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 №11.1**

Дисциплина: «основы алгоритмизации и программирования»

Тема: Динамические структуры данных

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 с однонаправленными списками;

**Постановка задачи**

Реализовать однонаправленный список. Тип информационного поля int. Удалить из списка все элементы с четными информационными полями.

**Анализ задачи**

**Какие типы данных будут использованы:**

* integer

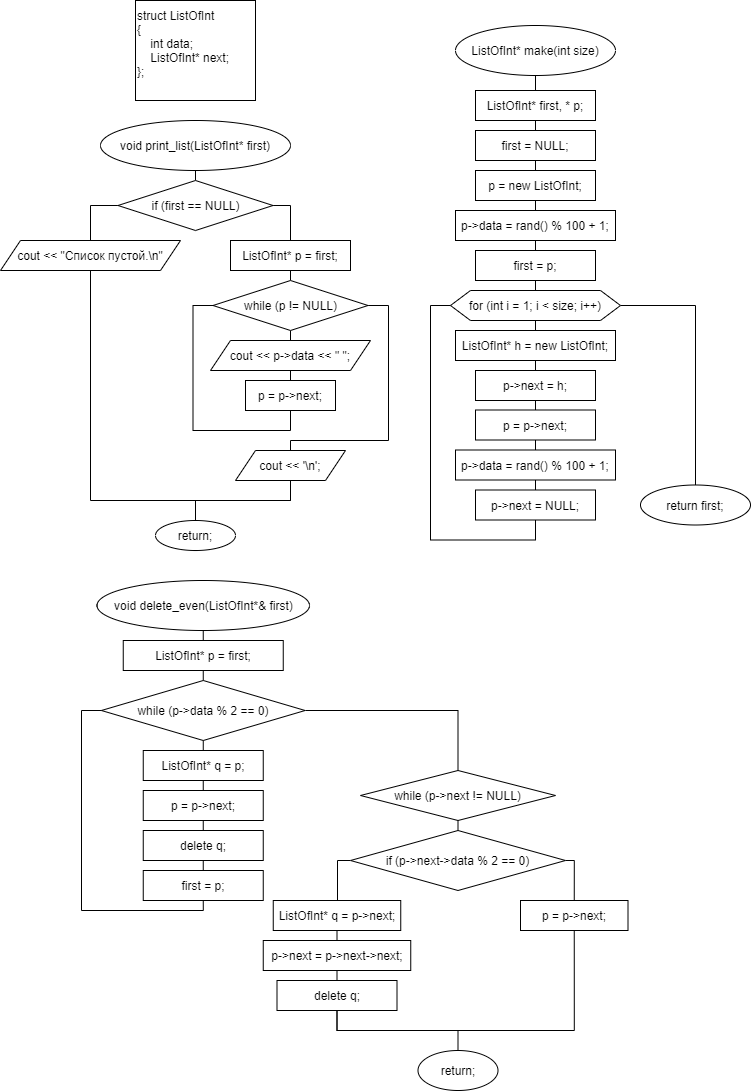
**Какие с этими данными выполняются действия:**

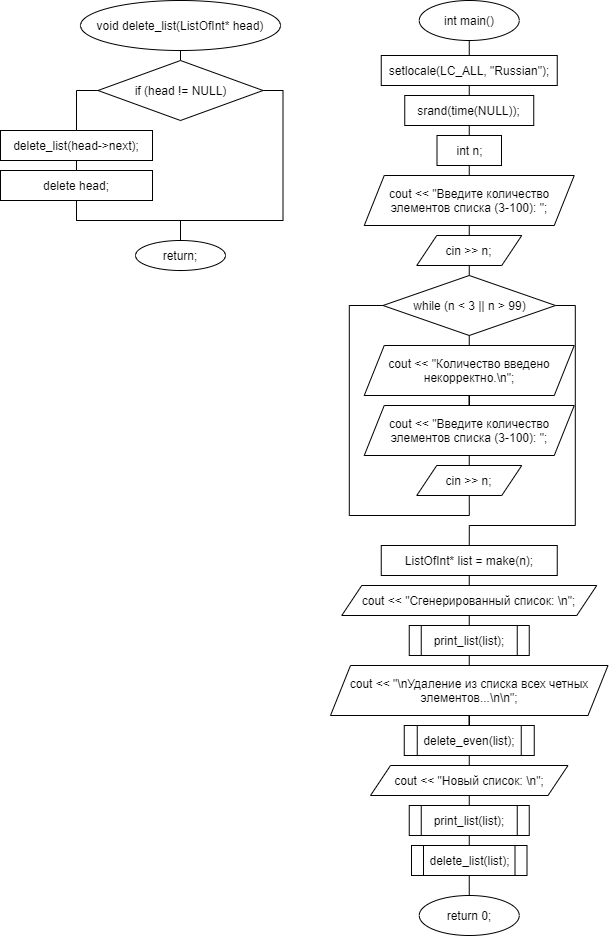
* integer – сравнение, математические операции.

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

1. Структура ListOfInt, реализующая однонаправленный список. Имеет 2 поля: целочисленное поле data, в котором хранится значение элемента и указатель на следующий элемент однонаправленного списка.
2. struct ListOfInt
3. {
4. int data;
5. ListOfInt\* next;
6. };
7. Функция print\_list для вывода однонаправленного списка в консоль.
8. void print\_list(ListOfInt\* first)
9. {
10. if (first == NULL) cout << "Список пустой.\n";
11. else
12. {
13. ListOfInt\* p = first;
14. while (p != NULL)
15. {
16. cout << p->data << " ";
17. p = p->next;
18. }
19. cout << '\n';
20. }
22. }
23. Функция make для создания однонаправленного списка. В качестве входного параметра функция принимает размер однонаправленного списка. Значения элементов списка генерируются с помощью ДСЧ. Функция возвращает указатель на первый элемент сгенерированного списка.
24. Функция delete\_even для удаления всех четных элементов из списка.
25. void delete\_even(ListOfInt\*& first)
26. {
27. ListOfInt\* p = first;
28. while (p->data % 2 == 0)
29. {
30. ListOfInt\* q = p;
31. p = p->next;
32. delete q;
33. first = p;
34. }
35. while (p->next != NULL)
36. {
37. if (p->next->data % 2 == 0)
38. {
39. ListOfInt\* q = p->next;
40. p->next = p->next->next;
41. delete q;
42. }
43. else p = p->next;
44. }
45. }
46. Функция delete\_list для удаления всех элементов однонаправленного списка. Функция вызывается рекурсивно.
47. void delete\_list(ListOfInt\* head)
48. {
49. if (head != NULL)
50. {
51. delete\_list(head->next);
52. delete head;
53. }
54. }

**Блок-схема программы**

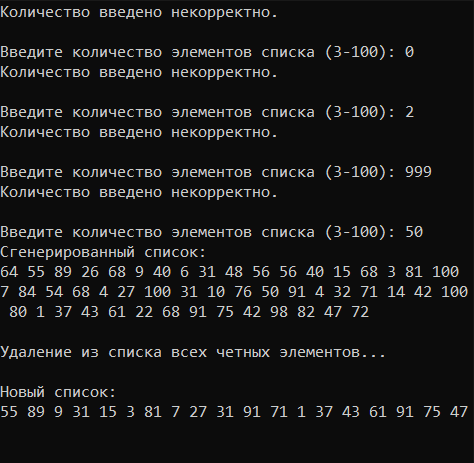
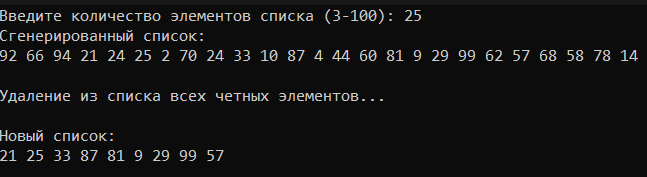




**Код программы**

1. #include <iostream>
2. #include <ctime>
3. using namespace std;
5. struct ListOfInt
6. {
7. int data;
8. ListOfInt\* next;
9. };
11. void print\_list(ListOfInt\* first)
12. {
13. if (first == NULL) cout << "Список пустой.\n";
14. else
15. {
16. ListOfInt\* p = first;
17. while (p != NULL)
18. {
19. cout << p->data << " ";
20. p = p->next;
21. }
22. cout << '\n';
23. }
25. }
27. ListOfInt\* make(int size)
28. {
29. ListOfInt\* first, \* p;
30. first = NULL;
31. p = new ListOfInt;
32. p->data = rand() % 100 + 1;
33. first = p;
34. for (int i = 1; i < size; i++)
35. {
36. ListOfInt\* h = new ListOfInt;
37. p->next = h;
38. p = p->next;
39. p->data = rand() % 100 + 1;
40. p->next = NULL;
41. }
42. return first;
43. }
45. void delete\_even(ListOfInt\*& first)
46. {
47. ListOfInt\* p = first;
48. while (p->data % 2 == 0)
49. {
50. ListOfInt\* q = p;
51. p = p->next;
52. delete q;
53. first = p;
54. }
55. while (p->next != NULL)
56. {
57. if (p->next->data % 2 == 0)
58. {
59. ListOfInt\* q = p->next;
60. p->next = p->next->next;
61. delete q;
62. }
63. else p = p->next;
64. }
65. }
67. void delete\_list(ListOfInt\* head)
68. {
69. if (head != NULL)
70. {
71. delete\_list(head->next);
72. delete head;
73. }
74. }
76. int main()
77. {
78. setlocale(LC\_ALL, "Russian");
79. srand(time(NULL));
81. int n;
83. cout << "Введите количество элементов списка (3-100): ";
84. cin >> n;
85. while (n < 3 || n > 99) {
86. cout << "Количество введено некорректно.\n";
87. cout << "\nВведите количество элементов списка (3-100): ";
88. cin >> n;
89. }
91. ListOfInt\* list = make(n);
92. cout << "Сгенерированный список: \n";
93. print\_list(list);
94. cout << "\nУдаление из списка всех четных элементов...\n\n";
95. delete\_even(list);
96. cout << "Новый список: \n";
97. print\_list(list);
99. delete\_list(list);
101. return 0;
102. }

**Скриншоты результатов работы программы**

****