Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

Низкоуровневое программирование

Отчет по лабораторной работе №1
EDSAC

Работу выполнил:
Кечин В.В.
Группа:
3530901/90004
Преподаватель:
Алексюк А.О.

Санкт-Петербург
2021

Содержание

1. Цель работы

- Разработать программу для EDSAC, реализующую сортировку выбором массива чисел in-place, 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.
- 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.

2. Программа работы

- Установить симулятор EDSAC
- Изучить и понять принцип работы EDSAC
- Разработать требуемые программы для EDSAC в соответствии с заданным вариантом

3. Теория

Initial Orders 1 – простая программа, которая решает следующую задачу: чтение символов (т.е., разумеется, кодов символов) с перфоленты, формирование кодов инструкций (и констант) и запись их в память. Initial Orders 1 состоит инструкций, которые загружаются в ячейки 0 – 30 и начинают исполняться после нажатия кнопки «Start». Программа, записанная на ленте, загружается в следующие друг за другом ячейки памяти, начиная с адреса 31. Первая инструкция (загружаемая в ячейку 31) обязательно должна иметь вид T <N+1> S, где N – адрес ячейки, в которую загружается последняя инструкция программы. Так, «пустая» программа состоит из 1 инструкции: T 32 S (результатом ее выполнения будет запись значения аккумулятора в ячейку 32).

Initial Orders 2 – значительно более сложная (семантически) программа, чем Initial Orders 1, имеющая, тем не менее, немногим больше инструкций (41 вместо 31), и решающая схожую задачу. Форма записи инструкции при использовании Initial Orders 2 незначительно отличается от формы записи инструкций в случае Initial Orders 1. Однако, в отличие от Initial Orders 1, при использовании Initial Orders 2 на ленте находятся не только инструкции (или константы) загружаемой программы, но и управляющие последовательности (“control combinations” в терминах Initial Order 2, современный термин – “директивы”), определяющее поведение самой программы Initial Order 2 в процессе загрузки. Директивы Initial Order 2 записываются в той же форме, что и инструкции загружаемой программы. Initial Orders 2 состоит из 41 инструкции, которые загружаются в ячейки 0 – 40 и начинают исполняться после нажатия кнопки «Start». Ячейки 41 – 43 являются для Initial Orders 2 рабочими. По умолчанию, программа, записанная на ленте, загружается в следующие друг за другом ячейки памяти, начиная с адреса 44. Однако, в отличие от Initial Orders 1, адрес загрузки в любой момент может быть изменен с помощью соответствующих директив.

4. Ход выполнения работы

Для удобного написания программы реализуем небольшие программы на языке Python для индексирования программ для загрузчиков IO1 и IO2. Общая идея программы для IO1: Вводим длину массива, адрес его первого элемента и сам массив. Далее в цикле до тех пор, пока у нас есть необработанные элементы, входим во внутренний цикл, в котором ищем максимальный элемент из оставшихся. После меняем максимальный элемент с последним необработанным. Листинг программы IO1:

```
[31] T 136[N+1] S

[32] X0S

[33] A 122 [len] S
[34] A 123 [1] S
[35] T 121 [cur_len] S

[36] A 102 [sw1] S
[37] A 125 [f_el] S
[38] A 122 [len] S
[39] A 122 [len] S
[40] S 124 [2] S
[41] T 102 [sw1] S   [set last_el]

[42] A 106 [sw2] S
[43] A 125 [f_el] S
[44] A 122 [len] S
[45] A 122 [len] S
[46] S 124 [2] S
[47] T 106 [sw2] S     [set last_el]

[48] A 71 [for_max] S
[49] A 125 [f_el] S
[50] T 71 [for_max] S   [set 1 el]

[51] A 0[f_el] S [fl]
[52] U 0[f_el] S [tfl]

[53] A 51 [fl] S
[54] A 125 [f_el] S
[55] T 51 [fl] S

[56] A 52 [tfl] S
[57] A 125 [f_el] S
[58] T 52 [tfl] S   указатели[ на первый элемент]

[loop]

[59] T 0 S [l1]

[60] A 121 [cur_len] S
[61] S 123 [1] S
[62] T 121 [cur_len] S    [-1 количество необработанных элементов]

[63] A 51 [fl] S
[64] U 86 [cur] S
[65] T 100 [ss1] S

[66] A 52 [tfl] S
```

```
[67] U 87 [cur_d] S
[68] T 103 [ss2] S          вовзвращаемся[ к первому элементу массива для поиска
    ↪ максимума]

[69] A 123 [1] S
[70] T 120 [count2] S       счетчик[ второго цикла = 1]

[71] A 0 F [for_max]
[72] T 118 [max] S          первый[ элемент = макс]

[loop2]
[73] T 0 S [l2]          обнуление[ acc]

[74] A 120 [count2] S
[75] S 121 [cur_len] S
[76] E 99 [l11] S           выход[ из цикла]
[77] A 121 [cur_len] S
[78] A 123 [1] S
[79] T 120 [count2] S     счетчик[2+=1]

[80] A 86 [cur] S     итерация[ по элементам]
[81] A 124 [2] S       итерация[ по элементам]
[82] T 86 [cur] S        итерация[ по элементам]
[83] A 87 [cur_d] S         итерация[ по элементам]
[84] A 124 [2] S              итерация[ по элементам]
[85] T 87 [cur_d] S             итерация[ по элементам]
[86] A 0 S [cur]      итерация[ по элементам]
[87] U 0 S [cur_d]    итерация[ по элементам]

[88] S 118 [max] S           [-max]
[89] G 73 [l2] S             возврат[ к первой строчке второго цикла если макс не
    ↪ поменялся]
[90] A 118 [max] S           [+max, получаем число, которое было]
[91] T 118 [max] S           присваиваем[ max значение]

[92] A 86 [cur] S
[93] T 100 [ss1] S         присваиваем[ max значение для swap]

[94] A 87 [cur_d] S
[95] T 103 [ss2] S          присваиваем[ max значение для swap]

[96] A 120 [count2] S
[97] S 121 [cur_len] S
[98] G 73 [l2] S      возврат[ к первой строчке второго цикла если не пройдены все
    ↪ значения]

[swap]

[99] T 0 S [l11]               [acc=0]
[100] A 0 S [ss1] загрузка[ в аккумулятор значения из ячейки с максимальным значением
    ↪ ]
[101] T 0 S           запись[ этого значения в рабочую ячейку, обнуление аккумулятора]
[102] A 0 S [sw1] загрузка[ в аккумулятор значения из последней ячейки]
[103] U 0 S [ss2] [ запись этого значения в ячейку с максимальным значением, обнуление
    ↪ аккумулятора]
[104] T 3 S
[105] A 0 S     загрузка[ в аккумулятор значения из ячейки 0]
[106] T 0 S [sw2] запись[ этого значения в последнюю, обнуление аккумулятора]

[107] A 102 [sw1] S
```

```
[108] S 124 [2] S
[109] T 102 [sw1] S

[110] A 106 [sw2] S
[111] S 124 [2] S
[112] T 106 [sw2] S          убираем[ обработанный элемент из дальнейшей обработки]

[113] A 119 [count] S
[114] A 123 [1] S
[115] U 119 [count] S
[116] S 122 [len] S
[117] G 59 [11] S     конец[ цикла или возврат к первому действию если не прошло
    ↪ итераций = длине]

[vars]
[118] P 0 S [max]
[119] P 0 L [count] [= 1]
[120] P 0 L [count2] [= 1]
[121] P 0 S [cur_len] [= 11]
[122] P 5 S [len] [=10]
[123] P 0 L [1]
[124] P 1 S [2]
[125] P 126 [<f_el>] S [f_el]

[array]
[126] P 0 S [<f_el>] [0]
[127] P 2 L    [5]
[128] P 1 S    [2]
[129] P 3 S    [6]
[130] P 4 L    [9]
[131] P 1 L    [3]
[132] P 0 L    [1]
[133] P 2 S    [4]
[134] P 4 S    [8]
[135] P 3 L    [7]
```

Теперь сделаем эту программу подпрограммой - будем ее вызывать из другого места в коде. Для этого добавим установку параметров в начальное положение (для многократного вызова), а также воспользуемся инструкциями загрузчика IO2.

```
T 56 K [ директива IO2, установка адреса загрузки ]
G K    [ директива IO2, фиксация начального адреса подпрограммы ]
[0] A 3 F [ пролог: формирование кода инструкции возврата в Acc ]
[1] T 89 [<ret>] @ [ пролог: запись инструкции возврата ]

[2] A 5[len] F
[3] A 94 [1] @
[4] T 93 [cur_len] @
[5] A 94 [1] @
[6] T 91 [count] @
[7] A 94 [1] @
[8] T 92 [count2] @      [set_params]

[9] A 69 [sw1] @
[10] A 4 F
[11] T 69 [sw1] @    [set last_el]

[12] A 73 [sw2] @
[13] A 4 F
[14] T 73 [sw2] @      [set last_el]

```

```
[15] A 38 [for_max] @
[16] A 1 F
[17] T 38 [for_max] @    [set 1 el]
[18] A 0[f_el] F [fl]
[19] U 0[f_el] F [tfl]

[20] A 18 [fl] @
[21] A 1 F
[22] T 18 [fl] @

[23] A 19 [tfl] @
[24] A 1 F
[25] T 19 [tfl] @      указатели[ на первый элменет]

[loop]

[26] T 0 F [l1]

[27] A 93 [cur_len] @
[28] S 94 [1] @
[29] T 93 [cur_len] @     [-1 количество необработанных элементов]

[30] A 18 [fl] @
[31] U 53 [cur] @
[32] T 67 [ss1] @

[33] A 19 [tfl] @
[34] U 54 [cur_d] @
[35] T 70 [ss2] @        вовзвращаемся[ к первому элементу массива для поиска
    ↪ максимума]

[36] A 94 [1] @
[37] T 92 [count2] @     счетчик[ второго цикла = 1]

[38] A 0[f_el] F [for_max]
[39] T 90 [max] @        первый[ элемент = макс]

[loop2]

[40] T 0 F [l2]        обнуление[ acc]

[41] A 92 [count2] @
[42] S 93 [cur_len] @
[43] E 66 [l11] @        [check cond]
[44] A 93 [cur_len] @
[45] A 94 [1] @
[46] T 92 [count2] @    счетчик[2+=1]

[47] A 53 [cur] @    итерация[ по элементам]
[48] A 2 F       итерация[ по элементам]
[49] T 53 [cur] @      итерация[ по элементам]
[50] A 54 [cur_d] @        итерация[ по элементам]
[51] A 2 F             итерация[ по элементам]
[52] T 54 [cur_d] @              итерация[ по элементам]

[53] A 0[f_el] F [cur]      итерация[ по элементам]
[54] U 0[f_el] F [cur_d]    итерация[ по элементам]

[55] S 90 [max] @          [-max]
[56] G 40 [l2] @            возврат[ к первой строчке второго цикла если макс не
```

```
    ↪ поменялся]
[57] A 90 [max] @          [+max, получаем число, которое было]
[58] T 90 [max] @           присваиваем[ max значение]

[59] A 53 [cur] @
[60] T 67 [ss1] @        присваиваем[ max значение для swap]

[61] A 54 [cur_d] @
[62] T 70 [ss2] @         присваиваем[ max значение для swap]

[63] A 92 [count2] @
[64] S 93 [cur_len] @
[65] G 40 [l2] @     возврат[ к первой строчке второго цикла если не пройдены все
    ↪ значения]

[swap]
[66] T 0 F [l11]              [acc=0]
[67] A 0[f_el] F [ss1] загрузка[ в аккумулятор значения из ячейки с max значением]
[68] T 0 F          запись[ этого значения в рабочую ячейку, обнуление аккумулятора]
[69] A 0[l_el] F [sw1] загрузка[ в аккумулятор значения из последней ячейки]
[70] U 0[f_el] F [ss2] запись[ этого значения в ячейку, где было max значение]
[71] T 6 F
[72] A 0 F     загрузка[ в аккумулятор значения из ячейки 0]
[73] T 0[l_el] F [sw2]  [ запись этого значения в последнюю ячейку]


[74] A 69 [sw1] @
[75] S 2 F
[76] T 69 [sw1] @

[77] A 73 [sw2] @
[78] S 2 F
[79] T 73 [sw2] @         убираем[ обработанный элемент из дальнейшей обработки]

[80] A 91 [count] @
[81] A 94 [1] @
[82] U 91 [count] @
[83] S 5[len] F
[84] G 26 [l1] @     конец[ цикла или возврат к первому действию если не прошло
    ↪ итераций = длине]

[85] T 0 F [ обнуление аккумулятора ] [exit:]

[86] T 1 F
[87] T 4 F
[88] T 5 F    [reset]

[89] E 0 F [<ret>] [ эпилог: инструкция возврата из подпрограммы ]

[90] P 0 F [max]
[91] P 0 F [count] [= 0]
[92] P 0 F [count2] [= 0]
[93] P 0 F [cur_len] [= ...]
[94] P 0 D [1]

G K       директива[ IO2фиксация, начального адреса программы ]
[0] X 0 F [ для пошаговой отладки использовать Z 0 F ]
[1] A 14 [<f_el>] @ [ адрес 1 элемента ]
[2] T 1 F [ запись адреса 1 эл ячейку 1, обнуление аккумулятора ]
[3] A 14 [<f_el>] @ [ адрес последнего элемента ]
```

```
[4] A 13 [<len>] @
[5] A 13 [<len>] @
[6] S 2 F
[7] T 4 F [ запись адреса посл эл в ячейку 2, обнуление аккумулятора ]
[8] A 13 [<len>] @ [ длина массива ]
[9] T 5 F [ запись длины массива в ячейку 3, обнуление аккумулятора ]
[10] A 10 вызов[] @ вызов[]
[11] G 56 [<sub>] F [/ подпрограммы ]
[12] Z 0 F [ останов ]

[13] P 5 D [<len>] [=11]
[14] P 15 [<f_el1>] @ [<f_el>]

[array:]
[15] P 4 F [<f_el1>]    [ 8 ]
[16] P 0 D    [ 1 ]
[17] P 1 F    [ 2 ]
[18] P 1 D    [ 3 ]
[19] P 5 F    [ 10 ]
[20] P 2 D    [ 5 ]
[21] P 3 F    [ 6 ]
[22] P 4 D    [ 9 ]
[23] P 0 F    [ 0 ]
[24] P 3 D    [ 7 ]
[25] P 2 F    [ 4 ]
EZ PF директива[ IO2, переход к исполнению]
```

Обе программы реализуют заданный функционал, в чем можно удостовериться, запустив их в симуляторе EDSAC.

5. Выводы

Реализованы программы для загрузчиков IO1 и IO2, которые реализуют сортировку выбором массива чисел in-place.