

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. 3530901/000004

_____ Кручинин К. А.
(подпись)

Руководитель

_____ Соболев 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.

```
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 асс]
[33]A 98 S   [Берём i из ячейки 98 и записываем в асс]
[34]X 0 S
[35]A 102 S  [Берём указатель на первый элемент массива из ячейки 102 и записываем в асс]
[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 асс 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 асс < 0 значит list(j) < list(i), следовательно идём в конец цикла]
[54]X 0 S
[55]T 0 S    [Обнуление асс]
[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 асс на 1 разряд влево]
[60]A 97 S   [Добавляем команду T 0 S]
[61]T 63 S   [Записываем в ячейку 63, обнулем асс]
[62]A 2 S    [Кладём в асс 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 Ю1 строки 31–67.

```

[KОНЕЦ ЦИКЛА j]
[68]T 0 S [Обнуляем асс]
[69]A 99 S [Верём j]
[70]A 102 S [Добавляем указатель на 1 элемент массива]
[71]A 101 S [+1]
[72]L 0 L [Сдвиг асс на 1 разряд влево]
[73]A 97 S [Добавляем команду T 0 S]
[74]T 76 S [Записываем всё из асс в 76 ячейку, обнуляем асс]
[75]A 1 S [Верём insertElement]
[76]T 0 S [list(j+1) = insertElement]
[77]A 98 S [Верём 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 асс]
[83]A 102 S [Верём указатель на первый элемент массива]
[84]A 103 S [+k]
[85]S 101 S [-1]
[86]L 0 L [Сдвиг асс 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 асс]
[98]P 0 L [i]
[99]P 0 S [j]
[100]A 0 S [Команда для добавления в асс]
[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 Ю1 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.

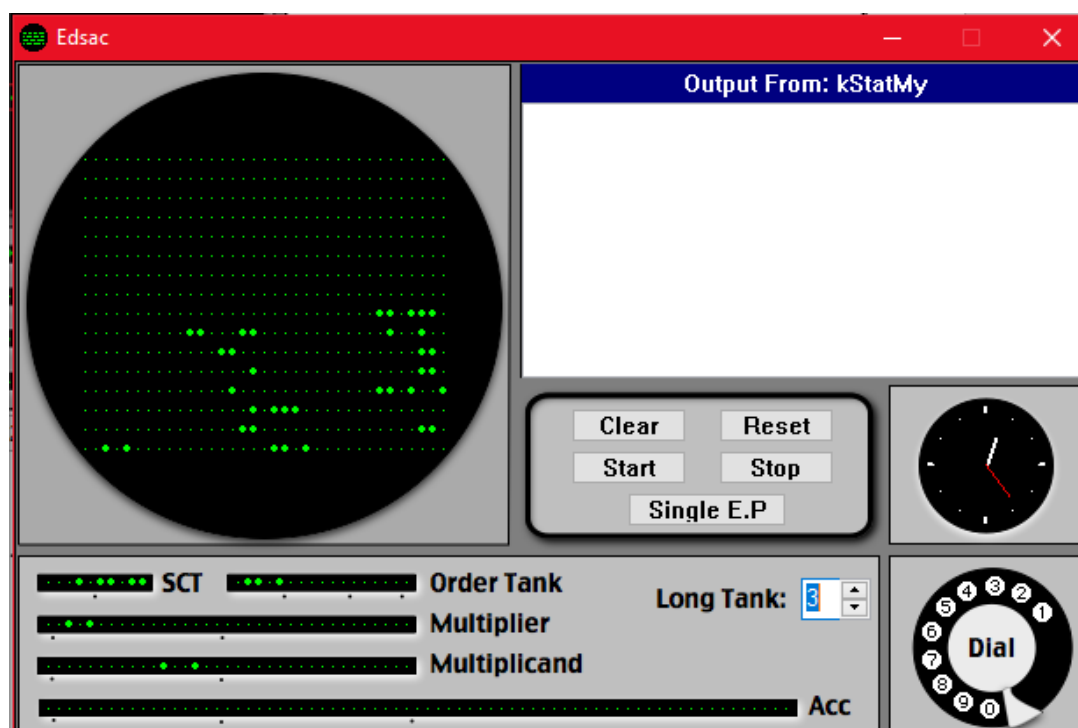


Рис. 5 Массив данных в симуляторе после выполнения программы.

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.

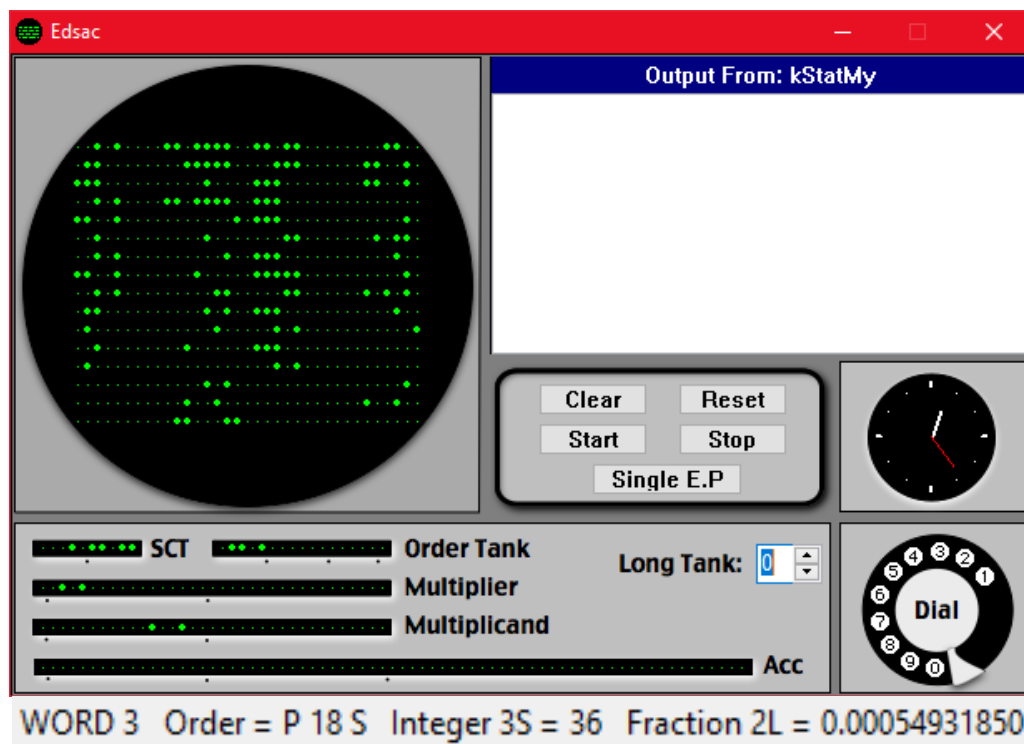


Рис. 12 и 13 Результат выполнения программы.

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: $S \rightarrow F$, $L \rightarrow D$.

	[[ВХОДНЫЕ ДАННЫЕ]	
T 55 K	[55 - начальная ячейка]	
GK	[@ = 55]	
[@+0]X 0 F	[Будущий указатель на 1 элемент массива]	[55]
[@+1]P 3 F	[Длина массива]	[56]
[@+2]X 0 F	[Будущий k-й элемент]	[57]
[@+3]P 0 D	[1]	[58]
[@+4]P 0 D	[i]	[59]
[@+5]P 0 F	[j]	[60]
[@+6]A 0 F	[Команда записи в асс]	[61]
[@+7]T 0 F	[Команда чтения из асс]	[62]
	[МАССИВ]	
T 300 K	[300 - начальная ячейка]	
GK	[@ = 300]	
[@+0]P 6 F	[1-й элем = 12]	[300]
[@+1]P 18 F	[2-й элем = 36]	[301]
[@+2]P 55 F	[3-й элем = 110]	[302]
[@+3]P 0 D	[4-й элем = 1]	[303]
[@+4]P 3 F	[5-й элем = 6]	[304]
[@+5]P 49 D	[6-й элем = 99]	[305]
	[Подпрограмма]	
T 100 K	[100 - начальная ячейка]	
GK	[@ = 100]	
[@+0]A 3 F	[Формирование инструкции возврата в асс]	
[@+1]T 58 @	[Указатель на точку выхода из подпрограммы]	
	[Цикл i]	
[@+2]T 0 F	[Обнуление асс]	
[@+3]A 59 F	[Берём i]	
[@+4]A 55 F	[Добавляем указатель на 1 элемент массива]	
[@+5]L 0 D	[Сдвиг асс на 1 разряд влево]	
[@+6]X 0 F		
[@+7]A 61 F	[Добавляем команду A 0 F из ячейки 64]	
[@+8]T 9 @	[Записываем результат в ячейку 9 подпрограммы, обнуляем асс]	
[@+9]A 0 F	[Прочитали i-й элемент]	
[@+10]T 1 F	[Положили его в рабочую ячейку 1 - insertElement]	
[@+11]A 59 F	[Берём i]	
[@+12]S 58 F	[-1]	
[@+13]U 60 F	[Запоминаем i-1 в j]	
	[ЦИКЛ j]	
[@+14]A 55 F	[Берём указатель на 1 элемент массива]	
[@+15]L 0 D	[Сдвиг асс на 1 разряд влево]	
[@+16]A 61 F	[Добавляем команду A 0 F]	
[@+17]T 18 @	[Записываем результат в ячейку 18 подпрограммы, обнуляем асс]	
[@+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 асс < 0 значит list(j) < list(i) следовательно конец цикла]	
[@+22]T 0 F	[Обнуление асс]	
[@+23]A 55 F	[Берём указатель на 1 элемент массива]	
[@+24]A 60 F	[+j]	
[@+25]A 58 F	[+1]	
[@+26]L 0 D	[Сдвиг асс на 1 разряд влево]	
[@+27]A 62 F	[Добавляем команду T 0 F]	
[@+28]T 30 @	[Записываем результат в ячейку 30 подпрограммы, обнуляем асс]	
[@+29]A 4 F	[Берём list(j) из рабочей ячейки 4]	
[@+30]T 0 F	[list(j+1) = list(j), обнуляем асс]	
[@+31]A 60 F	[Берём j]	
[@+32]S 58 F	[-1]	
[@+33]U 60 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 асс]
[@+36]A 60 F	[Берём j]
[@+37]A 55 F	[Добавляем указатель на 1 элемент массива]
[@+38]A 58 F	[+1]
[@+39]L 0 D	[Сдвиг асс на 1 разряд влево]
[@+40]A 62 F	[Добавляем команду T 0 F]
[@+41]T 43 @	[Записываем результат в ячейку 43 подпрограммы, обнуляем асс]
[@+42]A 1 F	[Берём insertElement из рабочей ячейки 1]
[@+43]T 0 F	[list(j+1) = insertElement, обнуляем асс]
[@+44]A 59 F	[Берём i]
[@+45]A 58 F	[+1]
[@+46]U 59 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 асс]
[@+50]A 55 F	[Берём указатель]
[@+51]A 57 F	[+k]
[@+52]S 58 F	[-1]
[@+53]L 0 D	[Сдвиг асс на 1 разряд влево]
[@+54]A 61 F	[Добавление команды A 0 F]
[@+55]T 56 @	[Записываем результат в ячейку 56 подпрограммы, обнуляем асс]
[@+56]A 0 F	[k-й элем отсортированного массива]
[@+57]T 5 F	[Вывод в рабочую ячейку 5]
[@+58]E 0 F	[Возврат в программу]
	[Программа]
T 200 K	[программа начинается с 200 ячейки]
GK	[@ = 200]
[@+0]T 0 F	[Обнуление асс]
[@+1]A 8 @	[Берём указатель на 1 элемент данного массива из 8 ячейки этой программы]
[@+2]T 55 F	[Записываем указатель в 55 ячейку (подпрограмма констант)]
[@+3]A 9 @	[Берём k-й элемент из 9 ячейки этой программы]
[@+4]T 57 F	[Записываем k-й элемент в 57 ячейку (подпрограмма констант)]
[@+5]A 5 @	[Запись в аккумулятор текущего адреса для]
	[формирования в подпрограмме инструкции возврата]
[@+6]G 100 F	[Вызов подпрограммы]
[@+7]Z 0 F	[Конец программы]
[@+8]P 150 F	[Указатель на 1 элемент массива]
[@+9]P 2 F	[k-й элемент]
[@+10]EZPF	[Переход к первой инструкции программы]

Рис. 15 программа для загрузчика IO2 (2).

	[МАССИВ]	
T 300 K	[300 - начальная ячейка]	
GK	[@ = 300]	
[@+0]P 6 F	[1-й элем = 12]	[300]
[@+1]P 18 F	[2-й элем = 36]	[301]
[@+2]P 55 F	[3-й элем = 110]	[302]
[@+3]P 0 D	[4-й элем = 1]	[303]
[@+4]P 3 F	[5-й элем = 6]	[304]
[@+5]P 49 D	[6-й элем = 99]	[305]

Рис. 16 Исходный массив.

Результаты работы программы видно на рисунках 17–25.

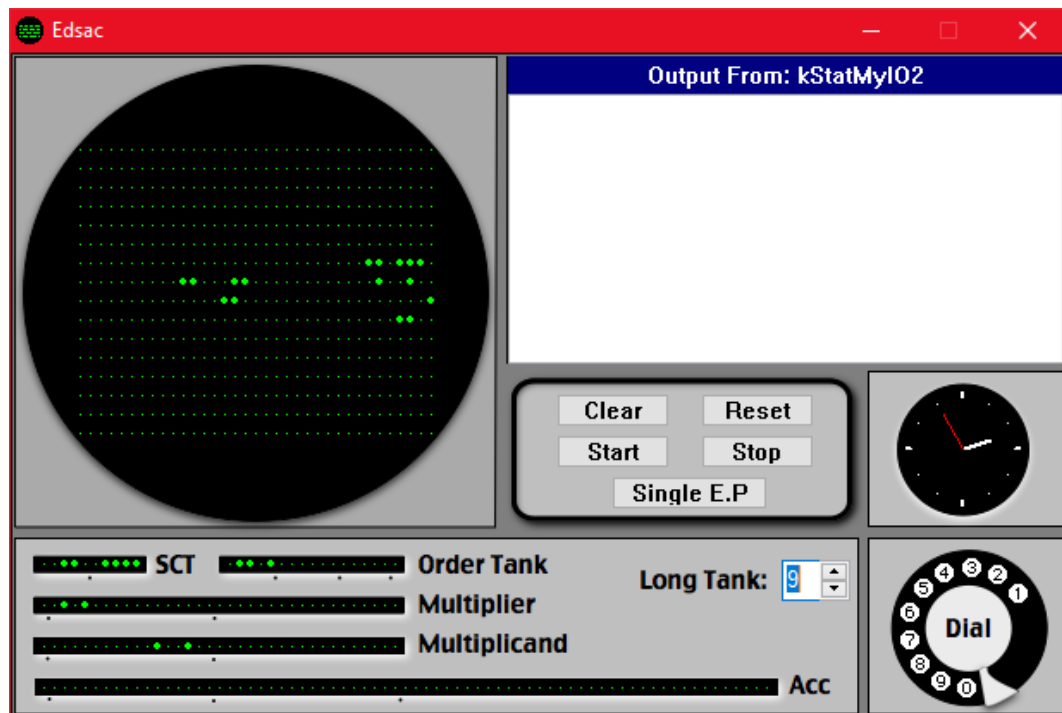


Рис. 17 Массив данных после выполнения программы

WORD 300	Order = P 0 D	Integer 300F = 1	Fraction 300F = 0.000015
WORD 301	Order = P 3 F	Integer 301F = 6	Fraction 300D = 0.00009155279
WORD 302	Order = P 6 F	Integer 302F = 12	Fraction 302F = 0.000183
WORD 303	Order = P 18 F	Integer 303F = 36	Fraction 302D = 0.00054931710
WORD 304	Order = P 49 D	Integer 304F = 99	Fraction 304F = 0.001511
WORD 305	Order = P 55 F	Integer 305F = 110	Fraction 304D = 0.00167847256

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

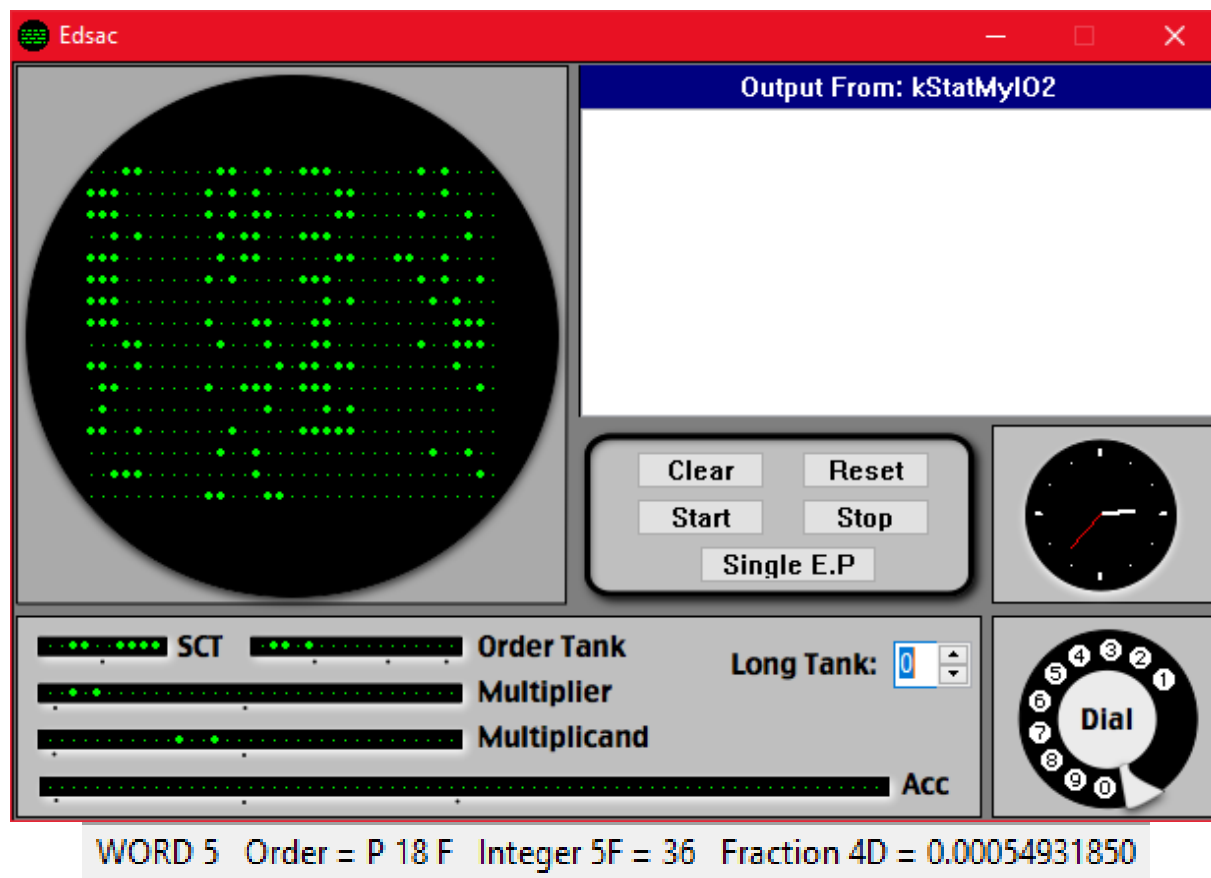


Рис. 24 и 25 Результаты работы программы.

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.