A20F  [index;]
116  [11] T6F   [memory 6 = index]
117  [12] A12@
118  [13] G56F
119  [14] Z0F
120  [15] P0F         [var count = 0]
121  [16] P0F         [var smalNum = 0]
122  [17] P0F         [var numCheck = 0]
123  [18] P1D           [size ;3]
124  [19] P74F [listStart]
125  [20] P0F           [var index]
126  [21; array] P3F [first]   [2*3 + 0 = 6]
127  [22]       P5F [second]   [2*3 + 1 = 7]
```

[Реализовать нахождение наибольшего общего делителя (НОД) для массива чисел при помощи EDSAC; ответ сохраняется в ячейка 4]

```
T56K
GK
[0] A3F
[1] X0F
[2] T68@
[3] T7F             [acc = 0]
[4] A0F  [acc = lineOfFirst]
[5] T6F  [index = lineOfFirst; acc = 0]
[6] T7F             [acc = 0; next]
[7] A2F  [acc = count]
[8] A69@            [acc += 1]
[9] S1F             [acc = 1 - 3]
[10] E37@           [acc == 0]
[11] T7F
[12] A6F            [acc = index]
[13] A2F            [index += count]
[14] L0D            [(var index)]
[15] A70@           [A(var index)S]
[16] T17@           [put fun 1]
[17] A8F            [result first]
[18] T5F            [numCheck = acc, acc = 0]
[19] A2F            [count = 0]
[20] S69@           [count -= 1]
[21] E25@           [acc == -1]
[22] T7F            [acc = 0]
[23] A5F            [acc = numCheck]
[24] T4F            [smalNum = acc, acc = 0]
[25] T7F            [Skip]
[26] A2F            [count = 0]
[27] A69@           [count += 1]
[28] T2F[count = acc, acc = 0]
[29] A4F            [acc = smalNum]
[30] S5F            [acc -= numCheck]
[31] G35@           [acc = +smalNum]
[32] T7F            [acc = 0]
[33] A5F            [acc = numCheck]
[34] T4F            [smalNum = numCheck]
[35] T7F            [Skip2]
[36] E6@            [Next]
[37] T7F            [----------mainPart, acc = 0----------]
[38] T2F            [count = acc]
[39] T7F            [acc = 0; nextElement1]
[40] A2F            [acc = count]
[41] A69@           [acc += 1]
[42] S1F            [acc = 1 - 3]
[43] E68@           [acc == 0]
```

```
[44] T7F
[45] A6F          [acc = index]
[46] A2F          [index += count]
[47] L0D          [(var index)]
[48] A70@         [A(var index)S]
[49] T50@         [put fun 1]
[50] A8F          [result first]
[51] U5F          [numCheck = acc, acc = 0]
[52] S4F          [acc -= smalNum, substract]
[53] G58@         [acc == -]
[54] S69@         [acc -= 1]
[55] G63@  [acc == -]
[56] A69@         [acc += 1]
[57] E52@         [acc == -]
[58] T7F          [acc = 0, nextSmal]
[59] A4F          [acc = smalNum]
[60] S69@         [acc -= 1]
[61] T4F          [smalNum = acc, acc = 0]
[62] E37@         [acc == +]
[63] T7F          [acc = 0, nextElement]
[64] A2F          [acc = count]
[65] A69@         [acc += 1]
[66] T2F          [count = acc]
[67] E39@ [acc == +]
[68] E0F          [RETURN]
[69] P0D          [val one = 1]
[70] A0F          [fun 1]
GK
[0] A19@  [listStart;]
[1] T0F   [memory 0 = listStart]
[2] A18@  [size;]
[3] T1F   [memory 1 = size]
[4] A15@  [count;]
[5] T2F   [memory 2 = count]
[6] A16@  [smalNum;]
[7] T4F   [memory 4 = smalNum]
[8] A17@  [numCheck;]
[9] T5F   [memory 5 = numCheck]
[10] A20F  [index;]
[11] T6F   [memory 6 = index]
[12] A12@
[13] G56F
[14] Z0F
[15] P0F          [var count = 0]
[16] P0F          [var smalNum = 0]
[17] P0F          [var numCheck = 0]
[18] P1D          [size ;3]
[19] P74F [listStart]
[20] P0F          [var index]
```

```
[21; array] P3F [first]   [2*3 + 0 = 6]
[22]        P5F [second]  [2*3 + 1 = 7]
[23]        P4F [third]   [2*3 + 0 = 8]
EZPF
```

6. Работа программы Orders 2

Ответ сохраняется в ячейка 4.

7. Результат

Edsac считается первым компьютером, который может выполнять широкий спектр вычислительных задач. Возможности алгоритма очень просты, но, несмотря на это, он позволяет пользователю программировать все, что требуется. Самая большая проблема с edsac заключается в том, что пользователь не будет знать, работает программа или нет, пока не будет написан полный алгоритм. Если бы комментарии внутри алгоритма можно было распознать, edsac был бы намного более удобным для пользователя.