

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

Отчёт по лабораторной работе № 2

Дисциплина: Низкоуровневое программирование

Вариант 6.

Выполнил студент гр. 3530901/90002 _____ М.В. Дергачев
(подпись)

Принял преподаватель _____ Д.С. Степанов
(подпись)

“ ” _____ 2021 г.

Санкт-Петербург

2021

Задача

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.

Вариант задания

По варианту номер 6 необходимо реализовать нахождение медианы массива in-place.

Решение

Медиана – элемент массива, который находится ровно посередине после сортировки. Для выполнения поставленной задачи программа проходит по массиву, сравнивая каждое число с каждым, прибавляя к счетчику 1, если второе число больше текущего; вычитая 1, если число меньше текущего и не делая ничего, если они равны.

В конце каждого прохода проверяется значение счетчика, если он меньше 0 или больше 1 (количество чисел больше текущего меньше, чем количество чисел меньше текущего и наоборот, соответственно), то программа за место текущего числа берет следующее из массива и начинает проход заново, если значение счетчика равно 0 или 1 (0, когда в массиве нечетное количество элементов и 1, когда - четное), то медиана найдена и программа, записав ее в 0-ую ячейку, завершает работу.

Initial Orders 1

Листинг программы находится в приложении 1.

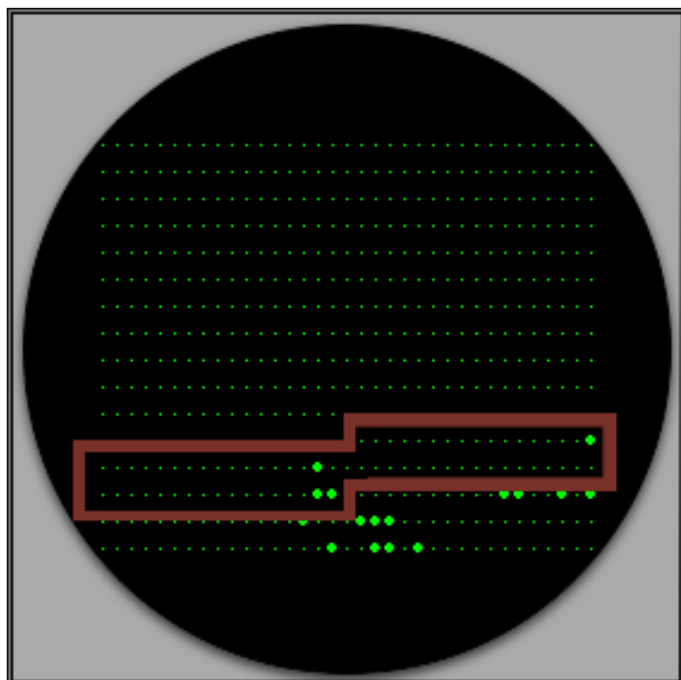


Рис. 1. Исходный массив {3; 0; 2; 1}

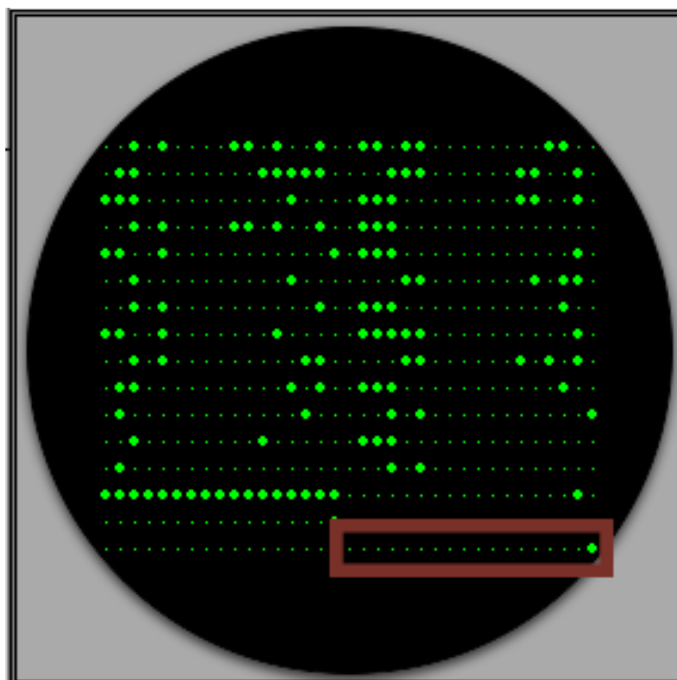


Рис. 2. Результат (1)

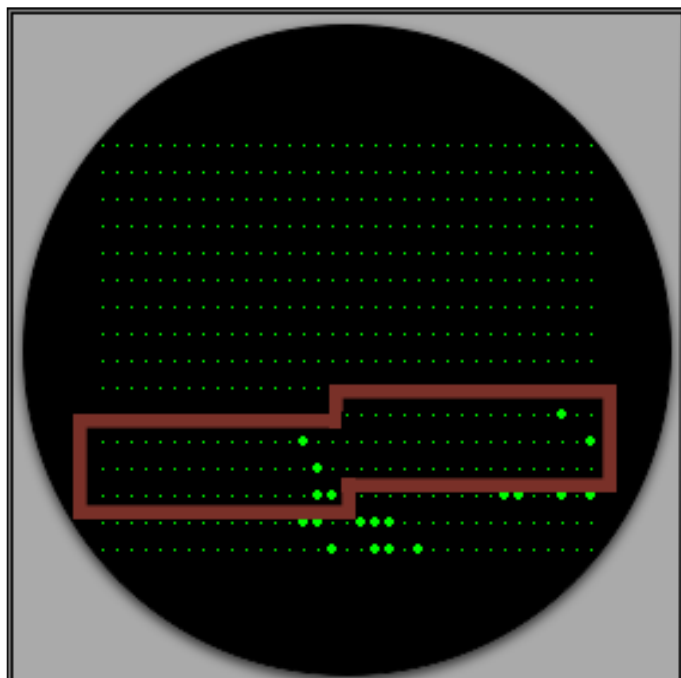


Рис. 3. Исходный массив {3; 0; 2; 1; 4; 4}

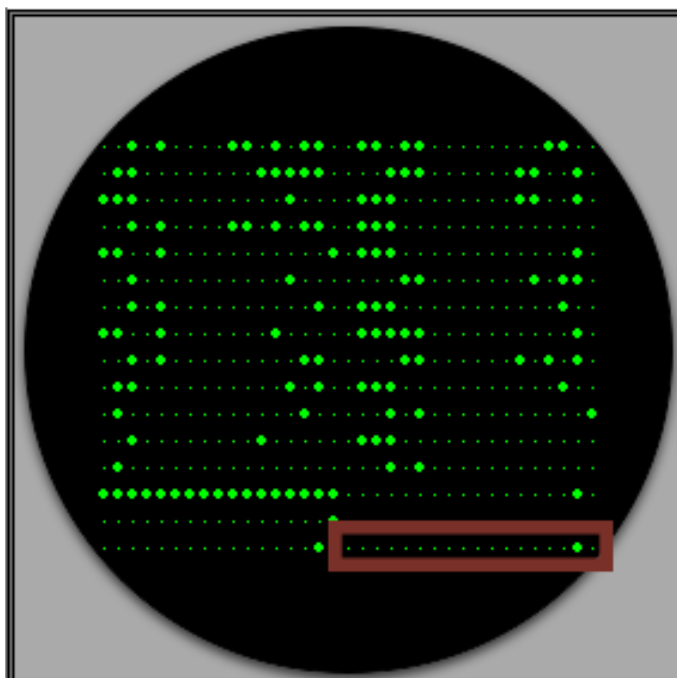


Рис. 4. Результат (2)

Initial Orders 2

С учётом возможностей загрузчика Initial Orders 2 была написана программа, листинг которой находится в приложении 1.

После была переписана программа из Initial Orders 1 с учётом директив Initial Orders 2.

Руководство программиста:

Разделы [address] [length] [array] отвечают за установку параметров для подпрограммы.

[sub] – подпрограмма нахождения медианы. Далее приведу пояснения по коду:

[0-1] Запись адреса возврата для выхода из подпрограммы.

[2-9] Запись параметров в выделенные для этого ячейки.

[10-13] Проверка «не пора ли закончить работу»

[14-15] Запись элемента массива в рабочую ячейку памяти

[16-19] Обновление счетчика, проверка «не прошли ли мы весь массив»

[20-33] Основной функционал: увеличение счетчика если $M > N$

[34-46] Проверяем, нашли ли мы медиану, готовимся к следующему шагу

[47-57] Обновляем адреса, готовимся к следующему шагу

[58-59] Выход из подпрограммы.

[60-63] Шаблоны команд подпрограммы.

[test prog] – тестовая программа, которая вызывает замкнутую подпрограмму.

Вывод

В ходе выполнения данной лабораторной работы был получен опыт программирования на EDSAC и работы с двумя его загрузчиками.

Приложение 1

Листинг initial orders 1. Файл прилагается к отчету.

1	T 106 S	[начало...]
2	X 0 S	[для пошаговой отладки использовать Z 0 S]
3	T 0 S	[обнуляем аккум, загружаем в 0-ую ячейку ноль]
4	T 3 S	[загружаем в 3 ячейку ноль (самый важный счетчик)]
5	A [len] 99 S	[Загружаем в аккум длину обрабатываемого массива	[i]
6	T 1 S	[Записываем в ячейку 1]
7	A [len] 99 S	[Загружаем в аккум длину обрабатываемого массива	[j]
8	T 2 S	[Записываем в ячейку 2]
9	A [addr] 100 S	[загружаем в аккум адрес 0-го элемента массива]
10	L 0 L	[Сдвигаем аккум на 1 разряд влево]
11	A [<r1>] 51 S	[прибавляем код инструкции]
12	T [<r1>] 51 S	[ставим на место инструкцию с новым адресом]
13	[f:] A [addr] 100 S	[43] [загружаем в аккум адрес 0-го элемента массива]
14	L 0 L	[сдвигаем аккум на 1 разряд влево]
15	A [<r2>] 57 S	[прибавляем код инструкции]
16	T [<r2>] 57 S	[ставим на место инструкцию с новым адресом]
17	[loop:] A 1 S	[47] [загружаем счетчик необработанных элементов массива (i)]
18	S [one] 97 S	[вычитаем 1]
19	G [<exit>] 96 S	[если результат меньше 0, завершаем работу]
20	T 1 S	[ставим на место значение счетчика и обнуляем аккум (i)]
21	[r1:] A 0 S	[51] [загружаем в аккум значение из ячейки N]
22	T 0 S	[записываем в рабочую ячейку, обнуляем аккум]
23	[loop2] A 2 S [j]	[53] [загружаем в аккум значение счетчика j]
24	S [one] 97 S	[вычитаем 1]
25	G 77 [<end loop2>] S	[если прошли все элементы -> на выход из цикла]
26	T 2 S	[ставим на место значение счетчика и обнуляем аккум (j)]
27	[r2:] A 0 S	[57] [загружаем в аккум значение из ячейки M	[array(j)]
28	S 0 S	[вычитаем N-ый элемента	[array(j)-array(i)]
29	G 67 [<f1>] S	[прыжок в 67 ячейку, если N > M]
30	S 97 [one] S	[вычитаем 1, чтобы проверить на равенство]
31	G 71 [<f2>] S	[прыжок в 71 ячейку, если N == M]
32	T 5 S	[если N < M, очищаем аккум и идем дальше]
33	A 3 S	[\]
34	A 97 [one] S	[самый важный счетчик++]
35	T 3 S	[/]
36	E 71 [<f2>] S	[пропускаем f1 и прыгаем в f2]
37	[f1:] T 5 S	[67] [очищаем аккум]
38	A 3 S	[\]
39	S 97 [one] S	[самый важный счетчик--]
40	T 3 S	[/]
41	[f2:] T 5 S	[71] [очищаем аккум]
42	A 97 [one] S	[загружаем в аккум 1]
43	L 0 L	[сдвигаем на 1 разряд влево]
44	A 57 [<r2>] S	[прибавляем код инструкции, исполненной на предыдущем шаге]
45	T 57 [<r2>] S	[записываем сформированную инструкцию в память]
46	E 53 [<loop2>] S	[повторяем все операции; аккумулятор обнулен]
47	[end loop2:] T 5 S	[77] [очищаем аккум]
48	A 3 S	[загружаем значение счетчика из 3 ячейки]
49	G 84 [<f3>] S	[если самый важный счетчик < 0, то -> (f3) (возврат в loop1)]]
50	S 97 [one] S	[\ иначе -> выходим (-> 96)]
51	G 96 [<exit>] S	[]
52	S 97 [one] S	[]
53	G 96 [<exit>] S	[/]
54	[f3:] T 5 S	[84] [очищаем аккум]
55	A 97 [one] S	[загружаем в аккум 1]
56	L 0 L	[сдвиг на 1 разряд влево]
57	A 51 [<r1>] S	[прибавляем код инструкции, исполненной на предыдущем шаге]
58	T 51 [<r1>] S	[записываем сформированную инструкцию в память]
59	A 98 S	[загружаем в аккум команду с 0 адресом из строки 98]
60	T 57 [<r2>] S	[ставим кк на место]
61	T 3 S	[обнуляем самый важный счетчик]
62	A 99 [len] S	[загружаем длину массива в аккум]
63	T 2 S	[записываем во 2 ячейку (loop2)]
64	E 43 [<f>] S	[возврат в цикл]
65	[end loop:] [exit:] O 0 S	[конец...]
66	Z 0 S	[96] [останов]
67	[one] P 0 L	[97] [константа (1)]
68	A 0 S	[98] [команда для копирования]
69	[len] P 2 L	[99] [длина массива (5)]
70	[addr] P 50 L [101]	[100] [индекс первого элемента массива (101)]
71	[array] P 1 L [3]		
72	[array] P 0 S [0]		
73	[array] P 1 S [2]		
74	[array] P 0 L [1]		
75	[array] P 2 S [4]		

Листинг initial orders 2. Файл прилагается к отчету.

1 T 56 K	[начало...]
2 GK	["контрольная точка"]
3 [sub]		
4 [0:] A 3 F	[пролог: формируем код инструкции возврата в Acc]
5 [1:] T 59 [<ret>] @	[пролог: записываем инструкцию возврата]
6		
7 [2:] A 6 [address] F	[загружаем в аккумулятор адрес 0-го элемента массива]
8 [3:] A 62 [r1init:] @	[прибавляем код инструкции с нулевым полем адреса]
9 [4:] T 14 [<r1>] @	[записываем сформированную инструкцию, обнуляем аккумулятор]
10		
11 [5:] A 1 F	[загрузка в аккумулятор размера массива]
12 [6:] T 7 F	[запись сформированной инструкции, обнуление аккумулятора]
13		
14 [f:]		
15 [7:] A 6 [address] F	[загружаем в аккумулятор адреса 0-го элемента массива]
16 [8:] A 63 [r2init:] @	[прибавляем код инструкции с полем адреса 1]
17 [9:] T 20 [<r2>] @	[записываем сформированную инструкцию, обнуляем аккумулятор]
18		
19 [loop:]	[loop1-----]
20 [10:] A 1 F	[загружаем счетчик необработанных элементов массива i]
21 [11:] S 61 [one] @	[уменьшаем на 1]
22 [12:] G 58 [<exit>] @	[если результат меньше 0, завершаем работу]
23 [13:] T 1 F	[обновляем значение счетчика и обнуляем аккумулятор i]
24 [r1:]		
25 [14:] A 0 F	[загружаем в аккумулятор значение из ячейки N]
26 [15:] T 0 F	[записываем это значение в рабочую ячейку, обнуляем аккумулятор] [array(i)]
27		
28 [loop2]	[loop2-----]
29 [16:] A 2 F	[загружаем в аккумулятор значение счетчика j]
30 [17:] S 61 [one] @		
31 [18:] G 40 [<end loop2>] @		
32 [19:] T 2 F	[ставим на место значение счётчика j]
33 [r2:]		
34 [20:] A 0 F	[загружаем в аккумулятор значения из ячейки M] [array(j)]
35 [21:] S 0 F	[вычитаем N-ого элемент] [array(j)-array(i)]
36		
37 [22:] G 30 [<f1>] @	[прыжок если N > M]
38 [23:] S 61 [one] @		
39 [24:] G 34 [<f2>] @	[прыжок если N == M]
40		
41 [25:] T 5 F	[обнуляем аккумулятор]
42 [26:] A 4 F	[\]
43 [27:] A 61 [one] @	[самый важный счетчик++]
44 [28:] T 4 F	[/]
45 [29:] E 34 [<f2>] @	[пропускаем f1 и прыгаем в f2]
46		
47 [f1:]		
48 [30:] T 5 F	[обнуляем аккумулятор]
49 [31:] A 4 F	[\]
50 [32:] S 61 [one] @	[самый важный счетчик--]
51 [33:] T 4 F	[/]
52		
53 [f2:]		
54 [34:] T 5 F	[обнуляем аккумулятор]
55 [35:] A 61 [one] @		
56 [36:] L 0 D	[сдвигаем на 1 разряд влево]
57 [37:] A 20 [<r2>] @	[прибавляем код инструкции, исполненной на предыдущем шаге]
58 [38:] T 20 [<r2>] @	[записываем сформированную инструкцию в память]
59 [39:] E 16 [<loop2>] @	[повторяем все операции; аккумулятор обнулен]

60	[end loop2:]	[~~~~~]]
61	[40:] T 5 F	[обнуляем аккумуля]
62	[41:] A 4 F	[загружаем значение счетчика из 4 ячейки]
63	[42:] G 47 [<f3>] @	[если самый важный счетчик < 0, то -> (f3) (возврат в loop1)]
64	[43:] S 61 [one] @	[\ иначе -> выходим (->58)]
65	[44:] G 58 [<exit>] @	[]
66	[45:] S 61 [one] @	[]
67	[46:] G 58 [<exit>] @	[/]
68			
69	[f3:]		
70	[47:] T 5 F		
71	[48:] A 61 [one] @		
72	[49:] L 0 D	[сдвигаем на 1 разряд влево]
73	[50:] A 14 [<r1>] @	[прибавляем код инструкции, исполненной на предыдущем шаге]
74	[51:] T 14 [<r1>] @	[записываем сформированную инструкцию в память]
75			
76	[52:] A 60 [index0] @	[адрес нулевого элемента массива, запомненный при инициализации подпрограммы]
77	[53:] T 6 F	[записываем адрес массива в ячейку 0, обнуляем аккумулятор]
78			
79	[54:] T 4 F	[обнуляем самый важный счетчик]
80	[55:] A 7 [length] F	[размер массива, запомненный при инициализации подпрограммы]
81	[56:] T 2 F	[записываем в 2 ячейку (for loop2)]
82	[57:] E 7 [<f>] @		
83	[end loop:]	[-----]
84	[exit:]		
85	[58:] T 5 F	[обнуляем аккумуля]
86	[ret:]		
87	[59:] E 0 F	[эпилог: инструкция возврата из подпрограммы]
88	[60:] [index0] P 133 F		
89	[61:] [one] P 0 D	[константа 1]
90	[62:] [r1init:] A 0 F	[основа для формирования инструкции с меткой r1]
91	[63:] [r2init:] A 0 F	[основа для формирования инструкции с меткой r2]
92			
93			
94			
95	[test prog]		
96	GK		
97	[0:] X 0 F		
98	[1:] A 11 [address] @	[адрес массива]
99	[2:] T 6 F	[запись адреса массива в ячейку 0, обнуление аккумулятора]
100	[3:] T 4 F		
101	[4:] A 12 [length] @	[длина массива]
102	[5:] T 1 F	[запись длины массива в ячейку 1, обнуление аккумулятора (i)]
103	[6:] A 12 [length] @	[длина массива]
104	[7:] T 2 F	[запись длины массива в ячейку 2, обнуление аккумулятора (j)]
105	[8:] A 8 @	[\ вызов]
106	[9:] G 56 [<sub>] F	[/ подпрограммы]
107	[10:] Z 0 F	[останов]
108			
109	[address:]		
110	[11:] P 13 [<array>] @	[адрес массива = <Начало программы>+12 (см. ниже)]	
111			
112	[length:]		
113	[12:] P 2 F	[15]	
114			
115	[13: array:]		
116			
117	P 0 D [1]		
118	P 1 F [2]		
119	P 1 D [3]		
120	P 0 F [0]		
121			
122	EZ PF [директива IO2,переход к исполнению]		