Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное

учреждение

высшего образования

«**Пермский национальный исследовательский**

**политехнический университет»**

Кафедра «Информационные технологии и автоматизированные

системы»

**ОТЧЁТ**

**по лабораторной работе №11.3**

Дисциплина: «Основы алгоритмизации и программирования»

Тема: стек

Вариант 17

Выполнил:

Студент группы ИВТ-20-2б

Сафронов Владислав Владиславович

Проверила:

Доцент кафедры ИТАС

Полякова О.А.

**Пермь, 2021**

**Цель задачи**

Получить практические навыки работы со стеками

**Постановка задачи**

Реализовать стек. Тип информационного поля int. Удалить из стека все элементы с четными информационными полями.

**Анализ задачи**

**Какие типы данных будут использованы:**

* integer

**Какие с этими данными выполняются действия:**

* integer – сравнение, математические операции.

**В каком виде эти данные будут представлены:**

**Реализация через структуры**

1. Структура Stack, реализующая стек. Имеет 2 поля: целочисленное поле data, в котором хранится значение элемента и указатель на предыдущий элемент стека.
2. struct Stack
3. {
4. int data;
5. Stack\* prev;
6. };
7. Функция make для создания стека. В качестве входного параметра функция принимает размер стека. Значения элементов стека генерируются с помощью ДСЧ. Функция возвращает указатель на верхний элемент стека.
8. Stack\* make(int n)
9. {
10. if (n == 0) return NULL;
11. Stack\* top, \* p;
12. top = NULL;
13. p = new Stack;
14. p->data = rand() % 100 + 1;
15. p->prev = NULL;
16. top = p;
17. for (int i = 1; i < n; i++)
18. {
19. Stack\* h = new Stack;
20. h->data = rand() % 100 + 1;
21. h->prev = top;
22. top = h;
23. }
24. return top;
25. }
26. Функция print\_stack для вывода элементов стека в консоль.
27. void print\_stack(Stack\* top)
28. {
29. if (top == NULL)
30. {
31. cout << "Стек пуст\n";
32. }
33. else
34. {
35. Stack\* p = top;
36. while (p != NULL)
37. {
38. cout << p->data << " ";
39. p = p->prev;
40. }
41. }
42. cout << endl;
43. }
44. Функция pop, которая удаляет верхний элемент из стека и возвращает его значение.
45. int pop(Stack\*& top)
46. {
47. Stack\* p = top;
48. int k = 0;
49. while (p != NULL)
50. {
51. k++;
52. p = p->prev;
53. }
54. p = top;
55. if (k == 1)
56. {
57. int x = top->data;
58. delete p;
59. top = NULL;
60. return x;
61. }
62. else
63. {
64. Stack\* t = p->prev;
65. int x = p->data;
66. top = t;
67. delete p;
68. return x;
69. }
70. }
71. Функция push для добавления элемента в стек.
72. Stack\* push(Stack\*& top, int value)
73. {
74. Stack\* p = new Stack;
75. p->data = value;
76. p->prev = top;
77. top = p;
78. return top;
79. }
80. Функция delete\_even, удаляющая из стека все элементы с четным информационным полем.
81. void delete\_even(Stack\*& top, int& size)
82. {
83. int k = 0;
84. Stack\* new\_stack = make(0);
85. for (int i = 0; i < size; i++)
86. {
87. int tmp = pop(top);
88. if (tmp % 2 != 0) push(new\_stack, tmp);
89. else k++;
90. }
91. size = size - k;
92. for (int i = 0; i < size; i++)
93. {
94. int tmp = pop(new\_stack);
95. push(top, tmp);
96. }
97. }
98. Функция clear\_stack для очистки памяти
99. void clear\_stack(Stack\*& top)
100. {
101. Stack\* p = top;
102. while (p != NULL)
103. {
104. p = top->prev;
105. delete top;
106. top = p;
107. }
108. }

**Реализация с помощью STL**

1. Функция print для вывода элементов стека в консоль.
2. void print(stack<int> st)
3. {
4. int n = st.size();
5. if (n == 0) cout << "Стек пуст";
6. else
7. {
8. for (int i = 0; i < n; i++)
9. {
10. cout << st.top() << " ";
11. st.pop();
12. }
13. }
14. cout << endl;
15. }
16. Функция delete\_even, удаляющая из стека все элементы с четным информационным полем.
17. void delete\_even(stack<int>& st, int& n) {
18. stack<int> st2;
19. for (int i = 0; i < n; i++) {
20. if (st.top() % 2 != 0) {
21. st2.push(st.top());
22. }
23. st.pop();
24. }
25. n = st2.size();
26. for (int i = 0; i < n; i++) {
27. st.push(st2.top());
28. st2.pop();
29. }
30. }

**Блок-схема программы**

**Код программы**

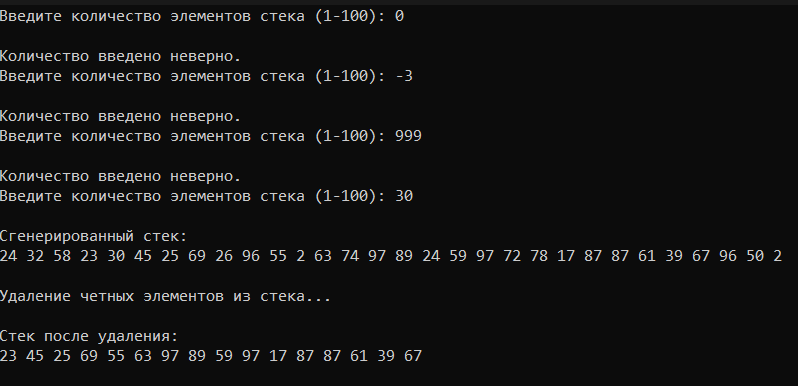
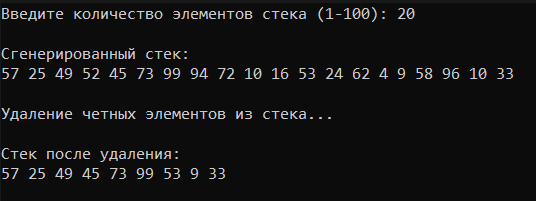
**Реализация через структуры**

1. #include <iostream>
2. #include <ctime>
3. using namespace std;
5. struct Stack
6. {
7. int data;
8. Stack\* prev;
9. };
11. Stack\* make(int n)
12. {
13. if (n == 0) return NULL;
14. Stack\* top, \* p;
15. top = NULL;
16. p = new Stack;
17. p->data = rand() % 100 + 1;
18. p->prev = NULL;
19. top = p;
20. for (int i = 1; i < n; i++)
21. {
22. Stack\* h = new Stack;
23. h->data = rand() % 100 + 1;
24. h->prev = top;
25. top = h;
26. }
27. return top;
28. }
30. void print\_stack(Stack\* top)
31. {
32. if (top == NULL)
33. {
34. cout << "Стек пуст\n";
35. }
36. else
37. {
38. Stack\* p = top;
39. while (p != NULL)
40. {
41. cout << p->data << " ";
42. p = p->prev;
43. }
44. }
45. cout << endl;
46. }
48. int pop(Stack\*& top)
49. {
50. Stack\* p = top;
51. int k = 0;
52. while (p != NULL)
53. {
54. k++;
55. p = p->prev;
56. }
57. p = top;
58. if (k == 1)
59. {
60. int x = top->data;
61. delete p;
62. top = NULL;
63. return x;
64. }
65. else
66. {
67. Stack\* t = p->prev;
68. int x = p->data;
69. top = t;
70. delete p;
71. return x;
72. }
73. }
75. Stack\* push(Stack\*& top, int value)
76. {
77. Stack\* p = new Stack;
78. p->data = value;
79. p->prev = top;
80. top = p;
81. return top;
82. }
84. void delete\_even(Stack\*& top, int& size)
85. {
86. int k = 0;
87. Stack\* new\_stack = make(0);
88. for (int i = 0; i < size; i++)
89. {
90. int tmp = pop(top);
91. if (tmp % 2 != 0) push(new\_stack, tmp);
92. else k++;
93. }
94. size = size - k;
95. for (int i = 0; i < size; i++)
96. {
97. int tmp = pop(new\_stack);
98. push(top, tmp);
99. }
100. }
102. void clear\_stack(Stack\*& top)
103. {
104. Stack\* p = top;
105. while (p != NULL)
106. {
107. p = top->prev;
108. delete top;
109. top = p;
110. }
111. }
113. int main()
114. {
115. setlocale(LC\_ALL, "Russian");
116. srand(time(NULL));
118. int n, k = 0;
119. cout << "Введите количество элементов стека (1-100): ";
120. cin >> n;
121. while (n < 1 || n > 99)
122. {
123. cout << "\nКоличество введено неверно.\n";
124. cout << "Введите количество элементов стека (1-100): ";
125. cin >> n;
126. }
127. Stack\* a = make(n);
128. cout << "\nСгенерированный стек: \n";
129. print\_stack(a);
130. cout << "\nУдаление четных элементов из стека...\n";
131. delete\_even(a, n);
132. cout << "\nСтек после удаления: \n";
133. print\_stack(a);
134. clear\_stack(a);
135. delete a;
136. }

**Реализация с помощью STL**

1. #include <iostream>
2. #include <ctime>
3. #include <stack>
4. using namespace std;
6. void print(stack<int> st)
7. {
8. int n = st.size();
9. if (n == 0) cout << "Стек пуст";
10. else
11. {
12. for (int i = 0; i < n; i++)
13. {
14. cout << st.top() << " ";
15. st.pop();
16. }
17. }
18. cout << endl;
19. }
21. void delete\_even(stack<int>& st, int& n)
22. {
23. stack<int> st2;
24. for (int i = 0; i < n; i++)
25. {
26. if (st.top() % 2 != 0)
27. {
28. st2.push(st.top());
29. }
30. st.pop();
31. }
32. n = st2.size();
33. for (int i = 0; i < n; i++)
34. {
35. st.push(st2.top());
36. st2.pop();
37. }
38. }
40. int main()
41. {
42. setlocale(LC\_ALL, "Russian");
43. srand(time(NULL));
44. stack<int> st;
45. int n;
46. cout << "Введите количество элементов стека (1-100): ";
47. cin >> n;
48. while (n < 1 || n > 99)
49. {
50. cout << "\nКоличество введено неверно.\n";
51. cout << "Введите количество элементов стека (1-100): ";
52. cin >> n;
53. }
54. for (int i = 0; i < n; i++)
55. {
56. st.push(rand() % 100 + 1);
57. }
58. cout << "\nСгенерированный стек: \n";
59. print(st);
60. delete\_even(st, n);
61. cout << "\nСтек после удаления: \n";
62. print(st);
63. }

**Скриншоты результатов работы программы**

****