**МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ**

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ

УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

**«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**

(**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**)» (**МАИ)**

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра 304 Вычислительные машины, системы и сети

СТРУКТУРЫ И АЛГОРИТМЫ ОБРАБОТКИ ДАННЫХ

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ **№ 4**

**“Списки”**

Группа: *М30-209Б-19*

Выполнил:

*Кузнецов И.И*

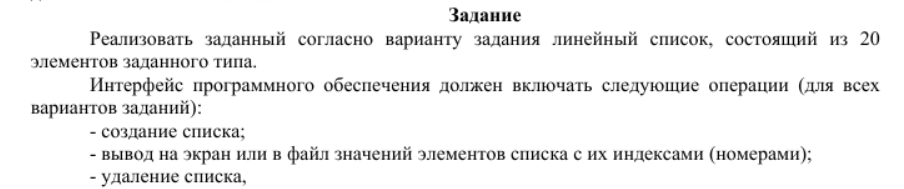
Проверил:

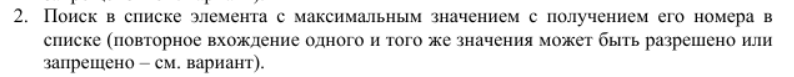
*Ким Роман Валерьевич*

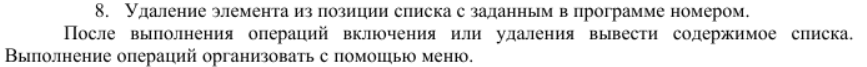
**Москва 2020**

**Задание:**

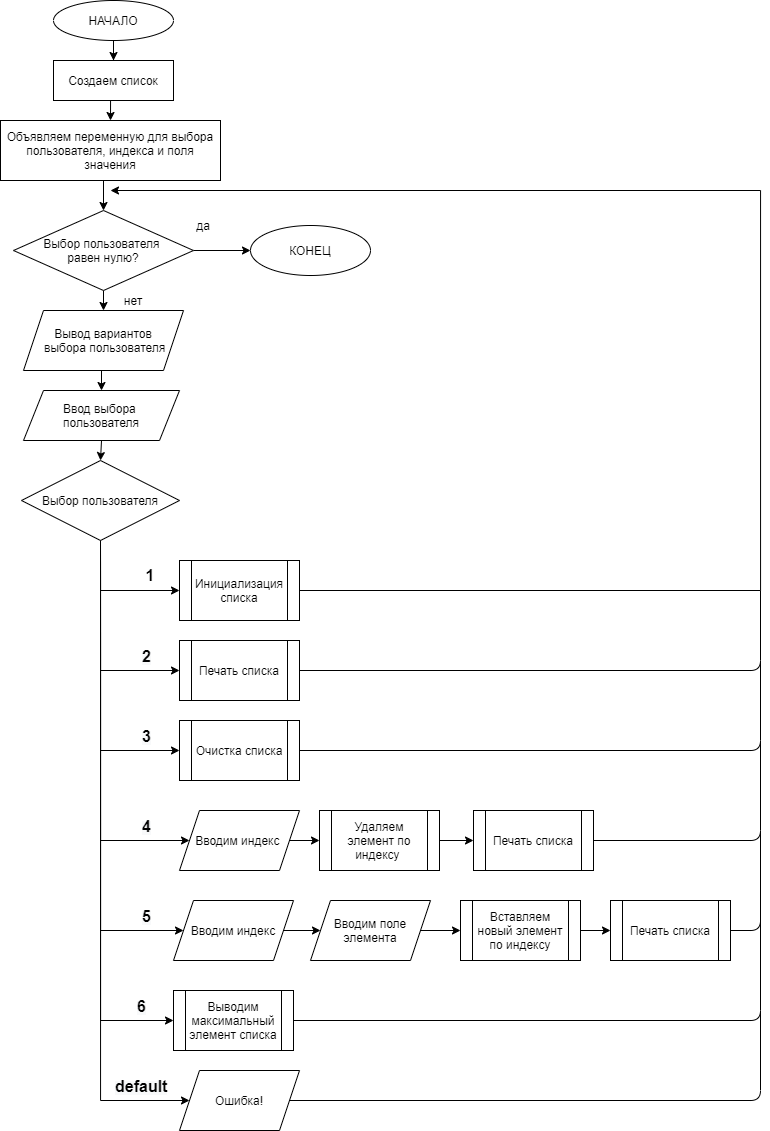
**Вариант №6**

****

****

**Структурная схема программы**

****

**Код программы**

**int main() {**

**List\* list = nullptr;**

**int choose = 1;**

**int index;**

**double field;**

**while (choose != 0) {**

**cout << "1.Init " << "2.Print " << "3.Clear " << "4.Delete " << "5.Insert " << "6.FindMax " << "0.Exit\n";**

**cin >> choose;**

**switch (choose) {**

**case 1:**

**Init(&list);**

**break;**

**case 2:**

**Print(list);**

**break;**

**case 3:**

**Clear(&list);**

**break;**

**case 4:**

**cout << "Enter index:";**

**cin >> index;**

**Delete(&list, index);**

**Print(list);**

**break;**

**case 5:**

**cout << "Enter index:";**

**cin >> index;**

**cout << "Enter field:";**

**cin >> field;**

**Insert(&list, field, index);**

**Print(list);**

**break;**

**case 6:**

**cout << "Max: " << FindMax(list) << endl;**

**break;**

**default:**

**cout << "Error!";**

**break;**

**}**

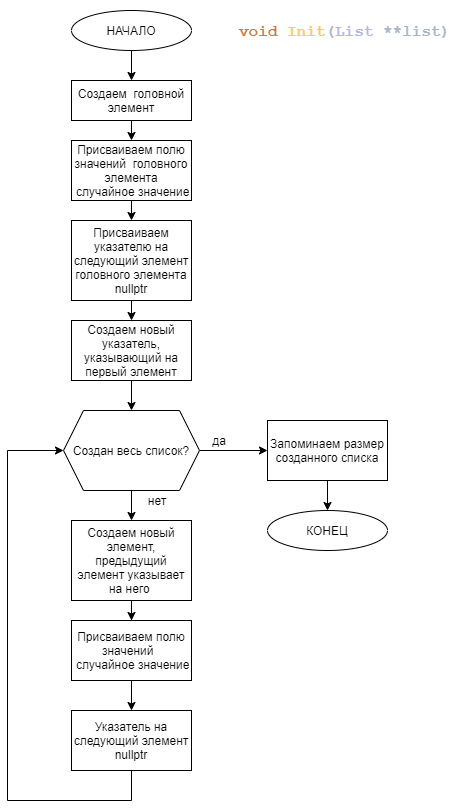
**}**

**return 0;**

**}**

**Структурная схема подпрограммы Init**

**Инициализация списка**

****

**Код подпрограммы Init**

**void Init(List\*\* list) {**

**\*list = new List();**

**(\*list)->field = rand() % 150;**

**(\*list)->pNext = nullptr;**

**List\* end = \*list; //указатель на первый элемент**

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

**end = end->pNext = new List(); //указатель на следующий элемент списка**

**end->field = rand() % 150;**

**end->pNext = nullptr;**

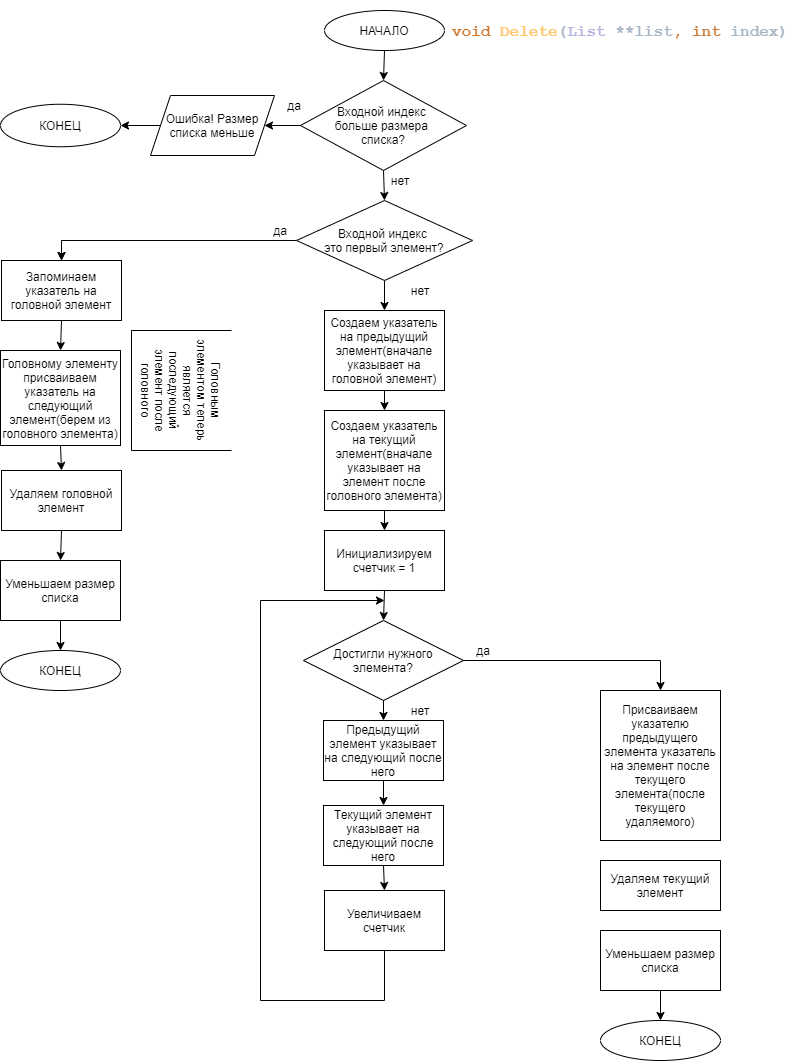
**}**

**(\*list)->amount = SIZE;**

**}**

**Структурная схема подпрограммы Delete**

**Удаление элемента из списка по индексу**

****

**Код подпрограммы Delete**

**void Delete(List\*\* list, int index) {**

**if (index > (\*list)->amount) {**

**cout << "List out of bounds!";**

**return;**

**}**

**if (index == 0) {**

**List\* temp = \*list;**

**\*list = (\*list)->pNext;**

**free(temp);**

**(\*list)->amount--;**

**return;**

**}**

**List\* prev = \*list;**

**List\* current = \*list;**

**current = current->pNext;**

**int counter = 1;**

**while (index > counter) {**

**prev = prev->pNext;**

**current = current->pNext;**

**counter++;**

**}**

**prev->pNext = current->pNext;**

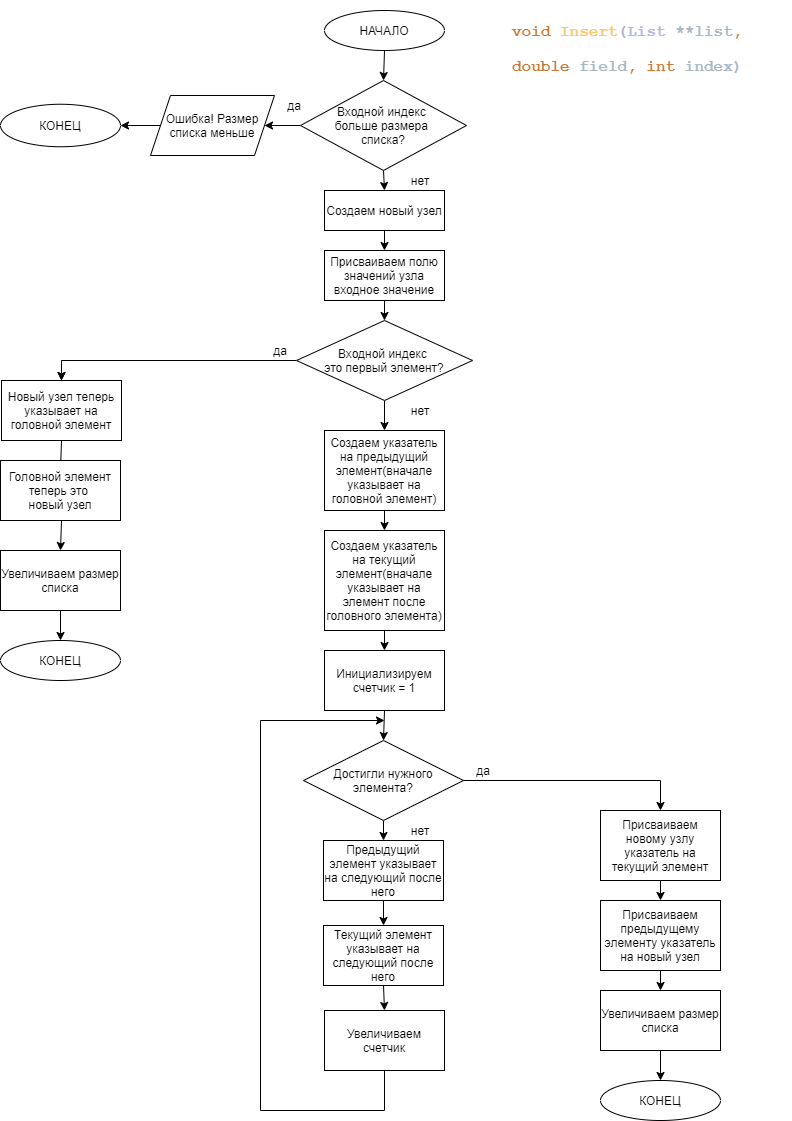
**free(current);**

**(\*list)->amount--;**

**}**

**Структурная схема подпрограммы Insert**

**Вставка элемента в список по индексу**

****

**Код подпрограммы Insert**

**void Insert(List\*\* list, double field, int index) {**

**if (index > (\*list)->amount) {**

**cout << "List out of bounds!";**

**return;**

**}**

**List\* temp = new List();**

**temp->field = field;**

**if (index == 0) {**

**temp->pNext = \*list;**

**\*list = temp;**

**(\*list)->amount++;**

**return;**

**}**

**List\* prev = \*list;**

**List\* current = \*list;**

**current = current->pNext;**

**int counter = 1;**

**while (index > counter) {**

**prev = prev->pNext;**

**current = current->pNext;**

**counter++;**

**}**

**temp->pNext = current;**

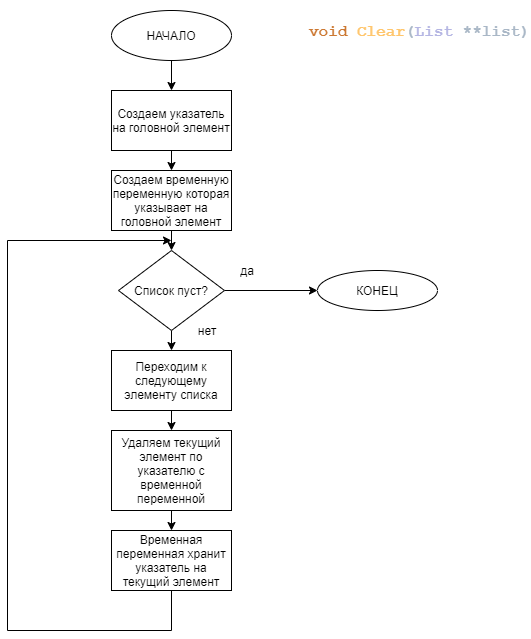
**prev->pNext = temp;**

**(\*list)->amount++;**

**}**

**Структурная схема подпрограммы Clear**

**Очистка списка**

****

**Код подпрограммы Clear**

**void Clear(List\*\* list) {**

**List\* pointer = \*list;**

**List\* temp = \*list;**

**while (pointer) {**

**pointer = pointer->pNext;**

**free(temp);**

**temp = pointer;**

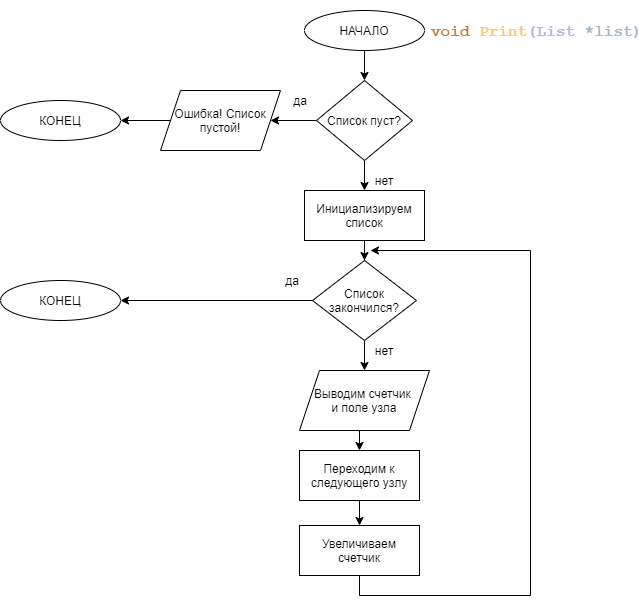
**}**

**\*list = nullptr;**

**}**

**Структурная схема подпрограммы Print**

**Печать списка**

****

**Код подпрограммы Print**

**void Print(List\* list) {**

**if (!list) {**

**cout << "List is empty!\n";**

**}**

**int counter = 0;**

**while (list) {**

**cout << counter << ": " << list->field << endl;**

**list = list->pNext; // переход к следующему узлу**

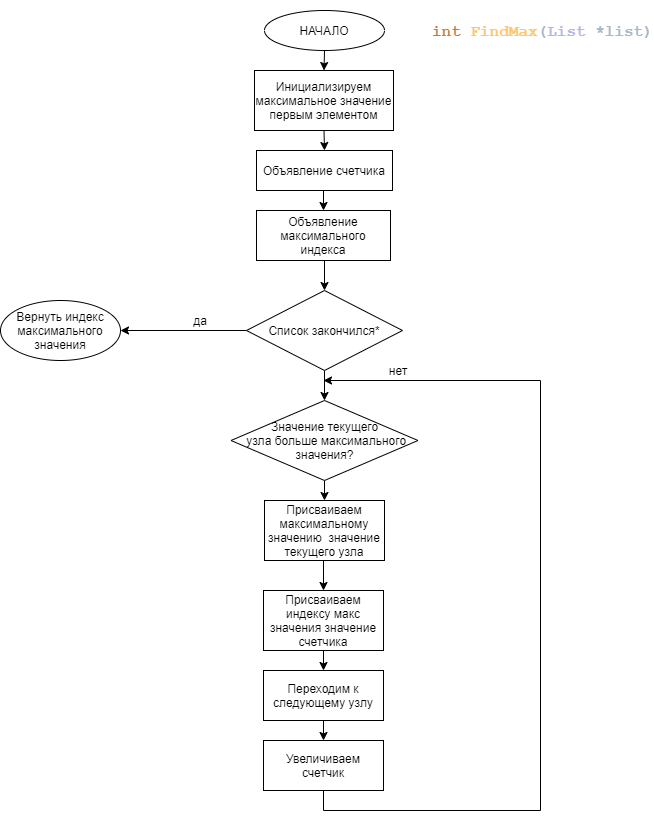
**counter++;**

**}**

**}**

**Структурная схема подпрограммы FindMax**

**Поиск максимального значения в списке**

****

**Код подпрограммы FindMax**

**int FindMax(List\* list) {**

**double max = list->field;**

**int counter = 0;**

**int indexMax;**

**while (list) {**

**if (list->field > max) {**

**max = list->field;**

**indexMax = counter;**

**}**

**list = list->pNext; // переход к следующему узлу**

**counter++;**

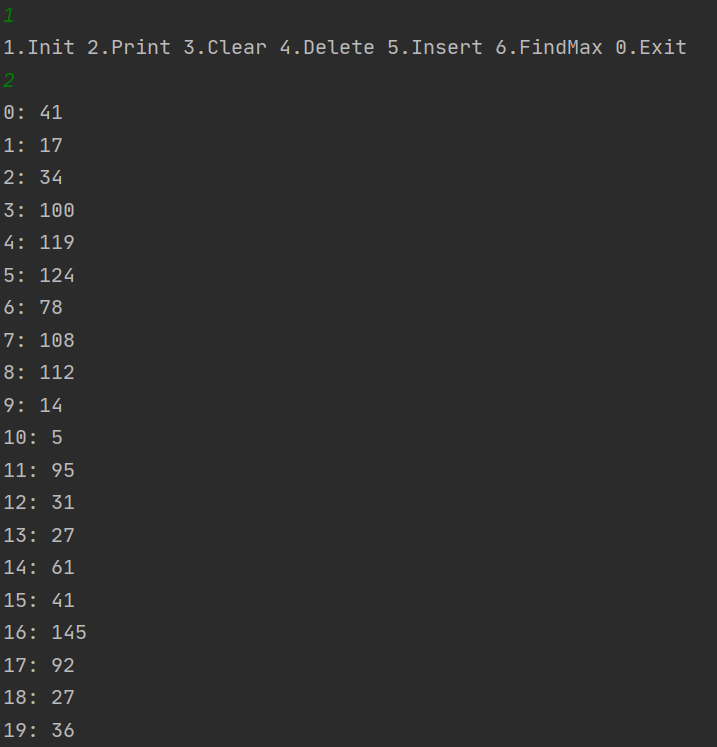
**}**

**return indexMax;**

**}**

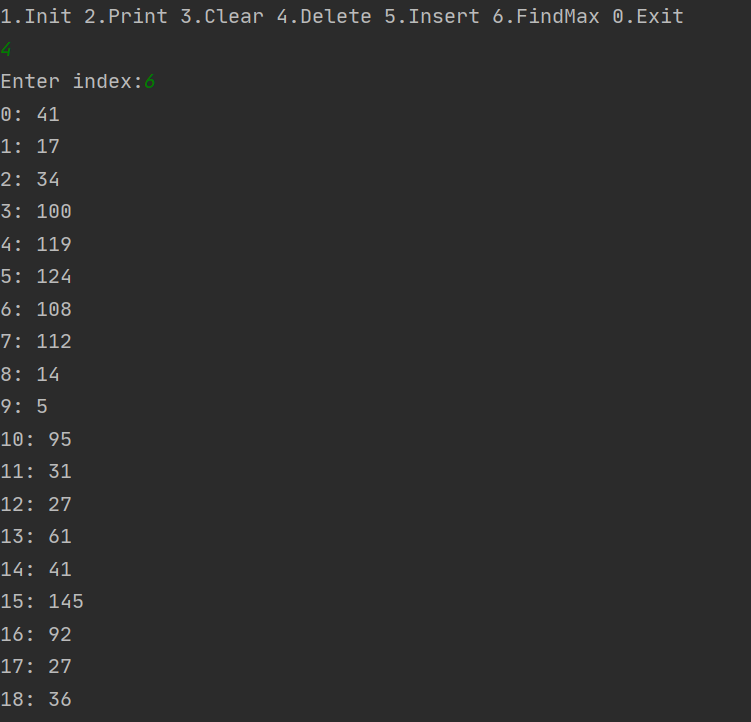
**Результаты программы:**

Инициализация списка

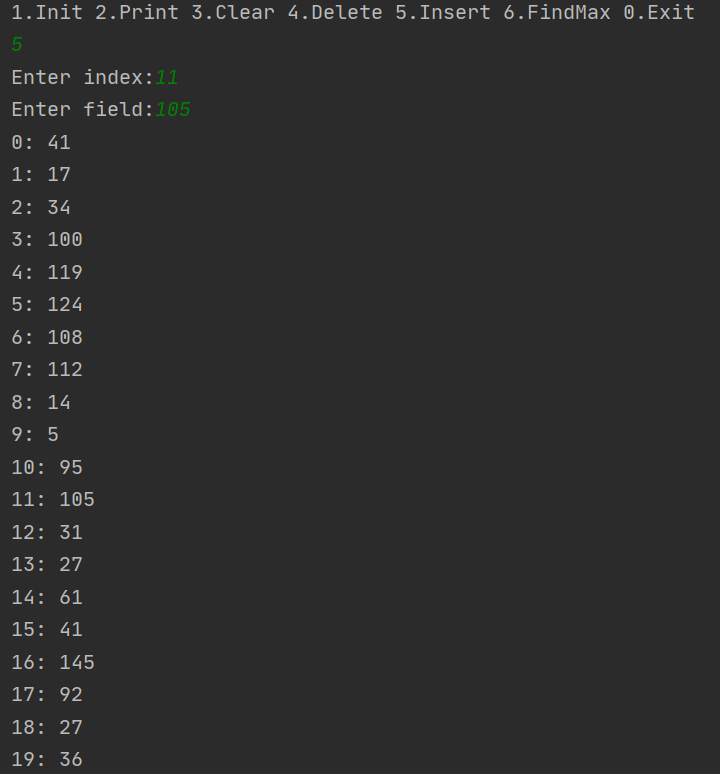


Удаление элемента по индексу

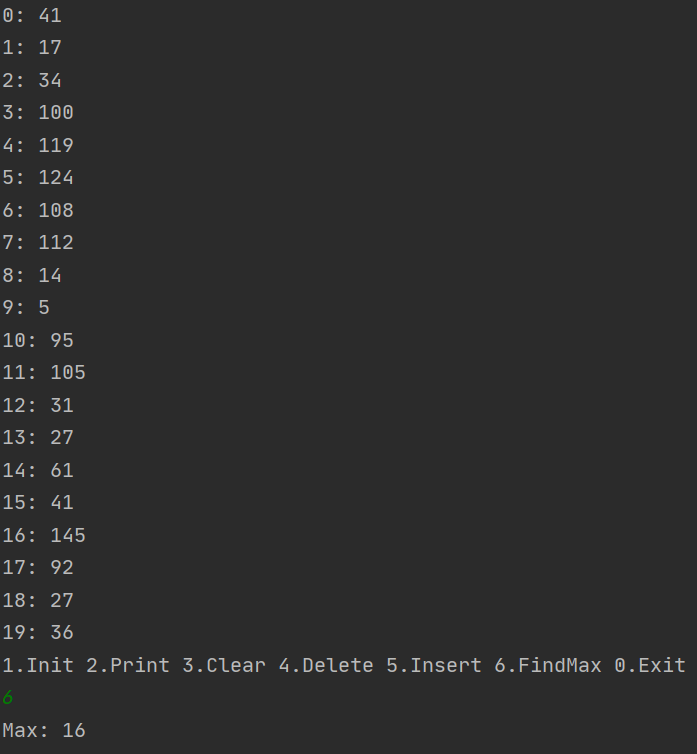
(на 6 индексе был 78)



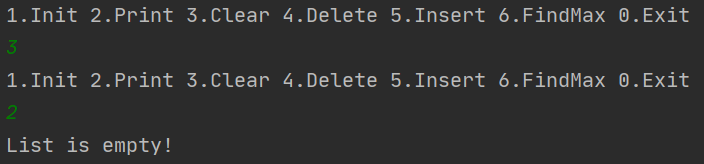
Вставка элемента по индексу



Поиск максимального значения(его индекс)



Очистка списка



**Вывод:**

Мы реализовали однонаправленный список и реализовали некоторые функции для работы с ним: инициализация списка, вывод списка, удаление списка, поиск максимального значения в списке, удаления элемента по его индексу, вставка элемента по индексу.

Таким образом, мы изучили принцип программной реализации одной из динамических структур данных – “однонаправленный линейный список”