ГУАП

КАФЕДРА № 43

ОТЧЕТ   
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Старший преподаватель |  |  |  | Рогачев С. А. |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

|  |
| --- |
| ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №2 |
| «Линейные и Циклические списки» |
| по курсу: Структуры и алгоритмы обработки данных |
|  |
|  |

РАБОТУ ВЫПОЛНИЛ

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| СТУДЕНТ ГР. № | М011 |  |  |  | Борисов С. И. |
|  |  |  | подпись, дата |  | инициалы, фамилия |

Санкт-Петербург 2021

**Цель работы**

Целью работы является изучение структур данных «линейный список» и «циклический список», а также получение практических навыков их реализации.

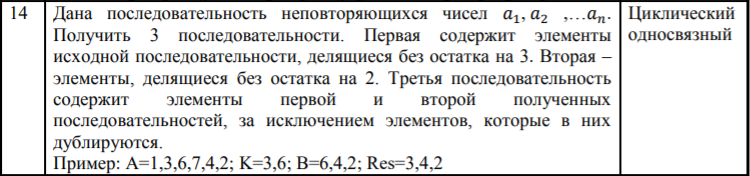
**Задание на лабораторную**

Реализовать структуры данных «линейный список» и «циклический список» в соответствии с заданным вариантом.

Дополнительно программа должна осуществлять следующие операции:

1. Добавление/удаление элемента в список (с клавиатуры);
2. Вывод исходного и результирующего списков на экран;
3. Если списки являются многочленами, в выводе должна быть отражена степень каждого элемента.

**Вариант задания:**



**Листинг программы, реализующей алгоритм**

// М011 ЛБ-2 Боривов С. И. Вариант 14.

#include <iostream>

#include<cmath>

using namespace std;

struct Node

{

int val;

Node\* next;

Node(int x, Node\* t)

{

val = x;

next = t;

}

};

struct cycle\_sp

{

Node\* first = new Node(NULL, NULL);

Node\* now = new Node(NULL, first);

bool is\_empty()

{

if (first->next == NULL)

return 0;

else

return 1;

}

void input(int value)

{

Node\* new\_elem = new Node(value, first);

if (!is\_empty())

{

first = new\_elem;

now = new\_elem;

now->next = first;

}

else

{

now->next = new\_elem;

now = new\_elem;

}

}

void erase\_first() // удаление эл в начале

{

Node\* erased = first;

first = first->next;

now->next = first;

delete erased;

}

void erase\_elem(int elem)

{

if (is\_empty() == 0)

return;

else

if (first->val == elem)

{

erase\_first();

return;

}

else

{

if (now->val == elem) // удаление эл в конце

{

if (now == first)

{

erase\_first();

return;

}

Node\* erased = first;

while (erased->next != now)

erased = erased->next;

erased->next = first;

delete now;

now = erased;

}

else

{

Node\* one = first; // удаление элемента в середине

Node\* second = first->next;

while (second->next != first && second->val != elem)

{

second = second->next;

one = one->next;

}

if (second->next == first && second->val != elem)

{

cout << "данного элемента нет в списке.";

return;

}

one->next = second->next;

delete second;

}

}

}

void print\_sp()

{

if (is\_empty() == 0)

{

return;

}

else

{

Node\* tmp = first;

while (tmp->next != first)

{

cout << tmp->val << ' ';

tmp = tmp->next;

}

cout << tmp->val << ' ';

tmp = tmp->next;

cout << endl << endl;

}

}

void zadanie\_1(cycle\_sp sp)

{

Node\* tmp = first;

if (tmp->val % 3 == 0)

sp.input(tmp->val);

tmp = tmp->next;

while (tmp != first)

{

if (tmp->val % 3 == 0)

sp.input(tmp->val);

tmp = tmp->next;

}

cout << "первый список: ";

sp.print\_sp();

cout << endl;

}

void zadanie\_2(cycle\_sp sp)

{

Node\* tmp = first;

if (tmp->val % 2 == 0)

sp.input(tmp->val);

tmp = tmp->next;

while (tmp != first)

{

if (tmp->val % 2 == 0)

sp.input(tmp->val);

tmp = tmp->next;

}

cout << "второй список: ";

sp.print\_sp();

cout << endl;

}

void zadanie\_3(cycle\_sp sp)

{

Node\* tmp = first;

if (((tmp->val % 3 == 0) || (tmp->val % 2 == 0)) && !((tmp->val % 2 == 0) && (tmp->val % 3 == 0)))

sp.input(tmp->val);

tmp = tmp->next;

while (tmp != first)

{

if (((tmp->val % 3 == 0) || (tmp->val % 2 == 0)) && !((tmp->val % 2 == 0) && (tmp->val % 3 == 0)))

sp.input(tmp->val);

tmp = tmp->next;

}

cout << "третий список: ";

sp.print\_sp();

cout << endl;

}

};

int main()

{

setlocale(LC\_ALL, "Russian");

int n, elem;

cout << "Введите количество элементов исходной последовательности: ";

cin >> n;

if (n < 2)

{

cout << "Число элементов списка слишком мало." << endl;

system("pause");

return 0;

}

cout << endl << endl;

cycle\_sp list;

for (int i = 0; i < n; i++)

{

cin >> elem;

list.input(elem);

}

cout << "изначальный список: ";

list.print\_sp();

cycle\_sp list\_1;

cycle\_sp list\_2;

cycle\_sp list\_3;

list.zadanie\_1(list\_1);

list.zadanie\_2(list\_2);

list.zadanie\_3(list\_3);

cout << endl << endl << "введите удаляемое число: ";

cin >> elem;

list.erase\_elem(elem);

cout << "результат после удаления элемента из 1-го списка: ";

list.print\_sp();

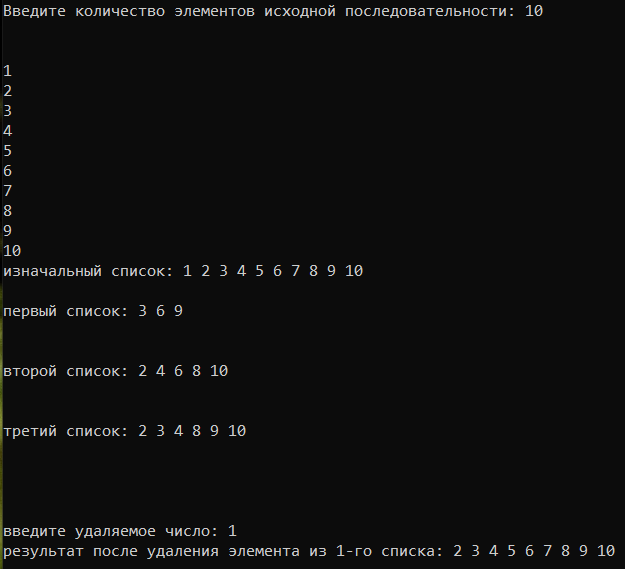
system("pause");

return 0;

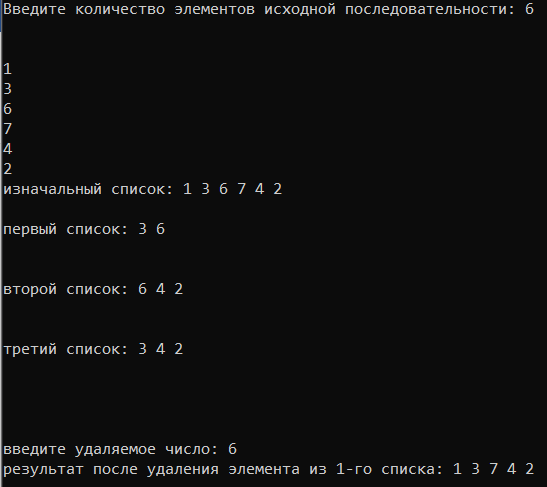
}

**Примеры**

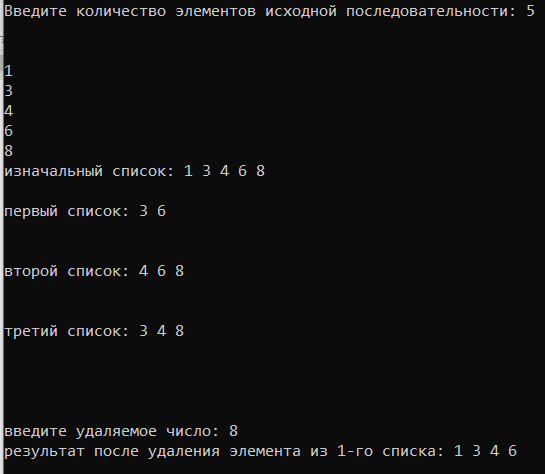
**Удаление в начале:**



**Удаление в середине:**



**Удаление в конце:**



**Выводы**

На основе этих расчетов можно сделать вывод, что был разработан алгоритм, характеристики которого соответствуют поставленному заданию.