|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №2  по дисциплине «Языки программирования»  Вариант 1 | | |
|  | | Работу выполнил  студент группы ФИТ-1,2-2023 1 курса  Лисуков Д.В  «22» Июня 2024 г. |
| Работу проверил  Иммис М.В  «22» Июня 2024 г. |
| Пермь 2024 | | |

Содержание

# **1.Постановка задачи**

«Хитрый купец» Два купца отправились торговать за море. Каждый из них повез по N одинаковых тюков с товаром. В пути корабль попал в шторм и дал течь. Чтобы корабль не затонул, капитан приказал поднять весь груз на палубу корабля, расставить его вдоль бортов по периметру и выбросить за борт половину груза. Груз выбрасывается по следующему правилу:

• каждый тюк получает порядковый номер, начиная с тюка, стоящего на носу корабля;

• их нумерация осуществляется по часовой стрелке;

• номер первого выброшенного тюка соответствует текущему числу месяца M;

• через K тюков следующий тюк с товаром выбрасывается. Один из купцов прознал про это правило. Как необходимо расставить ему груз, чтобы ни один из его тюков не был выброшен? Использовать циклический однонаправленный список.

Данная задача подразумевает использование динамических структур данных, необходимо реализовать входные данные, обработку и удаление данных.

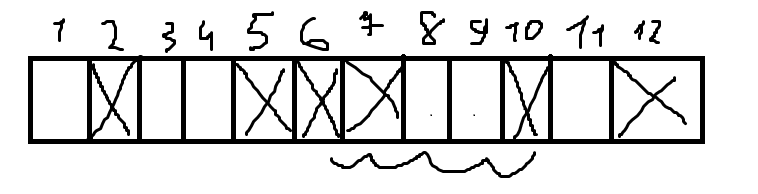
# **2. Алгоритм решения**

Задачу можно облегчить, до следующих пунктов:

1. 3 входных данных типа int. N – количество грузов. M – номер месяца. K – периодичность.
2. Есть список, который хранит элементы от 1 до N, все грузы
3. Необходимо удалить половину грузов следующим образом: Первым удаляется груз с номером месяца(M), после чего, начиная с удаленного груза, необходимо удалять следующий груз, находящийся после периодичности K по часовой стрелке.
4. Так как список циклический, то пройдя вся грузы, мы вновь пройдемся до следующего элемента, после периодичности K.
5. Организовать вывод данных и проверку на дурака

Необходимо создать структуру односвязного списка, далее заполнить ее элементами, после чего производить обработку.

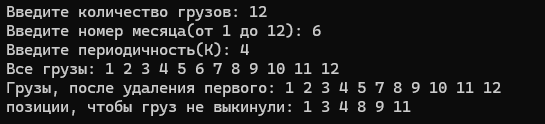
Так, например, при входных данных N = 12, M = 6, K= 4 первым делом мы удалим элемент номер 6, затем, начиная с следующего элемента, будем отсчитывать 4 и удалять данный элемент.



Таким образом, уцелевшие грузы будут с номерами 1, 3, 4, 8, 9, 11

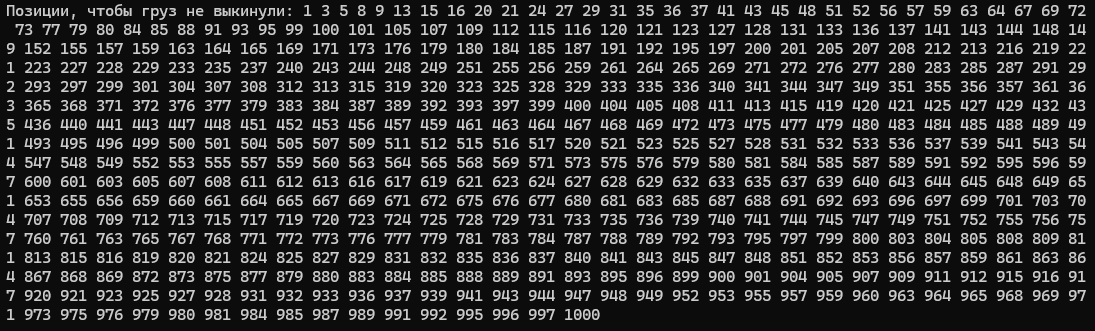
# **3.Тестирование программы**

Тест 1:   
Входные данные, которые были доказаны N=12, M=6, K=4:



Тест 2:

Большие входные данные, N = 1000, M = 7, K = 4



Тест 3:

Неправильные входные данные:



Тест 4:

Неправильный номер месяца:  


# **4. Код программы**

#include <iostream>

#include <string>

using namespace std;

// Структура для элементов односвязного списка

struct Node {

int data;

Node\* next;

//для более простого создания узла

Node(int val) : data(val), next(nullptr) {}

};

// поиск элемента в списке

Node\* searchElement(Node\* head, int target) {

Node\* current = head; //создаем еще один указатель на начало

while (current != nullptr) { // проходимся по всему списку

if (current->data == target) {

return current; // возвращаем узел, если найден элемент

}

current = current->next; // двигаем в цикле

}

return nullptr; // если элемент не был найден

}

void deleteNode(Node\*& head, int value) { // удаение элементов

if (head == nullptr) {

return; // если пусто, ретурн

}

Node\* current = head; // указатель с начала

Node\* previous = nullptr; // с конца

do {

if (current->data == value) { //если найденный элемент соответсвует

if (current == head) { // если это первый

if (head->next == head) { // и второй равен указателю первого

delete head; // то удаляем первый

head = nullptr;

}

else { // если следующий не имеет указатель первого

Node\* temp = head; // запоминаем указатель на голову

while (temp->next != head) {

temp = temp->next; //двигаемся до последнего

}

temp->next = head->next; // последнему меняем указатель на второй элемент

head = head->next; // второй становится первым

delete current; // первый удаляется

}

}

else { // если не первый

previous->next = current->next; // то стандартное удаление

delete current;

}

return;

}

previous = current; // двигаем указатель в цикле

current = current->next;

} while (current != head); // пока не пройдем по кругу

cout << "Element not found" << endl;

}

int main() {

setlocale(LC\_ALL, "Russian");

string input;

cout << "Введите количество грузов: ";

int n; // кол-во грузов

cin >> input;

try { // проверка на входные данные

size\_t pos;

n = stoi(input, &pos); // преобразуем строку в число

if (pos < input.size()) {

throw invalid\_argument("Ошибка: Введенная строка не является целым числом.");

return 0;

}

} catch (const invalid\_argument& e) { // встроенное исключение в c++

cerr << "Не может быть словом / символом" << endl; // если это не число, выбрасываем исключение

return 1;

} catch (const out\_of\_range& e) {

cerr << "Ошибка: Число находится вне диапазона типа int." << endl;

}

cout << "Введите номер месяца(от 1 до 12): ";

int m; // номер месяца

cin >> input;

try { // проверка на входные данные

size\_t pos;

m = stoi(input, &pos); // преобразуем строку в число

if (pos < input.size()) {

throw invalid\_argument("Ошибка: Введенная строка не является целым числом.");

return 0;

}

}

catch (const invalid\_argument& e) { // встроенное исключение в c++

cerr << "Не может быть словом / символом" << endl; // если это не число, выбрасываем исключение

return 1;

}

catch (const out\_of\_range& e) { // если число больше инта

cerr << "Ошибка: Число находится вне диапазона типа int." << endl;

}

if (m > 12 || m > n) { // если m не коректный

cout << "Месяц не может быть больше 12 или количества грузов";

return 0;

}

cout << "Введите периодичность(K): ";

int k; // переодичность

cin >> input;

try { // проверка на входные данные

size\_t pos;

k = stoi(input, &pos); // преобразуем строку в число

if (pos < input.size()) {

throw invalid\_argument("Ошибка: Введенная строка не является целым числом.");

return 0;

}

}

catch (const invalid\_argument& e) { // встроенное исключение в c++

cerr << "Не может быть словом / символом" << endl; // если это не число, выбрасываем исключение

return 1;

}

catch (const out\_of\_range& e) {

cerr << "Ошибка: Число находится вне диапазона типа int." << endl;

}

Node\* head = new Node(1); // создаем узел, начиная нумеровать

Node\* current = head; // указатель на голову

for (int i = 2; i <= n; i++) {

current->next = new Node(i); // добавляем все грузы в список

current = current->next;

}

current->next = head; // циклим список

current = head; // переходим в начало

cout << "Все грузы: ";

while (current->next != head) { // вывод изначального спика

cout << current->data << " ";

current = current->next;

}

cout << current->data << endl;

// находим элемент месяца

Node\* result = searchElement(head, m); // находим элемент месяца

Node\* tea = result->next; // чтобы проще удалять дальше, элемент на следующий запоминаем

deleteNode(head, m); // удаляем найденный элемент

cout << "Грузы, после удаления первого: ";

current = head;

while (current->next != head) { //вывод элементов после удаления номера месяца

cout << current->data << " ";

current = current->next;

}

cout << current->data << endl;

Node\* temp = tea; //еще один вспомогательный указатель

for (int i = 0; i < (n - 1) / 2; i++) { // цикл с половиной элементов

temp = tea;

int count = 1; // ставим счетчик = 1

while (count != k) { // проходим, пока не пройдет переодичность K

temp = temp->next;

count++;

}

tea = temp->next; // запоминаем указатель на следующий

deleteNode(head, temp->data); // удаляем элемент, который прошел после переодичности K

}

cout << "Позиции, чтобы груз не выкинули: ";

current = head; // конечное удаление элемента

while (current->next != head) {

cout << current->data << " ";

current = current->next;

}

cout << current->data << endl;

return 0;

}