Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Лабораторная работа № 2

EDSAC

по дисциплине «Низкоуровневое программирование»

Выполнил
студент гр. 3530901/00004

_____ Кручинин К. А.
(подпись)

Руководитель

_____ Соболь. В.
(подпись)

«___» _____ 2021 г.

Санкт-Петербург
2021

Задача

В соответствии с условием 7 варианта требуется написать программу для EDSAC осуществляющую определение k-й порядковой статистики in-place.

Лабораторная работа делится на две части:

1. Разработать программу для EDSAC, реализующую определенную вариантом задания функциональность, и предполагающую загрузчик Initial Orders 1. Массив (массивы) данных и другие параметры (преобразуемое число, длина массива, параметр статистики и пр.) располагаются в памяти по фиксированным адресам.
2. Выделить определенную вариантом задания функциональность в замкнутую (closed) подпрограмму, разработать вызывающую ее тестовую программу. Использовать возможности загрузчика Initial Orders 2. Адрес обрабатываемого массива данных и другие параметры передавать через ячейки памяти с фиксированными адресами.

Алгоритм

Необходимо смоделировать программу для EDSAC, которая определит такой элемент неупорядоченного массива, если бы он был k-ым в упорядоченном. Для реализации сначала отсортируем массив произвольной длины алгоритмом сортировки вставками, опираясь на написанный на языке Kotlin алгоритм (рис. 1). Затем выведем k-ый элемент массива.

```kotlin
fun sort(list: MutableList<Int>): List<Int> {
    for (i in 1 until list.size) {
        val insertElement = list[i]
        var j = i - 1
        while (j >= 0 && list[j] > insertElement) {
            list[j + 1] = list[j]
            j--
        }
        list[j + 1] = insertElement
    }
    return list
}
```

Рис.1 Сортировка вставками на языке Kotlin.

Программа для загрузчика Initial Orders 1.

```
[31]T 111 S [Указатель на последнюю команду]
            [Цикл i]
[32]T 0 S   [Обнуление acc]
[33]A 98 S  [Берём i из ячейки 98 и записываем в acc]
[34]X 0 S
[35]A 102 S [Берём указатель на первый элемент массива из ячейки 102 и записываем в acc]
[36]L 0 L   [Сдвиг аккумулятора на 1 разряд влево]
[37]A 100 S [Записываем команду A 0 S из ячейки 100 в аккумулятор]
[38]T 39 S  [Берём предыдущую команду из аккумулятора и записываем в ячейку 39]
[39]A 0 S   [Прочитали i-й элемент]
[40]T 1 S   [Положили в рабочую ячейку (insertElement)]
[41]A 98 S  [Берём i]
[42]S 101 S [Делаем i--]
[43]U 99 S  [Кладём результат в j]
            [Цикл j]
[44]X 0 S
[45]X 0 S
[46]A 102 S [Берём указатель на первый элемент массива]
[47]L 0 L   [Сдвиг acc на 1 разряд влево]
[48]A 100 S [Добавляем команду A 0 S]
[49]T 50 S  [Записываем результат в ячейку 50]
[50]A 0 S   [Прочитали list(j)]
[51]U 2 S   [Запомнили list(j)]
[52]S 1 S   [Сравнили с insertElement]
[53]G 68 S  [Если acc < 0 значит list(j) < list(i), следовательно идём в конец цикла]
[54]X 0 S
[55]T 0 S   [Обнуление acc]
[56]A 102 S [Берём указатель на первый элемент массива]
[57]A 99 S  [+j]
[58]A 101 S [+1]
[59]L 0 L   [Сдвиг acc на 1 разряд влево]
[60]A 97 S  [Добавляем команду T 0 S]
[61]T 63 S  [Записываем в ячейку 63, обнулем acc]
[62]A 2 S   [Кладём в acc list(j)]
[63]T 0 S   [list(j+1) = list(j), потом обнуляем]
[64]A 99 S  [Берём j]
[65]S 101 S [-1]
[66]U 99 S  [j--]
[67]E 44 S  [Возвращаемся в начало j-цикла]
            [КОНЕЦ ЦИКЛА j]
```

Рис. 2 Программа для загрузчика IO1 строки 31–67.

```
            [КОНЕЦ ЦИКЛА j]
[68]T 0 S   [Обнуляем acc]
[69]A 99 S  [Берём j]
[70]A 102 S [Добавляем указатель на 1 элемент массива]
[71]A 101 S [+1]
[72]L 0 L   [Сдвиг acc на 1 разряд влево]
[73]A 97 S  [Добавляем команду T 0 S]
[74]T 76 S  [Записываем всё из acc в 76 ячейку, обнуляем acc]
[75]A 1 S   [Берём insertElement]
[76]T 0 S   [list(j+1) = insertElement]
[77]A 98 S  [Берём i]
[78]A 101 S [+1]
[79]U 98 S  [i++]
[80]S 104 S [i - length]
[81]G 32 S  [Если i < length то идём в начало цикла]
            [СОРТИРОВКА ЗАКОНЧЕНА]
[82]T 0 S   [Обнуление acc]
[83]A 102 S [Берём указатель на первый элемент массива]
[84]A 103 S [+k]
[85]S 101 S [-1]
[86]L 0 L   [Сдвиг acc на 1 разряд влево]
[87]A 100 S [Добавляем команду A 0 S]
[88]T 89 S  [Записываем результат в ячейку 89]
[89]A 0 S   [k-й элем отсорт массива]
[90]T 3 S   [Вывод результата в рабочую ячейку 3]
[91]Z 0 S   [Остановка машины]
[92]X 0 S
[93]X 0 S
[94]X 0 S
[95]X 0 S
[96]X 0 S
[97]T 0 S   [Команда для чтения из acc]
[98]P 0 L   [i]
[99]P 0 S   [j]
[100]A 0 S  [Команда для добавления в acc]
[101]P 0 L  [1]
[102]P 52 L [Указатель на первый элем массива]
[103]P 2 S  [k-й элем (сейчас 4-й)]
[104]P 3 S  [Длина массива]
[105]P 6 S  [1-й элем = 12]
[106]P 18 S [2-й элем = 36]
[107]P 55 S [3-й элем = 110]
[108]P 0 L  [4-й элем = 1]
[109]P 3 S  [5-й элем = 6]
[110]P 49 L [6-й элем = 99]
```

Рис. 3 Программа для загрузчика IO1 строки 68–110.

Выполним запуск программы. Массив чисел представлен на рисунке 4.

```
[105]P 6 S  [1-й элем = 12]
[106]P 18 S [2-й элем = 36]
[107]P 55 S [3-й элем = 110]
[108]P 0 L  [4-й элем = 1]
[109]P 3 S  [5-й элем = 6]
[110]P 49 L [6-й элем = 99]
```

Рис. 4 Исходный массив.

Результаты работы программы можно видеть на рисунках 5–13.

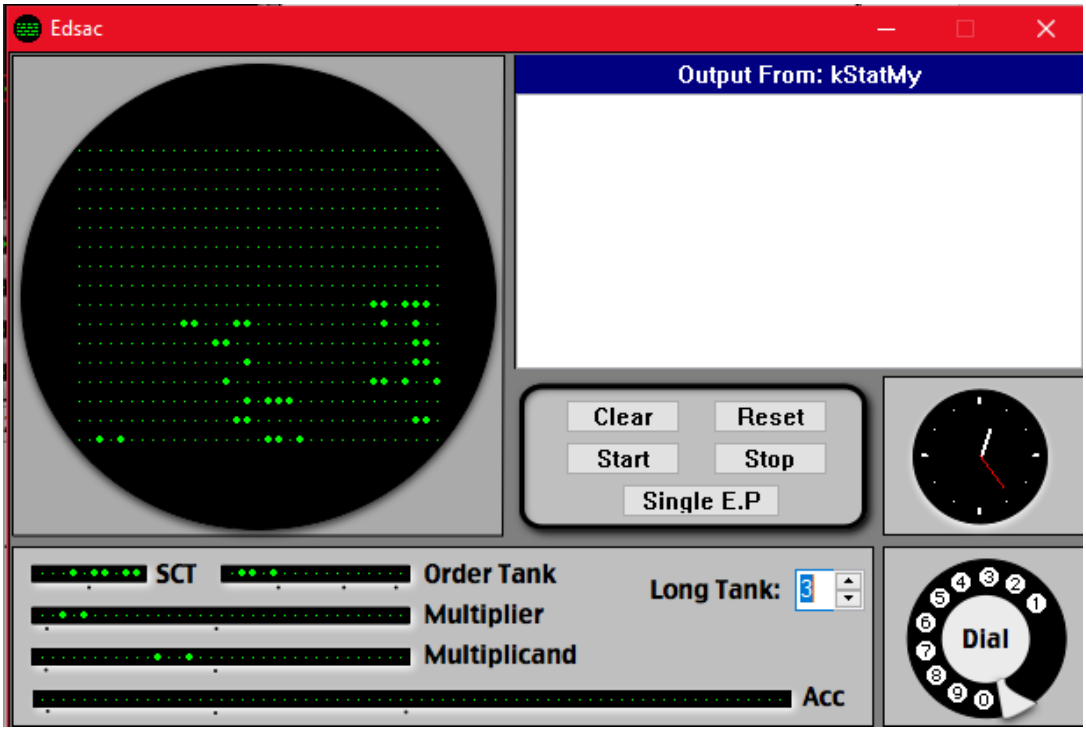

Рис. 5 Массив данных в симуляторе после выполнения программы.

WORD 105	Order = P 0 L	Integer 105S = 1	Fraction 104L = 0.00001525914
WORD 106	Order = P 3 S	Integer 106S = 6	Fraction 106S = 0.000092
WORD 107	Order = P 6 S	Integer 107S = 12	Fraction 106L = 0.00018310582
WORD 108	Order = P 18 S	Integer 108S = 36	Fraction 108S = 0.000549
WORD 109	Order = P 49 L	Integer 109S = 99	Fraction 108L = 0.00151062221
WORD 110	Order = P 55 S	Integer 110S = 110	Fraction 110S = 0.001678

Рис. 6-11 Каждый элемент массива данных после выполнения программы.

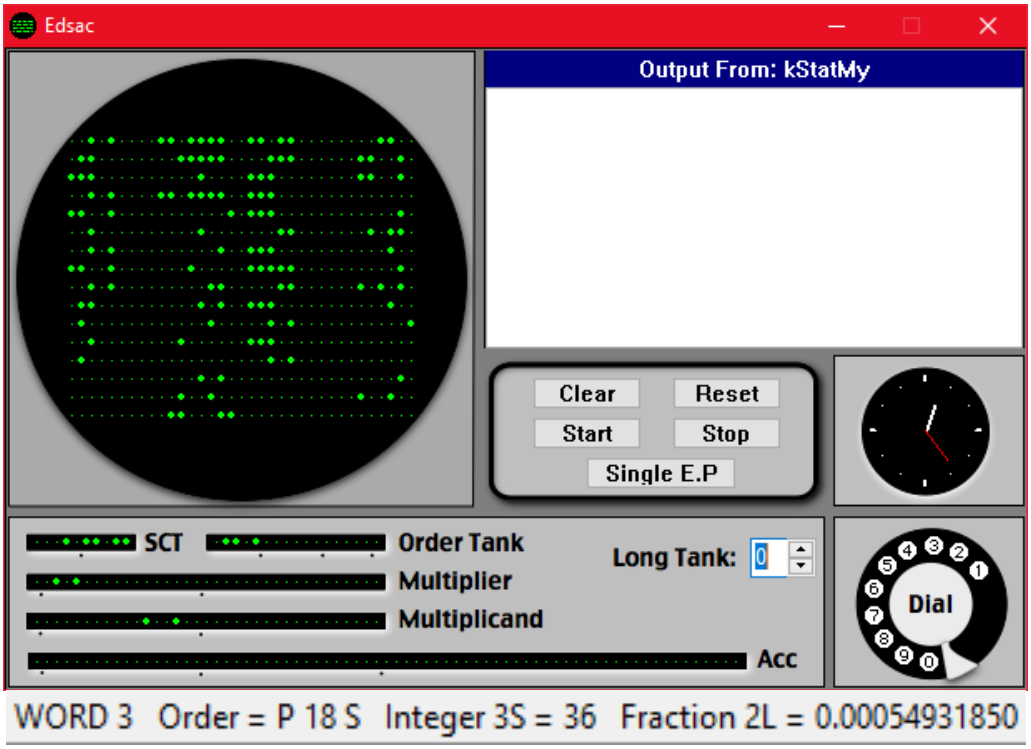

Рис. 12 и 13 Результат выполнения программы.

Вывод по IO1

Результаты выполнения программы на загрузчике IO1 полностью совпадают с ожидаемыми.

Построение программы для загрузчика Initial Orders 2

Основным отличием загрузчиков Initial Orders 2 от Initial Orders 1 является допустимость подпрограмм, где происходит относительная адресация ячеек, существенно упрощающая программирование. Абсолютная адресация обозначается буквой F, относительная @. Также немного меняется синтаксис: S → F, L → D.

```
T 55 K          [55 -  начальная ячейка]
GK              [@ = 55]
                [ВХОДНЫЕ ДАННЫЕ]
[@+0]P 28 D     [Указатель на 1 элемент массива] [55]
[@+1]P 3 F      [Длина массива]                  [56]
[@+2]P 6 F      [1-й элем = 12]                  [57]
[@+3]P 18 F     [2-й элем = 36]                  [58]
[@+4]P 55 F     [3-й элем = 110]                 [59]
[@+5]P 0 D      [4-й элем = 1]                   [60]
[@+6]P 3 F      [5-й элем = 6]                   [61]
[@+7]P 49 D     [6-й элем = 99]                  [62]
[@+8]T 0 F      [Команда чтения из acc]          [63]
[@+9]A 0 F      [Команда записи в acc]           [64]
[@+10]P 0 D     [1]                              [65]
[@+11]P 0 D     [i]                              [66]
[@+12]P 0 F     [j]                              [67]
[@+13]P 2 F     [k-й элемент (сейчас 4-й)]       [68]

                [Подпрограмма]
T 100 K         [100 - начальная ячейка]
GK              [@ = 100]
[@+0]A 3 F      [Формирование инструкции возврата в acc]
[@+1]T 58 @     [Указатель на точку выхода из подпрограммы]
                [Цикл i]
[@+2]T 0 F      [Обнуление acc]
[@+3]A 66 F     [Берём i]
[@+4]A 55 F     [Добавляем указатель на 1 элемент массива]
[@+5]L 0 D      [Сдвиг acc на 1 разряд влево]
[@+6]X 0 F
[@+7]A 64 F     [Добавляем команду A 0 F из ячейки 64]
[@+8]T 9 @      [Записываем результат в ячейку 9 подпрограммы, обнуляем acc]
[@+9]A 0 F      [Прочитали i-й элемент]
[@+10]T 1 F     [Положили его в рабочую ячейку 1 - insertElement]
[@+11]A 66 F    [Берём i]
[@+12]S 65 F    [-1]
[@+13]U 67 F    [Запоминаем i-1 в j]
                [ЦИКЛ j]
[@+14]A 55 F    [Берём указатель на 1 элемент массива]
[@+15]L 0 D     [Сдвиг acc на 1 разряд влево]
[@+16]A 64 F    [Добавляем команду A 0 F]
[@+17]T 18 @    [Записываем результат в ячейку 18 подпрограммы, обнуляем acc]
[@+18]A 0 F     [Прочитали list(j)]
[@+19]U 4 F     [Запомнили list(j) в рабочую ячейку 4]
[@+20]S 1 F     [Сравниваем с list(i)]
[@+21]G 35 @    [Если acc < 0 значит list(j) < list(i) следовательно конец цикла]
[@+22]T 0 F     [Обнуление acc]
[@+23]A 55 F    [Берём указатель на 1 элемент массива]
[@+24]A 67 F    [+j]
[@+25]A 65 F    [+1]
[@+26]L 0 D     [Сдвиг acc на 1 разряд влево]
[@+27]A 63 F    [Добавляем команду T 0 F]
[@+28]T 30 @    [Записываем результат в ячейку 30 подпрограммы, обнуляем acc]
[@+29]A 4 F     [Берём list(j) из рабочей ячейки 4]
[@+30]T 0 F     [list(j+1) = list(j), обнуляем acc]
[@+31]A 67 F    [Берём j]
[@+32]S 65 F    [-1]
[@+33]U 67 F    [Запомнили j--]
[@+34]E 14 @    [Вернулись в цикл с j]
                [КОНЕЦ ЦИКЛА j]
```

Рис. 14 программа для загрузчика IO2 (1).

```
                [КОНЕЦ ЦИКЛА j]
[@+35]T 0 F     [Обнуление acc]
[@+36]A 67 F    [Берём j]
[@+37]A 55 F    [Добавляем указатель на 1 элемент массива]
[@+38]A 65 F    [+1]
[@+39]L 0 D     [Сдвиг acc на 1 разряд влево]
[@+40]A 63 F    [Добавляем команду T 0 F]
[@+41]T 43 @    [Записываем результат в ячейку 43 подпрограммы, обнуляем acc]
[@+42]A 1 F     [Берём insertElement из рабочей ячейки 1]
[@+43]T 0 F     [list(j+1) = insertElement, обнуляем acc]
[@+44]A 66 F    [Берём i]
[@+45]A 65 F    [+1]
[@+46]U 66 F    [Запоминаем i++]
[@+47]S 56 F    [i - length]
[@+48]G 2 @     [Если i < length, то переходим в начало цикла]
                [СОРТИРОВКА ЗАКОНЧЕНА]
[@+49]T 0 F     [Обнуление acc]
[@+50]A 55 F    [Берём указатель]
[@+51]A 68 F    [+k]
[@+52]S 65 F    [-1]
[@+53]L 0 D     [Сдвиг acc на 1 разряд влево]
[@+54]A 64 F    [Добавление команды A 0 F]
[@+55]T 56 @    [Записываем результат в ячейку 56 подпрограммы, обнуляем acc]
[@+56]A 0 F     [k-й элем отсортированного массива]
[@+57]T 5 F     [Вывод в рабочую ячейку 5]
[@+58]Z 0 F     [Остановка машины]

                [Программа]
T 200 K         [программа начинается с 200 ячейки]
GK              [@ = 200]
[@+0]T 0 F      [Обнуление acc]
[@+1]A 2 @      [Запись в аккумулятор текущего адреса для]
                [формирования в подпрограмме инструкции возврата]
[@+2]X 0 F
[@+3]G 100 F    [Вызов подпрограммы]
[@+4]X 0 F
[@+5]EZPF       [Переход к в первой инструкции программы]
```

Рис. 15 программа для загрузчика IO2 (2).

```
[@+2]P 6 F      [1-й элем = 12]                 [57]
[@+3]P 18 F     [2-й элем = 36]                 [58]
[@+4]P 55 F     [3-й элем = 110]                [59]
[@+5]P 0 D      [4-й элем = 1]                  [60]
[@+6]P 3 F      [5-й элем = 6]                  [61]
[@+7]P 49 D     [6-й элем = 99]                 [62]
```

Рис. 16 Исходный массив.

Результаты работы программы видно на рисунках 17–25.

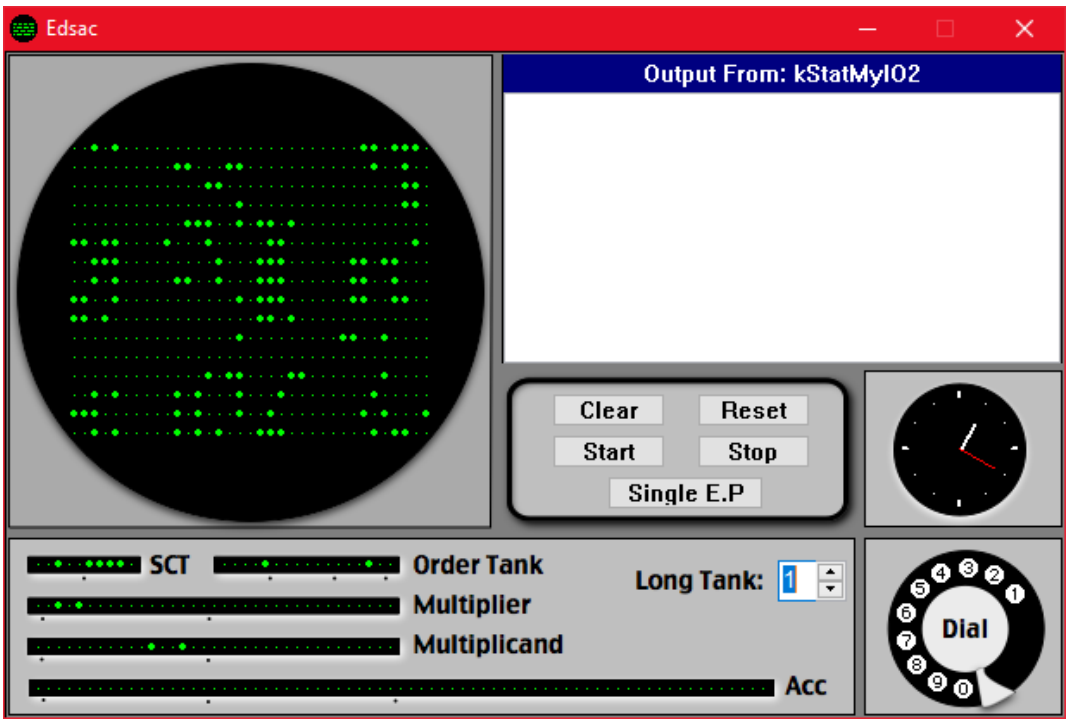

Рис. 17 Массив данных после выполнения программы

WORD 57 Order = P 0 D Integer 57F = 1 Fraction 56D = 0.00001525914
WORD 58 Order = P 3 F Integer 58F = 6 Fraction 58F = 0.000092
WORD 59 Order = P 6 F Integer 59F = 12 Fraction 58D = 0.00018310582
WORD 60 Order = P 18 F Integer 60F = 36 Fraction 60F = 0.000549
WORD 61 Order = P 49 D Integer 61F = 99 Fraction 60D = 0.00151062221
WORD 62 Order = P 55 F Integer 62F = 110 Fraction 62F = 0.001678

Рис. 18-23 Каждый элемент массива данных после выполнения работы.

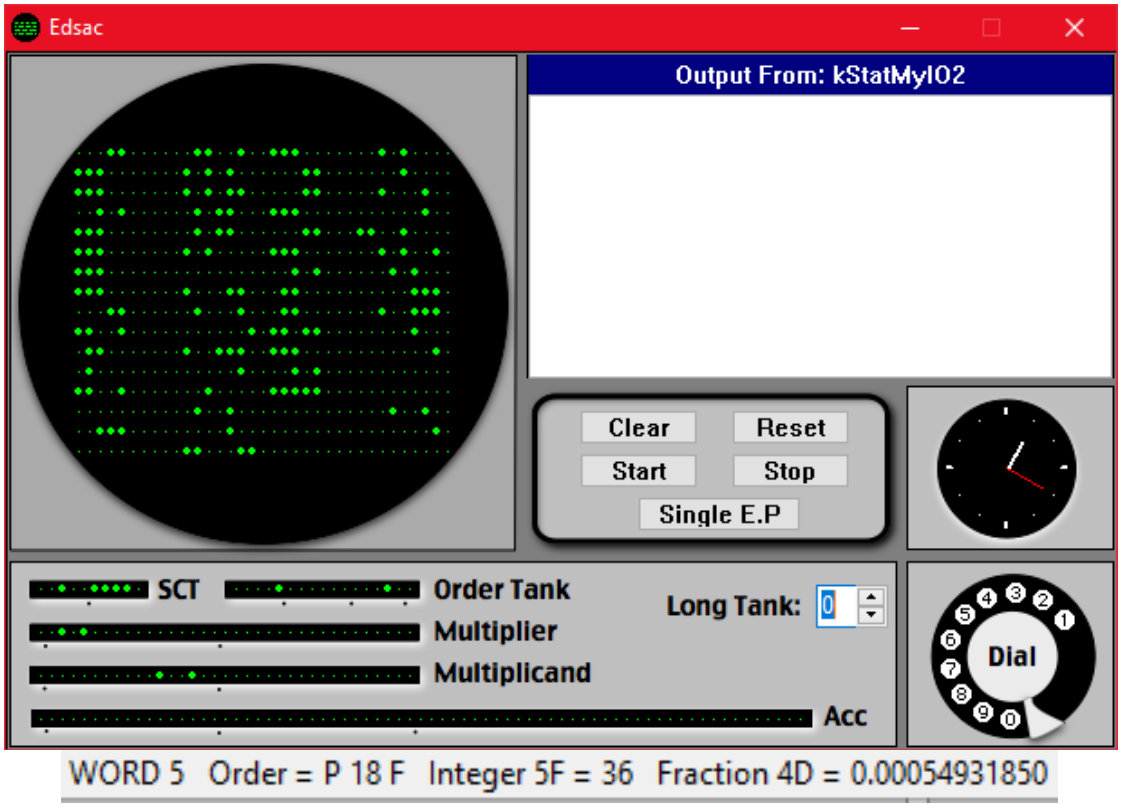

Рис. 24 и 25 Результаты работы программы.

Вывод

В ходе данной работы был осуществлён алгоритм сортировки вставками и вывод k-ого элемента массива в двух загрузчиках машины EDSAC. Результаты полностью соответствуют ожидаемым.