Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

**Кафедра «Информатика и вычислительная техника»**

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2**

Дисциплина: Низкоуровневое программирование

Тема: Программирование на Edsac

Выполнил

студент гр.3530901/10005 Л. С. Алексеев

Руководитель Коренев Д.А.

«12» октября 2022 г.

Санкт-Петербург

2022

Санкт-Петербургский государственный политехнический университет

Содержание

[**1** **Описание задачи** 2](#_Toc116500474)

[**2** **Алгоритм Евклида** 2](#_Toc116500475)

[**3** **Программа IO1** 2](#_Toc116500476)

[**4** **Работа программы IO1** 3](#_Toc116500477)

[**5** **Программа IO2** 4](#_Toc116500478)

[**6** **Работа программы IO2** 6](#_Toc116500479)

[**7** **Вывод** 7](#_Toc116500480)

# **ТЗ**

Обмен в массиве элементов попарно. Если 1-е число больше 2-го меняем их местами. Если 3-е число больше 4-го меняем их местами. И т.д.

# **Метод решения**

Идея алгоритма заключается в том, что мы вычитаем из N-ого элемента массива N+1 в цикле с шагом 2. Если получившаяся разность положительна, то осуществляется перестановка элементов.

# **Программа IO1**

[31] T 99 S

[32] X 0 S

[33] T 0 S [ обнуляем аккумулятор ]

[34] A 86 [len] S [ загружаем в аккум. длину массива ]

[35] T 1 S [ записываем длину массива в ячейку 1 ]

[36] A 87 [addr] S [ загружаем в аккум. адрес 0-го элемента массива ]

[37] L 0 L [ сдвиг аккумулятора на 1 разряд влево ]

[38] A 61 [r1] S [ прибавляем код инструкции с нулевым полем адреса ]

[39] U 61 [r1] S [ запиcываем сформированную инструкцию r1 ]

[40] U 57 [c1] S [ запиcываем сформированную инструкцию c1 ]

[41] A 85 [c] S [ прибавление 1 к полю адреса ]

[42] T 63 [r2] S [ запиcываем сформированную инструкцию r2 ]

[43] A 87 [addr] S [ загружаем в аккумулятор адрес 0-го элемента массива ]

[44] L 0 L [ сдвиг аккумулятора на 1 разряд влево ]

[45] A 64 [w1] S [ прибавляем код инструкции w1 с нулевым полем адреса ]

[46] U 64 [w1] S [ запиcываем сформированную инструкцию w1 ]

[47] A 85 [c] S [ прибавление 1 к полю адреса ]

[48] T 66 [w2] S [ запиcываем сформированную инструкцию w2 ]

[49] A 87 [addr] S [ загружаем в аккум. адрес массива ]

[50] L 0 L [ сдвиг аккум. на 1 разряд влево ]

[51] A 58 [c2] S [ прибавляем код инструкции c2 с нулевым полем адреса ]

[52] T 58 [c2] S [ записываем сформированную инструкцию c2 ]

[53] [loop:] A 1 S [ загружаем счетчик необработанных элементов массива ]

[54] S 85 [c] S [ уменьшаем на 2 ]

[55] G 84 [exit] S [ если результат меньше 0, завершаем работу ]

[56] T 1 S [ обновляем значение счетчика и обнуляем аккумулятор ]

[57] [с1:] A 0 S [ загружаем очередной элемент массива ]

[58] [с2:] S 1 S [ отнимаем от него следующий ]

[59] G 67 S [ если разность отрицательна, то пропускаем блок перестановки ]

[60] T 2 S [ очищаем аккум. ]

[61][r1:] A 0 S [ загружаем в аккум. элемент с индексом N ]

[62] T 0 S [ записываем этот элемент в буферную ячейку ]

[63] [r2:] A 1 S [ загружаем в аккум. элемент с индексом N+1 ]

[64] [w1:] T 0 S [ записываем этот элемент в ячейку с адресом N ]

[65] A 0 S [ загружаем в аккум. значение из буферную ячейки 0]

[66] [w2:] T 1 S [ записываем это значение в ячейку с адресом N+1 ]

[67] T 2 S [ очищаем аккум. ]

[68] A 63 [r2] S [ загружаем в аккум. код инструкции r2 ]

[69] A 85 [c] S [ прибавление 1 к полю адреса ]

[70] U 61 [r1] S [ запиcываем сформированную инструкцию r1 ]

[71] U 57 [m1] S [ запиcываем сформированную инструкцию m1 ]

[72] A 85 [c] S [ прибавление 1 к полю адреса ]

[73] T 63 [r2] S [ запись сформированной инструкции r2 ]

[74] A 66 [w2] S [ загружаем в аккум. код инструкции w2 ]

[75] A 85 [c] S [ прибавление 1 к полю адреса ]

[76] U 64 [w1] S [ запиcываем сформированную инструкцию w1 ]

[77] A 85 [c] S [ прибавление 1 к полю адреса ]

[78] T 66 [w2] S [ запиcываем сформированную инструкцию w2 ]

[79] A 85 [c] S [ загружаем в аккум. константу 2 ]

[80] L 0 L [ сдвиг на 1 разряд влево ]

[81] A 58 [m2] S [ прибавляем код инструкции m2 , исполненной на предыдущем шаге ]

[82] T 58 [m2] S [ записываем сформированную инструкцию ]

[83] E 53 [loop] S [ повторяем все операции; аккум. обнулен ]

[84] [exit:] Z 0 S [ останов ]

[85] [с:] P 1 S [ константа 2 ]

[86] [len:] P 5 L [ 11 - длина массива ]

[87] [addr:] P 44 S [ 88 - адрес массива ]

[88] [array:] P 5 S [ 10 ]

[89] P 4 L [ 9 ]

[90] P 4 S [ 8 ]

[91] P 3 L [ 7 ]

[92] P 3 S [ 6 ]

[93] P 2 L [ 5 ]

[94] P 2 S [ 4 ]

[95] P 1 L [ 3 ]

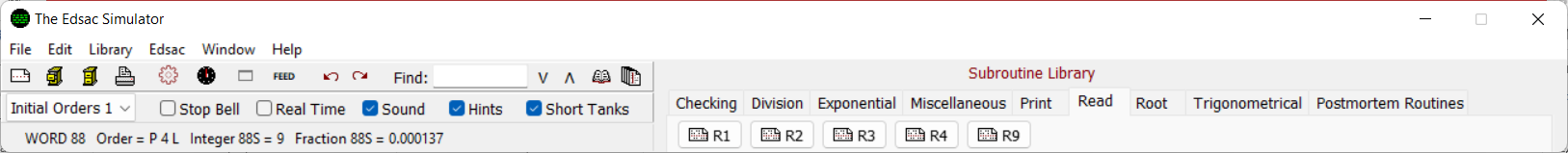
[96] P 1 S [ 2 ]

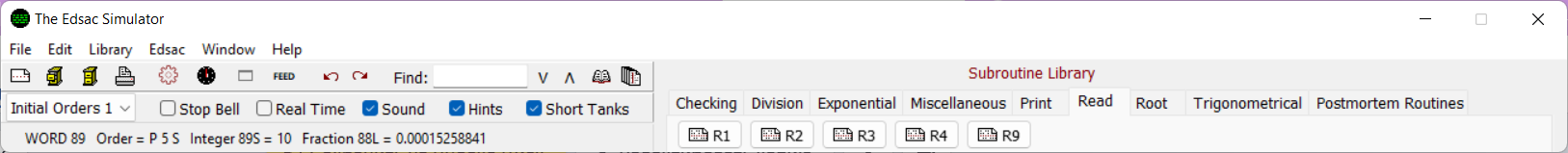
[97] P 0 L [ 1 ]

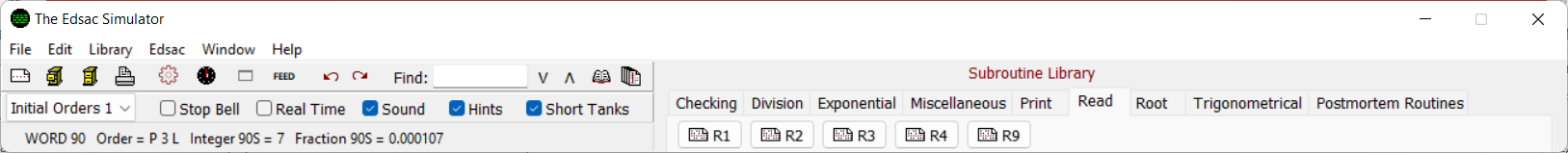
[98] P 0 S [ 0 ]

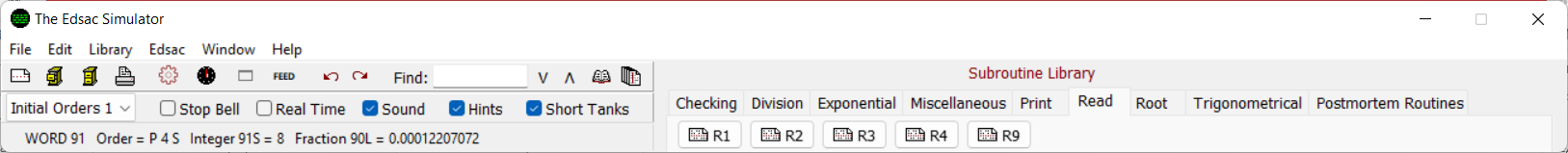
# **Работа программы IO1**

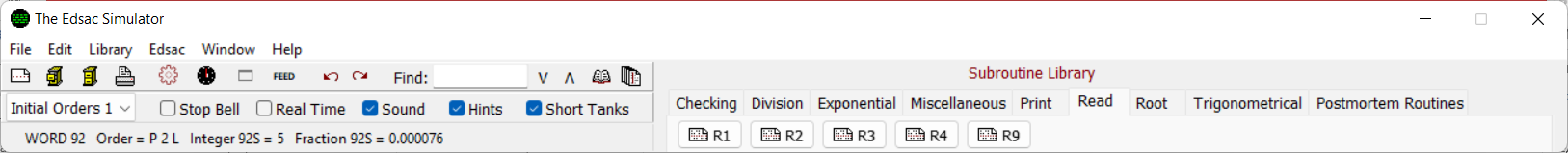
Для входных данных в виде массива (10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 0) результат работы программы будет выглядеть следующим образом:

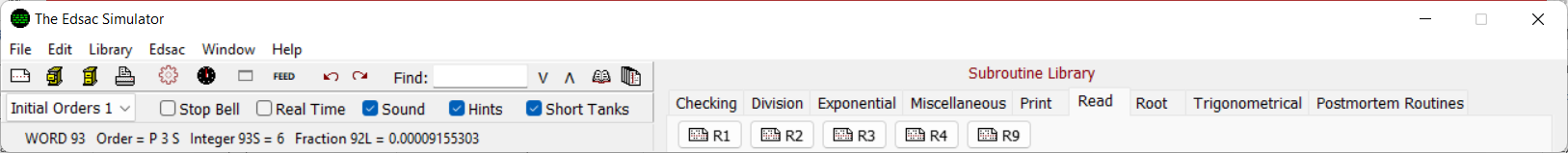


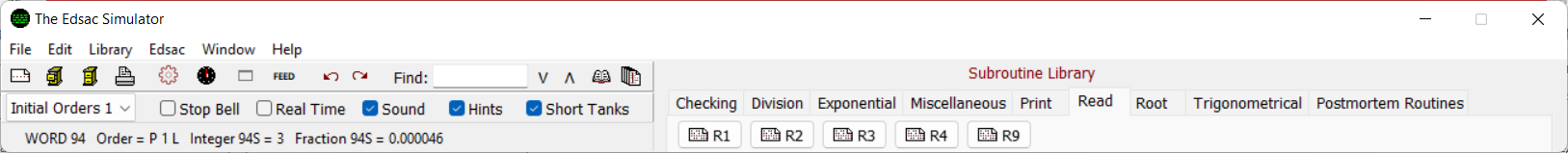


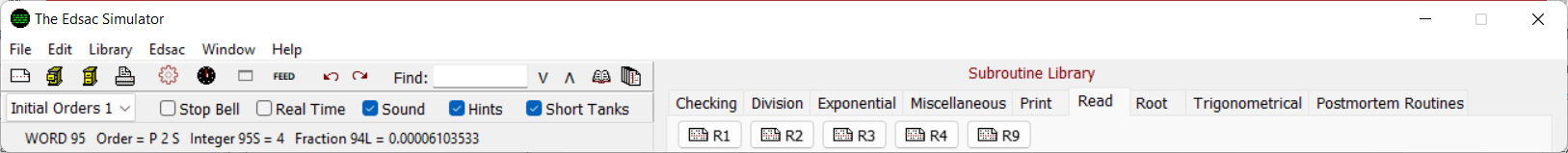


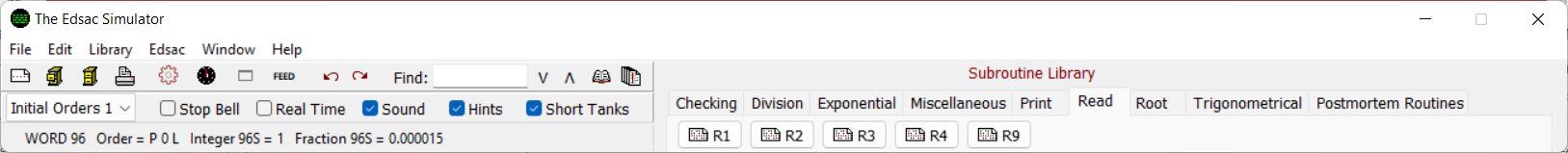


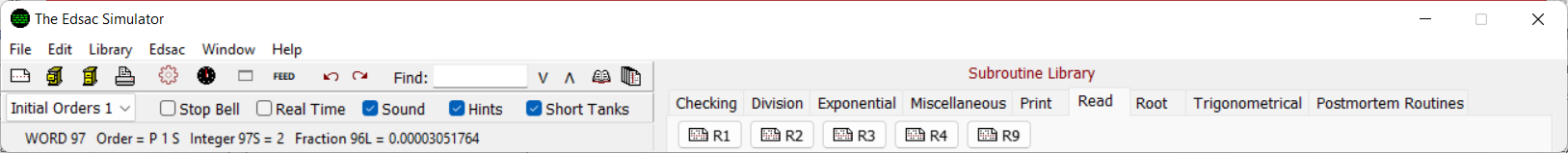


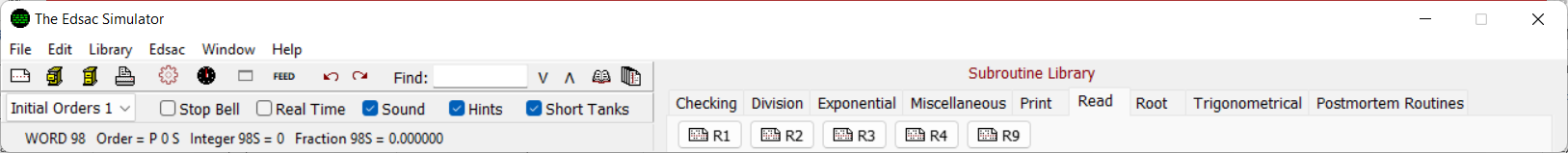












# **Программа IO2**

[55] T 56 K [ директива IO2, установка адреса загрузки ]

[56] [ 0:] G K [ директива IO2, фиксация начального адреса подпрограммы ] A 3 F [ пролог: формирование кода инструкции возврата в Acc ]

[57] [ 1:] T 59 [<ret>] @ [ пролог: запись инструкции возврата ]

[58] [ 2:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]

[59] [ 3:] A 61 [<r1init>] @ [ прибавляем код инструкции с нулевым полем адреса ]

[60] [ 4:] U 22 [<c1>] @ [ запись сформированной инструкции c1, обнуление аккумулятора ]

[61] [ 5:] T 26 [<r1>] @ [ запись сформированной инструкции r1, обнуление аккумулятора ]

[62] [ 6:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]

[63] [ 7:] A 62 [<w1init>] @ [ прибавляем код инструкции w1 с нулевым полем адреса ]

[64] [ 8:] T 29 [<w1>] @ [ запись сформированной инструкции w1, обнуление аккумулятора ]

[65] [ 9:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]

[66] [10:] A 63 [<r2init>] @ [ прибавляем код инструкции r2 с полем адреса 1 ]

[67] [11:] T 28 [<r2>] @ [ запись сформированной инструкции r2, обнуление аккумулятора ]

[68] [12:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]

[69] [13:] A 64 [<w2init>] @ [ прибавляем код инструкции w2 с полем адреса 1 ]

[70] [14:] T 31 [<w2>] @ [ запись сформированной инструкции w2, обнуление аккумулятора ]

[71] [15:] A 0 [<addr>] F [ загрузка в аккумулятор адреса 0-го элемента массива ]

[72] [16:] A 65 [<c2init>] @ [ прибавляем код инструкции c2 с полем адреса 1 ]

[73] [17:] T 23 [<c2>] @ [ запись сформированной инструкции c2, обнуление аккумулятора ]

[74] [18:] [loop:] [20:] A 1 F [ загружаем счетчик необработанных элементов массива ]

[75] [19:] S 60 [<c2>] @ [ уменьшаем на 2 ]

[76] [20:] G 58 [<exit>] @ [ если результат меньше 0, завершаем работу ]

[77] [21:] T 1 F [ обновляем значение счетчика и обнуляем аккумулятор ]

[78] [22:] [c1:] A 0 F [ загрузка в аккумулятор значения из ячейки N ]

[79] [23:] [c2:] S 1 F [ вычитание из аккумулятора значения из ячейки N+1 ]

[80] [24:] G 32 [<reverse end>] @ [ если разность <0, то пропускаем блок перестановки ]

[81] [25:] T 2 F [ очищаем аккумулятор ]

[82] [26:] [r1:] A 0 F [ загрузка в аккумулятор значения из ячейки N ]

[83] [27:] T 0 F [ запись этого значения в рабочую ячейку, обнуление аккумулятора ]

[84] [28:] [r2:] A 1 F [ загрузка в аккумулятор значения из ячейки N+1 ]

[85] [29:] [w1:] T 0 F [ запись этого значения в ячейку с адресом N, обнуление аккумулятора ]

[86] [30:] A 0 F [ загрузка в аккумулятор значения из ячейки 0 ]

[87] [31:] [w2:] T 1 F [ запись этого значения в ячейку с адресом N+1, обнуление аккумулятора ]

[88] [32:] [reverse end:] T 2 F [ очищаем аккумулятор ]

[89] [33:] A 60 [<c>] @ [ загрузка в аккумулятор константы 2 ]

[90] [34:] L 0 D [ сдвиг на 1 разряд влево ]

[91] [35:] A 22 [<c1>] @ [ прибавляем код инструкции c1, исполненной на предыдущем шаге ]

[92] [36:] T 22 [<c1>] @ [ записываем сформированную инструкцию в память ]

[93] [37:] A 60 [<c>] @ [ загрузка в аккумулятор константы 2 ]

[94] [38:] L 0 D [ сдвиг на 1 разряд влево ]

[95] [39:] A 23 [<c2>] @ [ прибавляем код инструкции c2, исполненной на предыдущем шаге ]

[96] [40:] T 23 [<c2>] @ [ записываем сформированную инструкцию в память ]

[97] [41:] A 60 [<c>] @ [ загрузка в аккумулятор константы 2 ]

[98] [42:] L 0 D [ сдвиг на 1 разряд влево ]

[99] [43:] A 26 [<r1>] @ [ прибавляем код инструкции r1, исполненной на предыдущем шаге ]

[100] [44:] T 26 [<r1>] @ [ записываем сформированную инструкцию в память ]

[101] [45:] A 60 [<c>] @ [ загрузка в аккумулятор константы 2 ]

[102] [46:] L 0 D [ сдвиг на 1 разряд влево ]

[103] [47:] A 29 [<w1>] @ [ прибавляем код инструкции w1, исполненной на предыдущем шаге ]

[104] [48:] T 29 [<w1>] @ [ записываем сформированную инструкцию в память ]

[105] [49:] A 60 [<c>] @ [ загрузка в аккумулятор константы 2 ]

[106] [50:] L 0 D [ сдвиг на 1 разряд влево ]

[107] [51:] A 28 [<r2>] @ [ прибавляем код инструкции r2, исполненной на предыдущем шаге ]

[108] [52:] T 28 [<r2>] @ [ записываем сформированную инструкцию в память ]

[109] [53:] A 60 [<c>] @ [ загрузка в аккумулятор константы 2 ]

[110] [54:] L 0 D [ сдвиг на 1 разряд влево ]

[111] [55:] A 31 [<w2>] @ [ прибавляем код инструкции w2, исполненной на предыдущем шаге ]

[112] [56:] T 31 [<w2>] @ [ записываем сформированную инструкцию в память ]

[113] [57:] E 18 [<loop>] @ [ повторяем все операции; аккумулятор обнулен ]

[114] [58:] [exit:] T 0 F [ обнуление аккумулятора ]

[115] [59:] [ret:] E 0 F [ эпилог: инструкция возврата из подпрограммы ]

[116] [60:] [с:] P 1 F [ константа 2 ]

[117] [61:] [r1init:] A 0 F [ основа для формирования инструкции с метками r1 и c1 ]

[118] [62:] [w1init:] T 0 F [ основа для формирования инструкции с меткой w1 ]

[119] [63:] [r2init:] A 1 F [ основа для формирования инструкции с меткой r2 ]

[120] [64:] [w2init:] T 1 F [ основа для формирования инструкции с меткой w2 ]

[121] [65:] [M2init:] S 1 F [ основа для формирования инструкции с меткой c2 ]

[122] [ 0:] G K [ директива IO2, фиксация начального адреса программы ] X 0 F

[123] [ 1:] A 8 [<addr>] @ [ загрузка в аккумулятор адреса массива ]

[124] [ 2:] T 0 F [ запись адреса массива в ячейку 0, обнуление аккумулятора ]

[125] [ 3:] A 9 [<len>] @ [ загрузка в аккумулятор длины массива ]

[126] [ 4:] T 1 F [ запись длины массива в ячейку 1, обнуление аккумулятора ]

[127] [ 5:] A 5 @ [\ вызов ]

[128] [ 6:] G 56 [<sub>] F [/ подпрограммы ]

[129] [ 7:] Z 0 F [ останов ]

[130] [ 8:] P 10 [<array>] @ [ адрес массива ]

[131] [ 9:] P 5 D [ длина массива - 11 ]

[132] [10:] P 4 D [ 10 ]

[133] [11:] P 5 F [ 9 ]

[134] [12:] P 4 F [ 8 ]

[135] [13:] P 3 D [ 7 ]

[136] [14:] P 3 F [ 6 ]

[137] [15:] P 2 D [ 5 ]

[138] [16:] P 2 F [ 4 ]

[139] [17:] P 1 D [ 3 ]

[140] [18:] P 1 F [ 2 ]

[141] [19:] P 0 D [ 1 ]

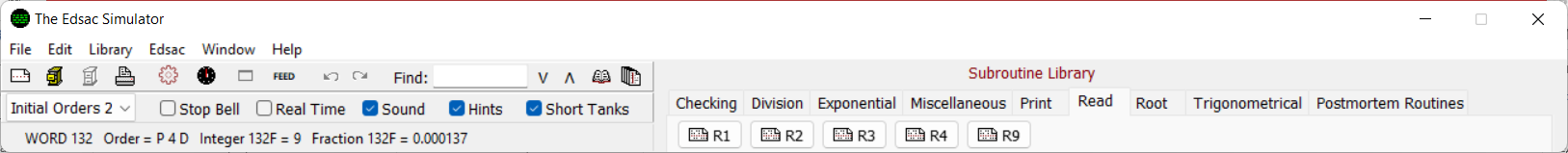
[142] [20:] P 2 D [ 5 ]

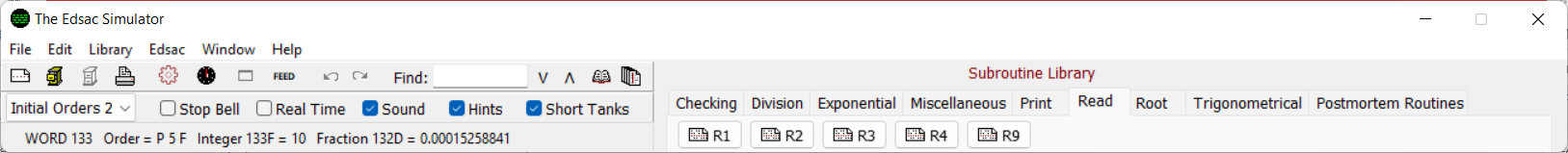
[143] EZ PF [ директива IO2, переход к исполнению ]

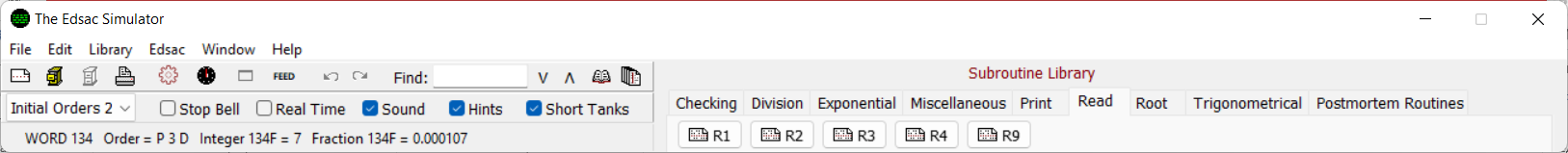
Так как функционал программы составляет только возможность перестановки элементов массива, результат работы программы находится в тех же ячейках, где располагаются исходные данные. В рассматриваемом случае это ячейки [132-142].

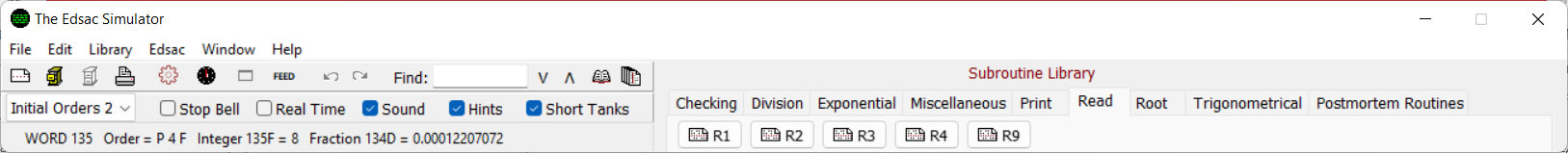
# **Работа программы IO2**

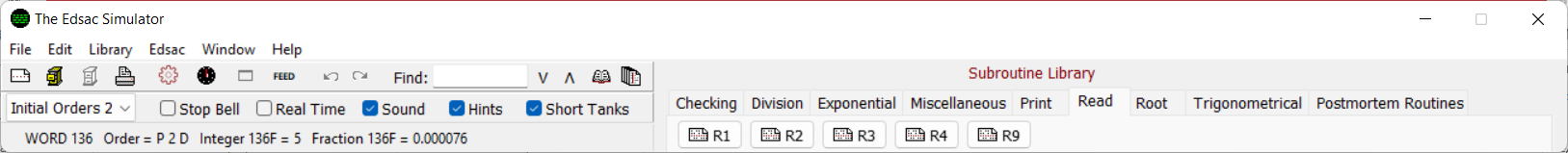
Для входных данных в виде массива (10, 9, 8, 7, 6, 5, 4, 3, 2, 1, 5) результат работы программы будет выглядеть следующим образом:

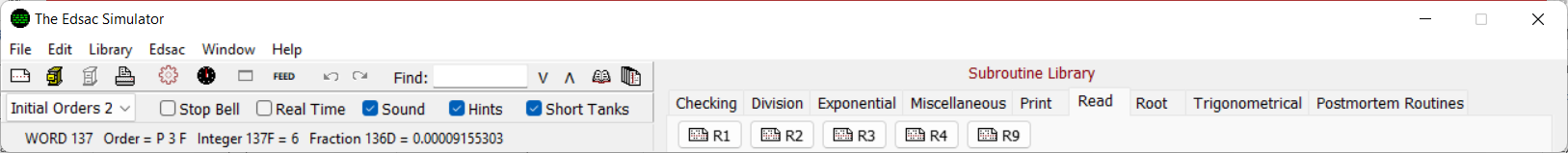


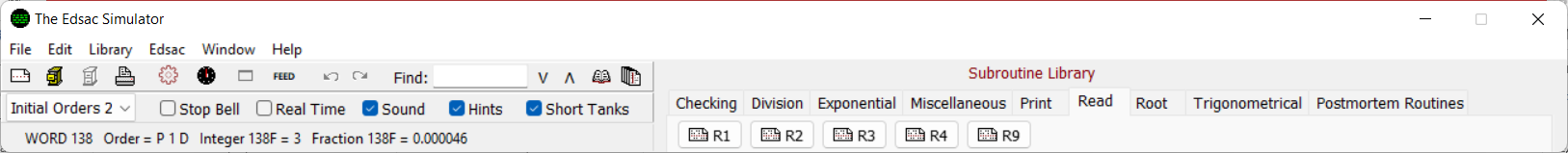


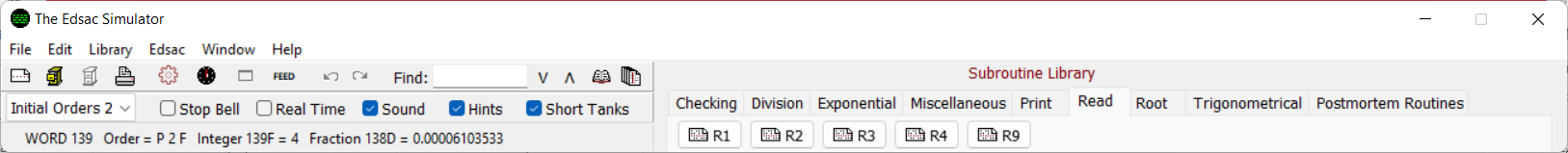


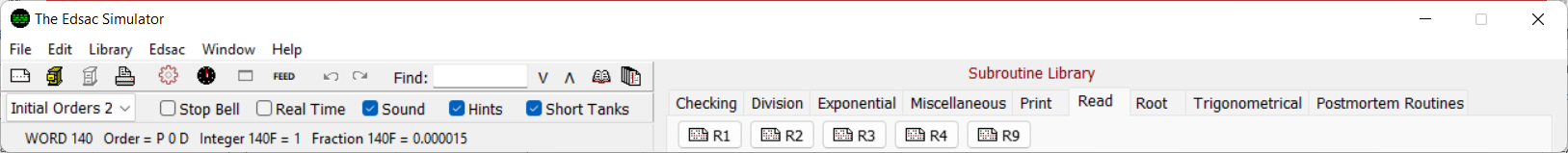


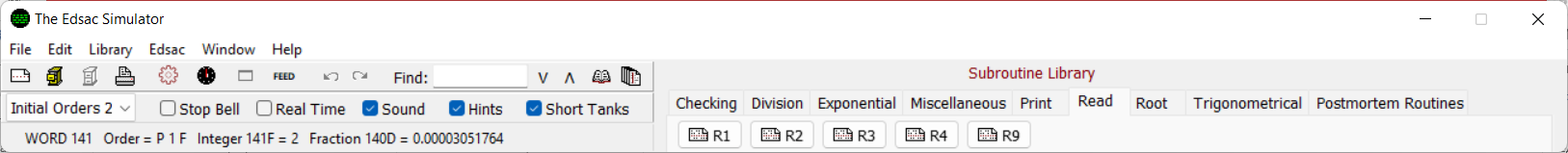


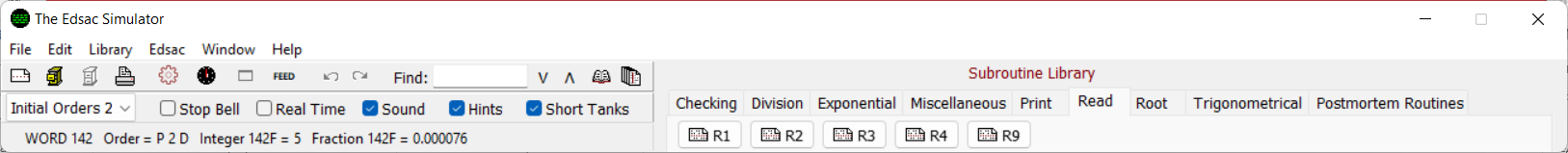












# **Вывод**

Являясь одной из первых ЭВМ, EDSAC может выполнять широкий спектр задач, несмотря на ограничения, вызванные неудобством программирования и малой вычислительной мощностью.