Министерство высшего образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

**«Пермский национальный исследовательский политехнический университет» (ПНИПУ)**

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

Отчет по лабораторной работе №11

Выполнил

Студент группы РИС-22-2б

Петров Д.М.

Проверил Доцент кафедры ИТАС

Полякова О. А.

Пермь 2023

**Постановка задачи:**

1. Сформировать однонаправленный и двунаправленный списки или стек и очередь. Тип информационного поля указан в варианте.

2. Распечатать полученную структуру.

3. Выполнить обработку структуры в соответствии с заданием.

4. Распечатать полученный результат.

5. Удалить соответствующую структуру из памяти.

**Анализ задачи 11.0:**

1. Создаем структуру Beast, в которой будет инфополе типа char и указатель типа структуры на следующий элемент и на предыдущий.
2. Далее создаем сам список через отдельную функцию, передав ей заранее количество элементов списка. Создаем указатели на начало списка и два вспомогательных. Создаем первый узел, заносим значение, очищаем его адресное поле, куда потом будет записан адрес следующего узла. Ставим указатель p на последний(первый) элемент и входим в цикл for. Создаем новый элемент, очищаем его адресное поле, а в адресное поле первого узла вносим адрес этого узла. Список создан.
3. Добавлять элементы следует в отдельной функции. Обозначаем номер первого добавленного элемента. Входим в цикл while, аналогичный первому. Ставим указатель на начало списка, создаем новый элемент. доходим до предшествующего элемента нужному элементу через цикл for. И если нужный элемент существует, тогда - Ставим на него указатель, доходим до следующего элемента после нужного и ставим другой указатель. Связываем их и искомый элемент добавлен. В конце увеличиваем k на 2.
4. Печатать будем в отдельной функции. В цикле while, пока у нас адрес следующего элемента не будет равен нулю будем выводить элементы узлов, переходя к следующим.

**Анализ задачи 11.1:**

1. Через ф-ию main вызываем функции, с помощью цикла n раз обращаемся к функции push, т.к. стек не создан – вершинка=0. Ф-ия push отвечает за добавление элементов в стек, в качестве аргументов – двойной указатель на вершинку(чтоб она менялась во всем коде, а не только в данной ф-ии) и значение добавляемого элемента. Создали структурный указатель, залили в инфополе значение, проверили, если стек пуст, то данный элемент будет первой вершинкой, если стек не пуст, вершинку переобозначаем с предыдущего элемента на настоящий. Для удаления элемента используем ф-ию pop, инициализировали указатели на вершинку и на предыдущий элементы, идем по всему стеку, проверяя условие, как только сошлось выполняем удаление и очищение, переназначение вершинки. Для печати использована ф-ия print и цикл while, который будет работать, пока стек не пуст, смотрим на верхний элемент, удаляем и смотрим следующий.
2. Во втором случае проще, создаем исходный и искомый стеки, используем библиотеку <stack> для использования встроенных функций. Заполняем первый стек через цикл и ф-ию push, в аналогичном цикле ищем нужный элемент, как только нашли, удалили и вышли из цикла. Если искомый элемент не в вершинке, то предыдущие заносим в новый стек и удаляем из старого. Вывод аналогичен первому способому.

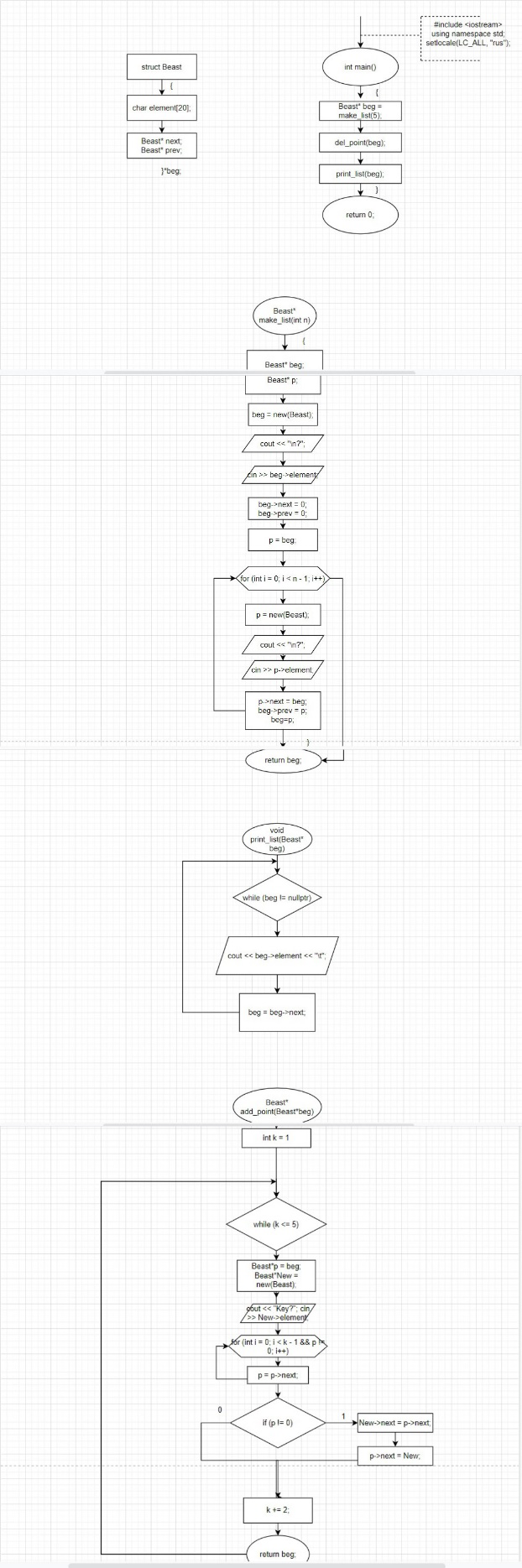
**Анализ задачи 11.2:**

1. Везде использован шаблон функции для свободного назначения типа поля структуры. Две структуры – первая(аналогична однонаправленному списку) и вторая(аналогична двунаправленному списку). Каждое действие – новая функция. В функции new\_queue создаем элементы и переходим к функции push, которая как раз и создает поле и работает с указателями. Вывод очереди аналогична выводу однонаправленного списка. Добавление элемента в очередь аналогично созданию очереди.
2. Для работы с готовыми командами я реализовал задачу через приоритетную очередь. Подключил библиотеку «queue», создал очередь в цикле for, перед этим задав количество членов очереди. Проверил, не пуста ли очередь. Добавлял элементы аналогично созданию очереди, в таком же цикле, только элементов было на n/2-1 меньше. Приоритетная очередь упорядочила их в порядке убывания. Вывод осуществил через доступ к верхнему(первому) элементу и его удаление в цикле while, пока очередь не пуста.

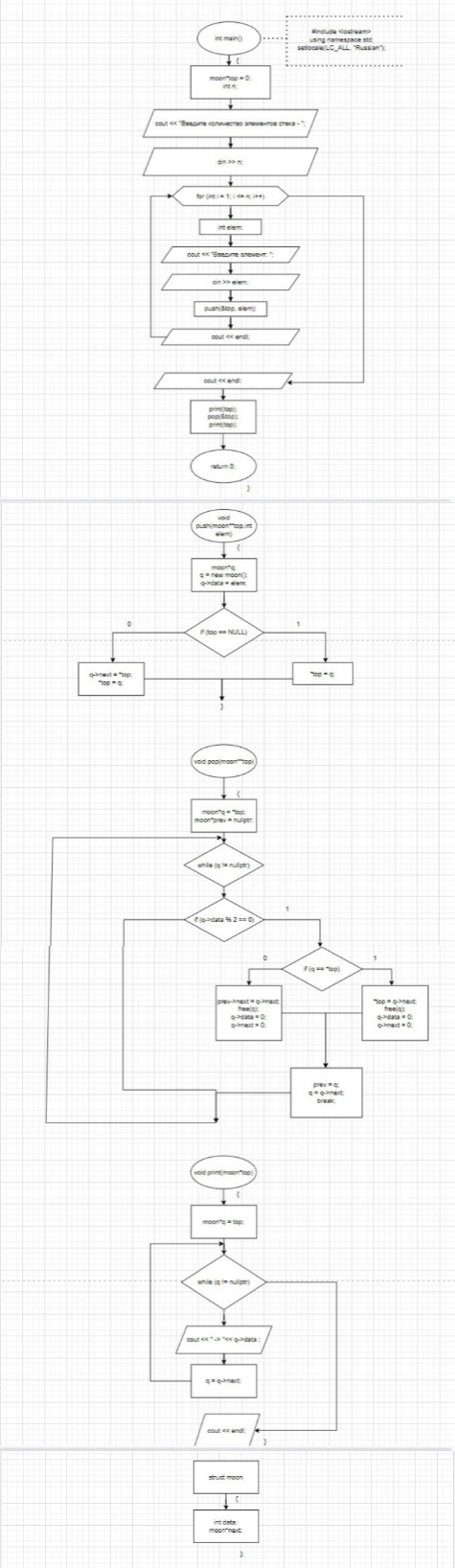
**Анализ задачи 11.3:**

1. Создаем структуру Beast, в которой будет инфополе типа int и указатель типа структуры на следующий элемент.
2. Далее создаем сам список через отдельную функцию, передав ей заранее количество элементов списка. Создаем указатели на начало списка и два вспомогательных. Создаем первый узел, заносим значение, очищаем его адресное поле, куда потом будет записан адрес следующего узла. Ставим указатель p на последний(первый) элемент и входим в цикл for. Создаем новый элемент, очищаем его адресное поле, а в адресное поле первого узла вносим адрес этого узла. Список создан.
3. Удалять элемент следует в отдельной функции. Ставим указатель на начало списка, инициализируем счетчик. Входим в цикл while, аналогичный предыдущему while-у, с каждым проходом увеличиваем счетчик на 1. Как только будет найден первый четный элемент, то у нас будет записан номер этого элемента. Вновь ставим указатель на начало, доходим до предшествующего элемента нужному элементу. Ставим на него указатель, доходим до следующего элемента после нужного и ставим другой указатель. Связываем их и искомый элемент удален.
4. Печатать будем в отдельной функции. В цикле while, пока у нас адрес следующего элемента не будет равен нулю будем выводить элементы узлов, переходя к следующим.

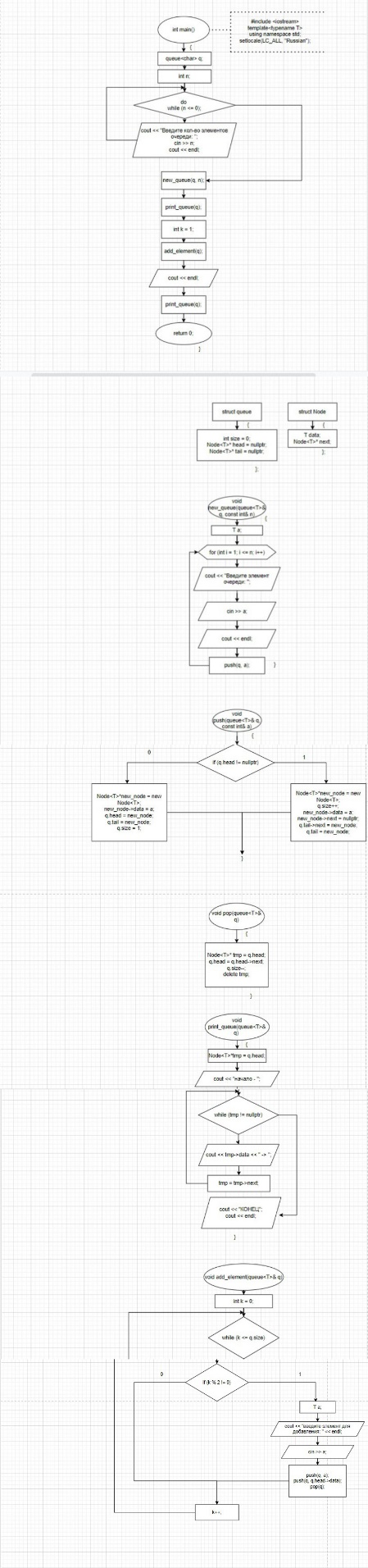
**Блок-схема 11.0:**

****

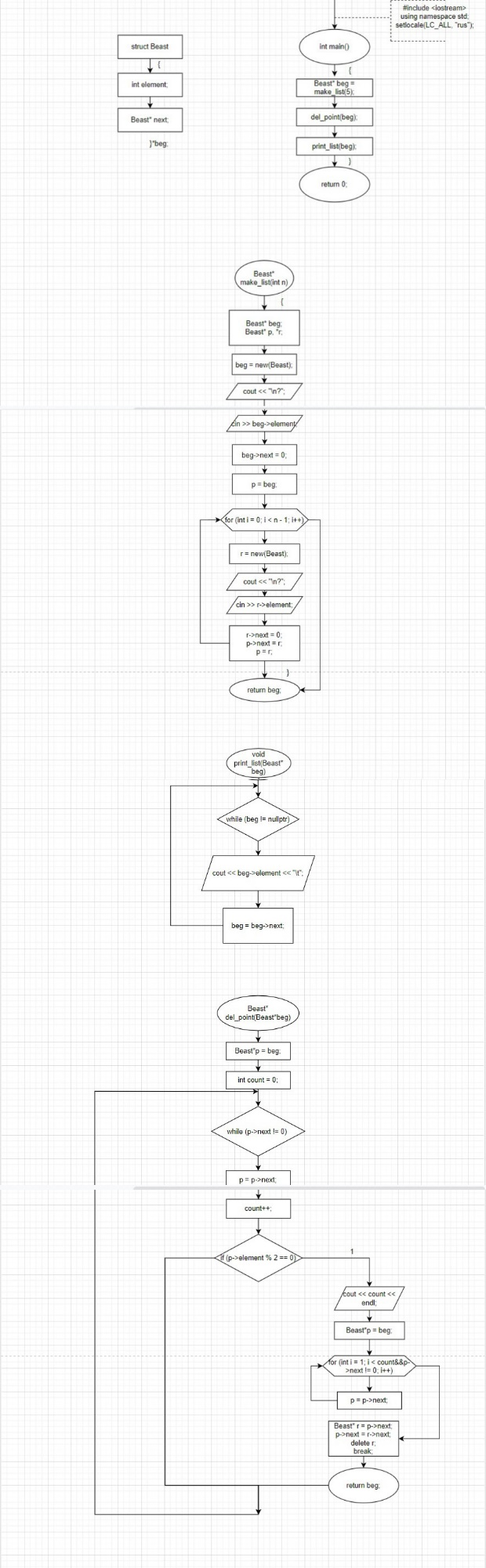
**Блок-схема 11.1:**

****

**Блок-схема 11.2:**

****

**Блок-схема 11.3:**

****

**Код программы 11.0:**

***#include<iostream>***

***using namespace std;***

***struct Beast***

***{***

***char element[20];***

***Beast\* next;***

***Beast\* prev;***

***}\*beg;***

***Beast\* make\_list(int n)***

***{***

***Beast\* beg;***

***Beast\* p;***

***beg = new(Beast);***

***cout << "\n?";***

***cin >> beg->element;***

***beg->prev = 0;***

***beg->next = 0;***

***p = beg;***

***for (int i = 0; i < n - 1; i++)***

***{***

***p = new(Beast);***

***cout << "\n?";***

***cin >> p->element;***

***p->next = beg;***

***beg->prev = p;***

***beg=p;***

***}***

***return beg;***

***}***

***void print\_list(Beast\* beg)***

***{***

***while (beg != nullptr)***

***{***

***cout << beg->element << "\t";***

***beg = beg->next;***

***}***

***}***

***Beast\* add\_point(Beast\*beg)***

***{***

***int k = 1;***

***while (k <= 5)***

***{***

***Beast\*p = beg;***

***Beast\*New = new(Beast);***

***cout << "key?"; cin >> New->element;***

***for (int i = 0; i < k - 1 && p != 0; i++)***

***p = p->next;***

***if (p != 0)***

***{***

***New->next = p->next;***

***p->next = New;***

***}***

***k += 2;***

***}***

***return beg;***

***}***

***int main()***

***{***

***Beast\* beg = make\_list(5);***

***add\_point(beg);***

***print\_list(beg);***

***return 0;***

***}***

**Код программы 11.1:**

***#include<iostream>***

***using namespace std;***

***struct moon***

***{***

***int data;***

***moon\*next;***

***};***

***void push(moon\*\*top,int elem)***

***{***

***moon\*q;***

***q = new moon();***

***q->data = elem;***

***if (top == NULL) { \*top = q; }***

***else***

***{***

***q->next = \*top;***

***\*top = q;***

***}***

***}***

***void pop(moon\*\*top)***

***{***

***moon\*q = \*top;***

***moon\*prev = nullptr;***

***while (q != nullptr)***

***{***

***if (q->data % 2 == 0)***

***{***

***if (q == \*top)***

***{***

***\*top = q->next;***

***free(q);***

***q->data = 0;***

***q->next = 0;***

***}***

***else***

***{***

***prev->next = q->next;***

***free(q);***

***q->data = 0;***

***q->next = 0;***

***}***

***prev = q;***

***q = q->next;***

***break;***

***}***

***}***

***}***

***void print(moon\*top)***

***{***

***moon\*q = top;***

***while (q != nullptr)***

***{***

***cout << " -> "<< q->data ;***

***q = q->next;***

***}***

***cout << endl;***

***}***

***int main()***

***{***

***moon\*top = 0;***

***int n;***

***cout << "Введите количество элементов стека - ";***

***cin >> n;***

***for (int i = 1; i <= n; i++)***

***{***

***int elem;***

***cout << "Введите элемент: ";***

***cin >> elem;***

***push(&top, elem);***

***cout << endl;***

***}***

***cout << endl;***

***print(top);***

***pop(&top);***

***print(top);***

***return 0;***

***}***

**Код программы 11.2:**

***#include<iostream>***

***using namespace std;***

***template<typename T>***

***struct Node***

***{***

***T data;***

***Node<T>\* next;***

***};***

***template<typename T>***

***struct queue***

***{***

***int size = 0;***

***Node<T>\* head = nullptr;***

***Node<T>\* tail = nullptr;***

***};***

***template<typename T>***

***void new\_queue(queue<T>& q, const int& n)***

***{***

***T a;***

***for (int i = 1; i <= n; i++)***

***{***

***cout << "Введите элемент очереди: ";***

***cin >> a;***

***cout << endl;***

***push(q, a);***

***}***

***}***

***template<typename T>***

***void push(queue<T>& q, const int& a)***

***{***

***if (q.head != nullptr)***

***{***

***Node<T>\*new\_node = new Node<T>;***

***q.size++;***

***new\_node->data = a;***

***new\_node->next = nullptr;***

***q.tail->next = new\_node;***

***q.tail = new\_node;***

***}***

***else***

***{***

***Node<T>\*new\_node = new Node<T>;***

***new\_node->data = a;***

***q.head = new\_node;***

***q.tail = new\_node;***

***q.size = 1;***

***}***

***}***

***template<typename T>***

***void pop(queue<T>& q)***

***{***

***Node<T>\* tmp = q.head;***

***q.head = q.head->next;***

***q.size--;***

***delete tmp;***

***}***

***template<typename T>***

***void print\_queue(queue<T>& q)***

***{***

***Node<T>\*tmp = q.head;***

***cout << "Start - ";***

***while (tmp != nullptr)***

***{***

***cout << tmp->data << " -> ";***

***tmp = tmp->next;***

***}***

***cout << "КОНЕЦ";***

***cout << endl;***

***}***

***template<typename T>***

***void add\_element(queue<T>& q)***

***{***

***int k = 0;***

***while (k <= q.size)***

***{***

***if (k % 2 != 0)***

***{***

***T a;***

***cout << "введите элемент для добавления: " << endl;***

***cin >> a;***

***push(q, a);***

***push(q, q.head->data);***

***pop(q);***

***}***

***k++;***

***}***

***}***

***int main()***

***{***

***queue<char> q;***

***int n;***

***do***

***{***

***cout << "Введите кол-во элементов очереди: ";***

***cin >> n;***

***cout << endl;***

***} while (n <= 0);***

***new\_queue(q, n);***

***print\_queue(q);***

***int k = 1;***

***add\_element(q);***

***cout << endl;***

***print\_queue(q);***

***return 0;***

***}***

**Код программы 11.3:**

***#include<iostream>***

***using namespace std;***

***struct Beast***

***{***

***int element;***

***Beast\* next;***

***}\*beg;***

***Beast\* make\_list(int n)***

***{***

***Beast\* beg;***

***Beast\* p, \*r;***

***beg = new(Beast);***

***cout << "\n?";***

***cin >> beg->element;***

***beg->next = 0;***

***p = beg;***

***for (int i = 0; i < n - 1; i++)***

***{***

***r = new(Beast);***

***cout << "\n?";***

***cin >> r->element;***

***r->next = 0;***

***p->next = r;***

***p = r;***

***}***

***return beg;***

***}***

***void print\_list(Beast\* beg)***

***{***

***while (beg != nullptr)***

***{***

***cout << beg->element << "\t";***

***beg = beg->next;***

***}***

***}***

***Beast\* del\_point(Beast\*beg)***

***{***

***Beast\*p = beg;***

***int count = 0;***

***while (p->next != 0)***

***{***

***p = p->next;***

***count++;***

***if (p->element % 2 == 0)***

***{***

***cout << count << endl;***

***Beast\*p = beg;***

***for (int i = 1; i < count&&p->next != 0; i++)***

***{***

***p = p->next;***

***}***

***Beast\* r = p->next;***

***p->next = r->next;***

***delete r;***

***return beg;***

***break;***

***}***

***}***

***return 0;***

***}***

***int main()***

***{***

***Beast\* beg = make\_list(5);***

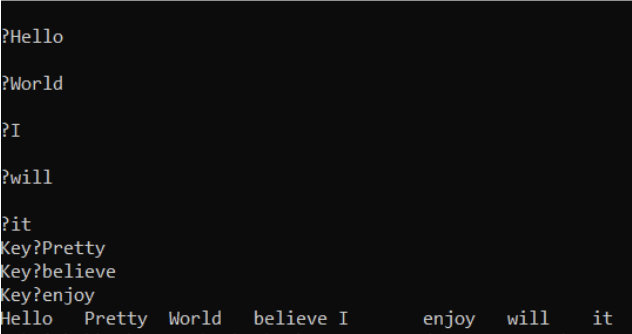
***del\_point(beg);***

***print\_list(beg);***

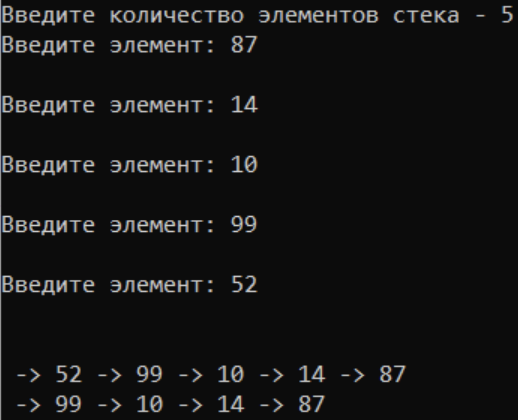
***return 0;***

***}***

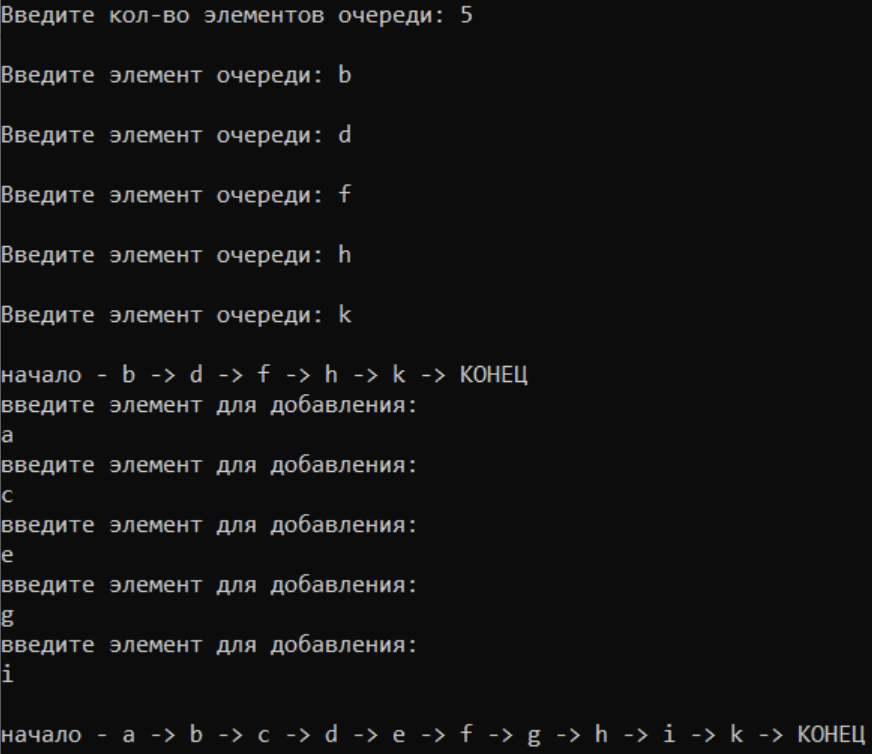
**Результат работы программы 11.0:**

****

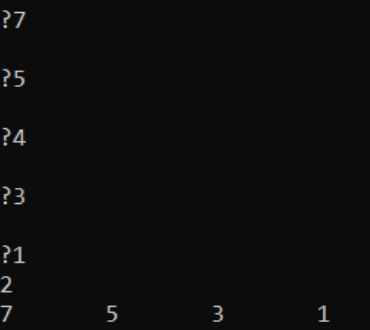
**Результат работы программы 11.1:**

****

**Результат работы программы 11.2:**

****

**Результат работы программы 11.3:**

****