

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
УЧРЕЖДЕНИЯ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
Кафедра интеллектуальных информационных технологий

Отчёт по лабораторной работе №7
по дисциплине Объектно-Ориентированные Технологии Программирования и Стандарты
Проектирования за I семестр
Тема: "Умные указатели"

Выполнил:
студент 2 курса
IV семестра
факультета ЭИС
группы ПО-4(1)
Галанин П. И.
«__» _____ 2021 г.

Проверил:
магистрант
кафедры ИИТ
Миндер А. В.
«__» _____ 2021 г.

Отчёт по лабораторной работе №7

Тема: «Умные указатели»

Цель работы: Изучить применение умных указателей.

Ход работы:

Вариант V задание 1

Условие: Одномерный массив. Удалить элемент с заданным номером.

Решение:

Листинг: main.cpp

```
1 #include <iostream>
2 #include <memory>
3
4 //функция варианта 5 задания 1
5 void option5_task1();
6 //функция напечатает массив
7 template <typename T>
8 void print_1d_array(T* arr, int size);
9 //функция сгенерирует массив из int элементов в промежутке от a до b
10 void generate_int_elements_in_1d_array(int* arr, int size, int a = 1, int b = 99);
11 //функция, которая удалит элемент из массива
12 template <typename T>
13 std::shared_ptr<T []> del_index_in_1d_array(
14     std::shared_ptr<T []> arr,
15     int *size,
16     int deleted_index
17 );
18
19 int main()
20 {
21     srand(100); //Seed рандома. Программа всегда будет давать один и тот же результат при рандоме
22     option5_task1();
23
24     return 0;
25 }
26
27 void option5_task1()
28 {
29     int size = 7; //размер массива
30     std::shared_ptr<int []> arr(new int[size]); //выделяем память под массив
31
32     generate_int_elements_in_1d_array(arr.get(), size); //генерируем числа в массив
33     print_1d_array(arr.get(), size); //печатаем массив
34
35     arr = move( //передаём владение
36         del_index_in_1d_array(move(arr), &size, 3) //возвратит умный указатель
37     );
38     print_1d_array(arr.get(), size); //печатаем массив
```

					ЛР.190333.ПО4.07 81 00		
Изм	Лист	№ докум.	Подп.	Дата	Отчёт по лабораторной работе №7 Умные указатели		
Разраб.	Галанин						
Пров.	Миндер						
Н. контр.							
Утв.							
					Лит.	Лист	Листов
					Л	2	10
					БрГТУ		

```

39 }
40
41 template <typename T>
42 void print_1d_array(T* arr, int size)
43 {
44     std::cout << "[" << std::endl;
45     for (int i = 0; i < size; i++)
46     {
47         std::cout << "\t" << i << ": " << arr[i] << "\n";
48     }
49     std::cout << "]" << std::endl;
50 }
51
52 void generate_int_elements_in_1d_array(int* arr, int size, int a, int b)
53 {
54     for (int i = 0; i < size; i++)
55     {
56         arr[i] = rand() % (b - a + 1) + a;
57     }
58 }
59
60 template <typename T>
61 std::shared_ptr<T []> del_index_in_1d_array(
62     std::shared_ptr<T []> arr,
63     int *size,
64     int deleted_index
65 )
66 {
67     if (deleted_index <= 0 || deleted_index >= *size)    //if index in (-00; 0] u [size; ++00]
68     {
69         std::cout << deleted_index << " not in (0; " << *size << ")\n";
70         return arr;
71     }
72
73     *size -= 1;    //уменьшаем размер массива
74     for (int i = deleted_index; i < *size; i++)    //от удаляемого элемента до конца
75     {
76         arr[i] = arr[i + 1];    //сдвигаем элементы
77     }
78
79     std::shared_ptr<T[]> temp_arr(new T[*size]);    //выделяем память под новый массив
80     for (int i = 0; i < *size; i++)
81     {
82         temp_arr[i] = arr[i];    //копируем элементы
83     }
84
85     arr = move(temp_arr);    //передаём права
86     return arr;
87 }

```

Листинг: Out

```

1 [
2     0: "12",
3     1: "24",
4     2: "45",
5     3: "39",
6     4: "71",
7     5: "29",
8     6: "47",

```

Изм	Лист	№ докум.	Подп.	Дата

ЛР.190333.ПО4.07 81 00

Лист

3

```

9 ]
10 [
11     0: "12",
12     1: "24",
13     2: "45",
14     3: "71",
15     4: "29",
16     5: "47",
17 ]

```

Вариант V задание 2

Условие: Двумерный массив. Добавить строку в начало матрицы

Решение:

Листинг: main.cpp

```

1 #include <iostream>
2 #include <memory>
3
4 //функция варианта 5 задания 2
5 void option5_task2();
6 //функция генерирует элементы для int массива
7 void generate_elements_in_2d_array(
8     const std::unique_ptr<std::unique_ptr<int[]>>> &arr,
9     const int rows,
10    const int cols,
11    int a = 1,
12    int b = 99
13);
14 //функция напечатает функцию
15 template <typename T>
16 void print_2d_array(
17     const std::unique_ptr<std::unique_ptr<T[]>>> &arr,
18     const int rows,
19     const int cols
20);
21 //функция вставит строку в функцию
22 template <typename T>
23 void add_line_to_2d_array(
24     std::unique_ptr<std::unique_ptr<T[]>>> &arr,
25     int *rows,
26     const int cols
27);
28
29 int main()
30 {
31     srand(100); //Seed рандома. Программа всегда будет давать один и тот же результат при рандоме
32     option5_task2();
33
34     return 0;
35 }
36
37 void option5_task2()
38 {
39     int rows = 5; //количество строк
40     int cols = 12; //количество столбцов
41     std::unique_ptr<
42         std::unique_ptr<int[]>>

```

Изм	Лист	№ докум.	Подп.	Дата

ЛР.190333.ПО4.07 81 00

Лист

4

```

43 > arr( std::make_unique<std::unique_ptr<int[]>[]>(rows) ); //выделение памяти под 2d массив
44 for (int i = 0; i < rows; i++) //заполняем одномерный массив массивом
45 {
46     arr[i] = std::make_unique<int[]>(cols); //в ячейке массива массив
47 }
48
49 generate_elements_in_2d_array(arr, rows, cols); //заполняем массив
50 print_2d_array(arr, rows, cols); //печатаем 2d массив
51
52 add_line_to_2d_array(arr, &rows, cols); //добавляем строку в массив
53 print_2d_array(arr, rows, cols); //печатаем 2d массив
54 }
55
56 void generate_elements_in_2d_array(
57     const std::unique_ptr<std::unique_ptr<int[]>[]> &arr,
58     const int rows,
59     const int cols,
60     int a,
61     int b
62 )
63 {
64     for (int i = 0; i < rows; i++)
65     {
66         for (int j = 0; j < cols; j++)
67         {
68             arr[i][j] = rand() % (b - a + 1) + a; //рандом от a до b
69         }
70     }
71 }
72
73 template <typename T>
74 void print_2d_array(
75     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
76     const int rows,
77     const int cols
78 )
79 {
80     for (int i = 0; i < rows; i++)
81     {
82         for (int j = 0; j < cols; j++)
83         {
84             std::cout << arr[i][j] << "\t"; //печатаем элементы через табуляцию
85         }
86         std::cout << std::endl; //новая строка массива
87     }
88     std::cout << std::endl; //чтобы матрицы не слиплись, вызвав функцию два раза
89 }
90
91 template <typename T>
92 void add_line_to_2d_array(
93     std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
94     int *rows,
95     const int cols
96 )
97 {
98     *rows += 1; //увеличиваем количество строк
99     std::unique_ptr<
100         std::unique_ptr<T[]>[]
101     > temp(std::make_unique<std::unique_ptr<T[]>[]>(*rows)); //память под 2d массив
102     for (int i = 0; i < *rows; i++) //заполняем массив массивами

```

```

103 {
104     temp[i] = std::make_unique<T[]>(cols); //в ячейке массива массив
105 }
106
107 for (int i = 1; i < *rows; i++)
108 {
109     for (int j = 0; j < cols; j++)
110     {
111         temp[i][j] = arr[i - 1][j]; //переносим элементы из массива в новый массив
112     }
113 }
114
115 arr = move(temp); //передаём права
116 }

```

Листинг: Out

1	12	24	45	39	71	29	47	74	95	8	10	14
2	27	18	82	71	92	25	4	7	83	3	11	12
3	37	17	44	28	54	21	93	66	42	38	3	13
4	66	48	85	59	55	94	72	79	11	52	51	3
5	77	54	7	58	54	17	68	90	31	10	18	85
6												
7	0	0	0	0	0	0	0	0	0	0	0	0
8	12	24	45	39	71	29	47	74	95	8	10	14
9	27	18	82	71	92	25	4	7	83	3	11	12
10	37	17	44	28	54	21	93	66	42	38	3	13
11	66	48	85	59	55	94	72	79	11	52	51	3
12	77	54	7	58	54	17	68	90	31	10	18	85

Вариант V задание 3

Условие: Двумерный массив. Для матрицы размером NxM вывести на экран все седловые точки. Элемент матрицы называется седловой точкой, если он является наименьшим в своей строке и одновременно наибольшим в своем столбце или наоборот.

Решение:

Листинг: main.cpp

```

1 #include <iostream>
2 #include <memory>
3
4 //функция варианта 5 задания 3
5 void option5_task3();
6 //функция сгенерирует элементы для int массива
7 void generate_elements_in_2d_array(
8     const std::unique_ptr<std::unique_ptr<int[]>[]> &arr,
9     const int rows,
10    const int cols,
11    int a = 1,
12    int b = 10
13);
14 //функция напечатает функцию
15 template <typename T>
16 void print_2d_array(
17     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
18     const int rows,

```

```

19     const int cols
20 );
21 //печатает седловые точки
22 template <typename T>
23 void printf_saddle_points(
24     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
25     const int rows,
26     const int cols
27 );
28 //если value равен минимальному элементу по колонке, то функция возвращает true, иначе false
29 template <typename T>
30 bool is_min_in_col_2d_array(
31     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
32     const int rows,
33     const int col_number,
34     const T value
35 );
36 //если value равен максимальному элементу по колонке, то функция возвращает true, иначе false
37 template <typename T>
38 bool is_max_in_col_2d_array(
39     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
40     const int rows,
41     const int col_number,
42     const T value
43 );
44 //если value равен минимальному элементу по строке, то функция возвращает true, иначе false
45 template <typename T>
46 bool is_min_in_row_2d_array(
47     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
48     const int row_number,
49     const int cols,
50     const T value
51 );
52 //если value равен максимальному элементу по строке, то функция возвращает true, иначе false
53 template <typename T>
54 bool is_max_in_row_2d_array(
55     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
56     const int row_number,
57     const int cols,
58     const T value
59 );
60
61 int main()
62 {
63     srand(1); //Seed рандома. Программа всегда будет давать один и тот же результат при рандоме
64     option5_task3();
65
66     return 0;
67 }
68
69 void option5_task3()
70 {
71     int rows = 3; //количество строк
72     int cols = 7; //количество столбцов
73     std::unique_ptr<
74         std::unique_ptr<int[]>[]
75     > arr( std::make_unique<std::unique_ptr<int[]>[]>(rows) ); //выделение памяти под 2d массив
76     for (int i = 0; i < rows; i++) //заполняем одномерный массив массивом
77     {
78         arr[i] = std::make_unique<int[]>(cols); //в ячейке массива массив

```

```

79     }
80
81     generate_elements_in_2d_array(arr, rows, cols); //заполняем массив
82     print_2d_array(arr, rows, cols);               //печатаем 2d массив
83     printf_saddle_points(arr, rows, cols);
84 }
85
86 void generate_elements_in_2d_array(
87     const std::unique_ptr<std::unique_ptr<int[]>>> &arr,
88     const int rows,
89     const int cols,
90     int a,
91     int b
92 )
93 {
94     for (int i = 0; i < rows; i++)
95     {
96         for (int j = 0; j < cols; j++)
97         {
98             arr[i][j] = rand() % (b - a + 1) + a; //рандом от a до b
99         }
100     }
101 }
102
103 template <typename T>
104 void print_2d_array(
105     const std::unique_ptr<std::unique_ptr<T[]>>> &arr,
106     const int rows,
107     const int cols
108 )
109 {
110     for (int i = 0; i < rows; i++)
111     {
112         for (int j = 0; j < cols; j++)
113         {
114             std::cout << arr[i][j] << "\t";      //печатаем элементы через табуляцию
115         }
116         std::cout << std::endl;                    //новая строка массива
117     }
118     std::cout << std::endl; //чтобы матрицы не слиплись, вызвав функцию два раза
119 }
120
121 template <typename T>
122 void printf_saddle_points(
123     const std::unique_ptr<std::unique_ptr<T[]>>> &arr,
124     const int rows,
125     const int cols
126 )
127 {
128     for (int i = 0; i < rows; i++)
129     {
130         for (int j = 0; j < cols; j++)
131         {
132             if (
133                 is_min_in_col_2d_array(arr, rows, j, arr[i][j])
134                 &&
135                 is_max_in_row_2d_array(arr, i, cols, arr[i][j])
136                 ||
137                 is_max_in_col_2d_array(arr, rows, j, arr[i][j])
138                 &&

```



```

139         is_min_in_row_2d_array(arr, i, cols, arr[i][j])
140     )
141     {
142         std::cout << "arr[" << i << "][" << j << "] = " << arr[i][j] << std::endl;
143     }
144 }
145 }
146 }
147
148 template <typename T>
149 bool is_min_in_col_2d_array(
150     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
151     const int rows,
152     const int col_number,
153     const T value
154 )
155 {
156     T min = arr[0][col_number];
157     for (int i = 0; i < rows; i++) //поиск минимального в колонке
158     {
159         if(arr[i][col_number] < min)
160         {
161             min = arr[i][col_number];
162         }
163     }
164     return (value == min);
165 }
166
167 template <typename T>
168 bool is_max_in_col_2d_array(
169     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
170     const int rows,
171     const int col_number,
172     const T value
173 )
174 {
175     T max = arr[0][col_number];
176     for (int i = 0; i < rows; i++) //поиск максимального в колонке
177     {
178         if(arr[i][col_number] > max)
179         {
180             max = arr[i][col_number];
181         }
182     }
183     return (value == max);
184 }
185
186 template <typename T>
187 bool is_min_in_row_2d_array(
188     const std::unique_ptr<std::unique_ptr<T[]>[]> &arr,
189     const int row_number,
190     const int cols,
191     const T value
192 )
193 {
194     T min = arr[row_number][0];
195     for (int j = 0; j < cols; j++) //поиск минимального в строке
196     {
197         if (arr[row_number][j] < min)
198     {

```

```

199         min = arr[row_number][j];
200     }
201 }
202 return (value == min);
203 }
204
205 template <typename T>
206 bool is_max_in_row_2d_array(
207     const std::unique_ptr<std::unique_ptr<T[]>> &arr,
208     const int row_number,
209     const int cols,
210     const T value
211 )
212 {
213     T max = arr[row_number][0];
214     for (int j = 0; j < cols; j++)           //поиск максимального в строке
215     {
216         if (arr[row_number][j] > max)
217         {
218             max = arr[row_number][j];
219         }
220     }
221     return (value == max);
222 }

```

Листинг: Out

```

1  4      7      8      6      4      6      7
2  3      10     2      3      8      1      10
3  4      7      1      7      3      7      2
4
5  arr[0][0] = 4
6  arr[2][1] = 7

```

Изм	Лист	№ докум.	Подп.	Дата

ЛР.190333.ПО4.07 81 00

Лист

10