Министерство науки и высшего образования РФ

Пензенский государственный университет

Кафедра «Вычислительная техника»

ОТЧЁТ

по лабораторной работе №11

по курсу «Программирование»

на тему «Динамические структуры данных»

Вариант-1

**Выполнили студентки группы**

**24ВВВ1:**

Куничкина В.А.

Суркова Д.А.

**Приняли:**

к.т.н., доцент Токарев А.Н.

Пенза 2024

**Название**

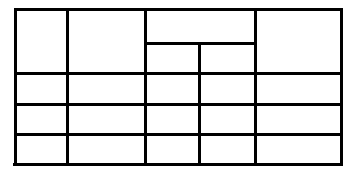
Динамические структуры данных

**Цель работы**

изучение способов организации и описания динамических структур, приобретение навыков работы с ними.

**Лабораторное задание**

1. Выбрать структуру данных



2. По номеру варианта выбрать предметную область. (Одежда)

3. Заполнить заданную структуру данными из выбранной предметной области.

4. Оформить структуру данных в виде динамического списка.

5. Разработать программу, осуществляющую:

а) ввод исходных данных в список, завершение ввода должно

отмечаться специальным символом (в список должно быть введено не

менее пяти записей);

б) вывод исходных данных на экран в виде таблицы;

в) перестановку местами первого и последнего элементов динамического списка;

г) вывод на экран модифицированного списка в виде таблицы.

6. Выполнить программу на компьютере и оценить правильность ее работы.

**Методические указания**

Данная работа подразумевает создание односвязного динамического списка с неограниченным количеством элементов. После ввода

очередного элемента списка на экран должен выдаваться запрос об

окончании ввода. Об окончании списка также может свидетельствовать ввод специального символа в 1-е поле элемента, например ‘\*’.

Перестановка элементов списка может быть произведена двумя способами: изменением содержимого полей связей и изменением содержимого информационных полей в соответствующих элементах списка.

Для выполнения лабораторной работы рекомендуется осуществлять

замену первым способом.

**Листинг**

#include <stdio.h>

#include <stdlib.h>

#include <conio.h>

#include <string.h>

#include <Windows.h>

struct Size {

char minS[100];

char maxS[100];

};

typedef struct Node {

char nom[10];

char nazv[100];

struct Size size; // Вложенная структура

float stoit;

struct Node\* next;

} Node;

typedef struct LinkedList {

Node\* head; // Указатель на первый элемент списка

} LinkedList;

// Функция для создания нового узла

Node\* createNode(const char\* nom, const char\* nazv, const char\* minS, const char\* maxS, float stoit) {

Node\* newNode = (Node\*)malloc(sizeof(Node));

strcpy(newNode->nom, nom);

strcpy(newNode->nazv, nazv);

strcpy(newNode->size.minS, minS);

strcpy(newNode->size.maxS, maxS);

newNode->stoit = stoit;

newNode->next = NULL;

return newNode;

}

// Функция для добавления узла в конец списка

void append(LinkedList\* list, const char\* nom, const char\* nazv, const char\* minS, const char\* maxS, float stoit) {

Node\* newNode = createNode(nom, nazv, minS, maxS, stoit);

if (list->head == NULL) {

list->head = newNode;

} else {

Node\* current = list->head;

while (current->next != NULL) {

current = current->next;

}

current->next = newNode;

}

}

void display(LinkedList\* list) {

Node\* current = list->head;

const char shapka[] =

"+-------------------------------------------------------------+\n"

"| | | размер | |\n"

"| # | Название |----------+---------| цена |\n"

"| | | мин | макс | |\n"

"+---+-----------------+----------+---------+------------------+\n";

printf("%s", shapka);

int index = 0; // Индекс для нумерации

while (current != NULL) {

printf("| %2d| %-15s | %8s | %7s | %16.2f |\n",

index + 1, current->nazv, current->size.minS, current->size.maxS, current->stoit);

printf("+---+-----------------+----------+---------+------------------+\n");

current = current->next; // Переход к следующему элементу

index++;

}

}

// Функция для перестановки первого и последнего элементов списка

void swapFirstLast(LinkedList\* list) {

if (list->head == NULL || list->head->next == NULL) {

return; // Список пуст или содержит только один элемент

}

Node\* first = list->head;

Node\* last = list->head;

Node\* prevToLast = NULL;

// Находим последний и предпоследний элементы

while (last->next != NULL) {

prevToLast = last;

last = last->next;

}

// Перестановка

if (prevToLast != NULL) {

prevToLast->next = first; // Предпоследний указывает на первый

last->next = first->next; // Последний указывает на второй элемент

list->head = last; // Новый голова списка - последний элемент

first->next = NULL; // Первый элемент теперь последний

}}

// Функция для освобождения памяти

void freeList(LinkedList\* list) {

Node\* current = list->head;

while (current != NULL) {

Node\* next = current->next;

free(current);

current = next;

}

list->head = NULL; // Обнуляем указатель на голову списка

}

int main() {

LinkedList list;

list.head = NULL;

SetConsoleOutputCP(1251);

SetConsoleCP(1251);

while (1) {

char nom[10];

char nazv[100];

char minS[100];

char maxS[100];

float stoit;

printf("Введите номер одежды (или '\*' для завершения ввода): ");

scanf("%s", nom);

if (strcmp(nom, "\*") == 0) {

break; }

printf("Введите название одежды: ");

scanf("%s", nazv);

printf("Введите min размер: ");

scanf("%s", minS);

printf("Введите max размер: ");

scanf("%s", maxS);

printf("Введите цену: ");

scanf("%f", &stoit);

append(&list, nom, nazv, minS, maxS, stoit);

}

printf("\nИсходные данные:\n");

display(&list);

swapFirstLast(&list);

printf("\nCписок после перестановки первого и последнего элементов:\n");

display(&list);

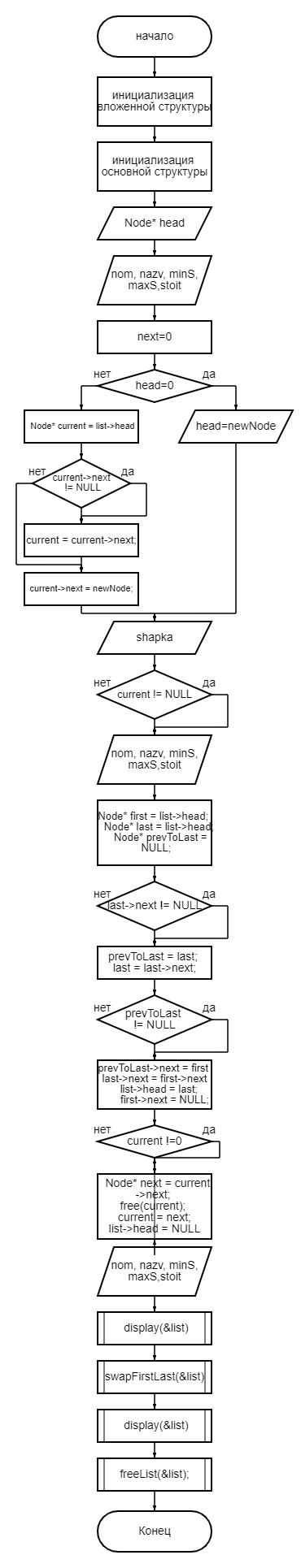
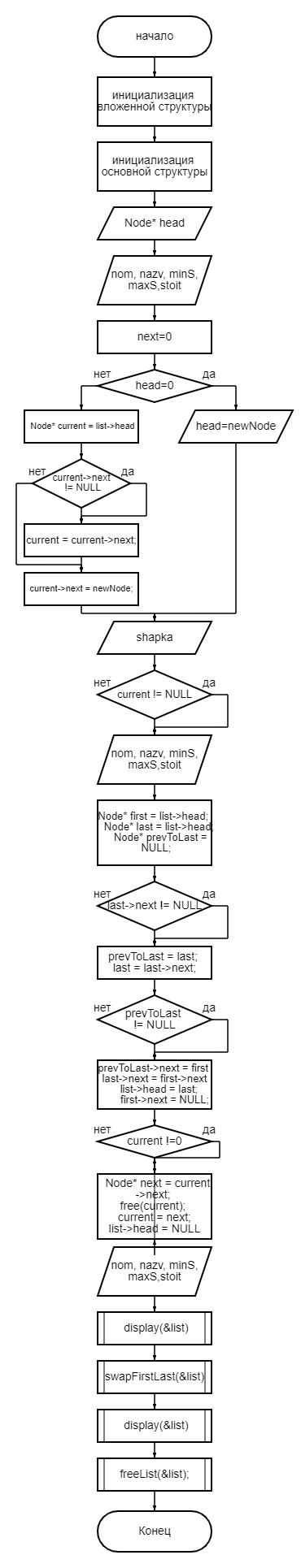
freeList(&list);

getch();

return 0;

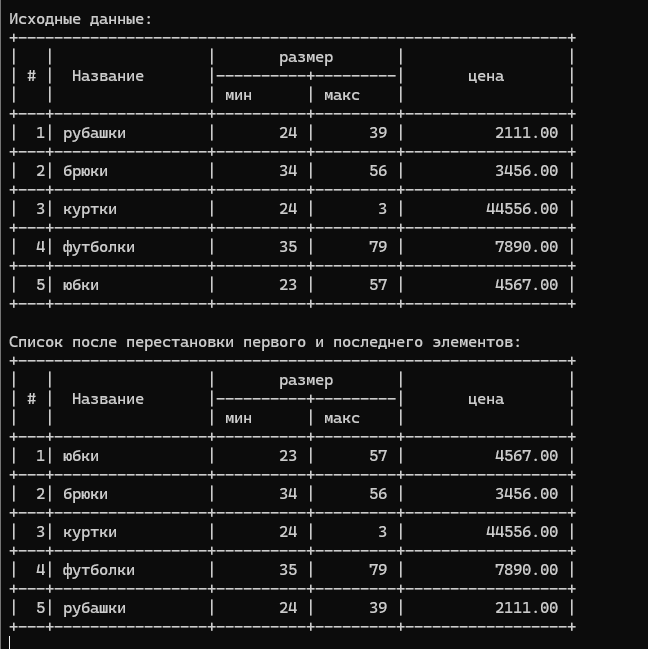
}

**Схема программы**

**Результаты работы программы**

Результаты работы программы показаны на рисунке 1.



**Рис.1 — Результаты работы программы**

**Выводы**

В ходе выполнения лабораторной работы были изучены динамические структуры данных и их использование.

Получили опыт в создании проектов в среде Microsoft Visual Studio, научились создавать программу на языке СИ.