# НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ

імені Ігоря Сікорського»

Факультет прикладної математики

Кафедра прикладної математики

Звіт

із лабораторної роботи №*7*

із дисципліни «Програмування»

на тему

*«Однозв’язні списки»*

|  |  |
| --- | --- |
| Виконав: | Керівник: |
| студент групи КМ-63 | *асистент Громова В. В.* |
| *Артеменко Я.К.* |  |

Київ — 2017

# **ЗМІСТ**

[1 ПОСТАНОВКА ЗАДАЧІ 2](#_Toc478382182)

[1.1 Мета роботи 2](#_Toc478382183)

[1.2 Завдання на лабораторну роботу 2](#_Toc478382184)

[2 ОПИС ПРОГРАМИ 3](#_Toc478382185)

[ВИСНОВКИ 5](#_Toc478382186)

[Додаток А (блок-схема) 5](#_Toc478382187)

[Додаток Б (відповіді на контрольні запитання) 7](#_Toc478382188)

[Додаток В (текст програми) 8](#_Toc478382189)

# **1 ПОСТАНОВКА ЗАДАЧІ**

* 1. Мета роботи

Набути практичних навичок розроблення програм, які працюють із динамічними структурами даних.

* 1. Завдання на лабораторну роботу

Варіант №13

1. Описати структуру з іменем PRICE, що містить такі поля:
2. Назва товару;
3. Назва магазину, у якому продається товар;
4. Вартість товару (у грн);
5. Написати програму, що виконує такі дії:
6. Уведення з клавіатури структур за допомогою діалогу із користувачем про потребу продовжувати уведення;
7. Упорядкування записів за алфавітом за назвами магазинів;
8. Виведення на екран інформації про товари, що продаються в магазині, назву якого уведено з клавіатури; якщо таких товарів немає, вивести на екран відповідне повідомлення.

**Постановка завдання**

Програма повинна задовільнять такі вимоги:

1. Указані в завданні дії має бути оформлено як окремі функції.
2. Пам’ять під черговий елемент динамічної структури даних потрібно виділяти динамічно.
3. Виконання дій з обслуговування динамічної структури даних повинно виконуватися в режимі діалогу з користувачем.
4. Уведення структур можна реалізувати одним із таких механізмів:

* уведення заздалегідь вибраної кількості структур (не менше за 5);
* уведення до появи заданої ознаки закінчення уведення структур;
* діалог із користувачем про потребу продовжувати уведення.

Для кожного з варіантів завдання потрібно розробити такі функції:

* створення списку;
* додавання елемента в список:
  1. у початок списку;
  2. у кінець списку;
  3. після елемента з заданим номером;
  4. після елемента з заданим ключем;
* видалення елемента зі списку:

1. з початку списку;
2. з кінця списку;
3. із заданим номером;
4. із заданим ключем;

* виведення вмісту списку на екран;
* упорядкування елементів списку за вибраною ознакою.

# **ОПИС ПРОГРАМИ**

Дана програма містить 11 функцій: menu\_main, integer, valid\_bit, struct PRICE \*create, menu, head\_table, PRICE \*add, PRICE \*del, sort, find, main. Ще програма структуру PRICE з полями, вказаними в п. 1.2.

Функція **menu\_main** виводить на екран дані про виконавця та надає можливість користувачу обрати завдання для виконання. Обрати завдання можна за допомогою стрілок «вверх» і «вниз» і за допомогою клавіші «Enter» перейти до його виконання. Пункти, які ми можемо обрати підсвічені білим кольором, а умови завдань виділяються сірим кольором. За допомогою оператора розгалуження «if – else» зроблено можливість переміщення «стрілки вибору» між пунктами меню та за допомогою десяткових індексів функціональних клавіш («80» – для стрілки вниз, а «72» – для стрілки вверх) зроблено можливим використання клавіш для вибору. За допомогою циклу «while» очікується натиснення клавіші Enter, після чого ця функція повертає значення вибраного пункту в функцію main.

Функції **integer та valid\_bit** перевіряють правильність введених даних. Якщо буде введена буква, а не цифра, то програма покаже повідомлення про помилку.

Функція **head\_table** задає заголовок таблиці і записує в неї введені дані.

Функція **sort** сортує введені дані по алфавіту за ключем, вказаним в умові задачі.

Функція **struct PRICE \*create** організовує введення даних у структуру з можливістю діалогу з користувачем про продовження введених даних.

Функція **menu** дозволяє обрати одну з операцій по роботі із структурую та виконати її.

Функція **struct PRICE \*add** організовує додавання запису по обраному методу (в початок, в кінець, після введеного номеру, після введеної назви магазину).

Функція **struct PRICE \*del** організовує видалення запису по обраному методу (початковий елемент, останній елемент, по введеному номеру, по введеній назві магазину).

Функція **find** організовує пошук та виведення інформації про магазин, введений з клавіатури. Якщо такого магазину немає, то виводиться відповідне повідомлення.

Функція **main** є головною і в ній викликаються всі функції і забезпечується робота меню за допомогу оператора «case».

# **ВИСНОВКИ**

На цій лабораторній роботі здобуто навички по роботі з типами даних, які визначені користувачем – структурами, та набуто практичні навички їх використання для написання програм.

# **Додаток А (блок-схема)**

Функція menu\_main:

Початок

Printf(“Умова завдання”)

Printf(“Інформація про виконавця”)

System(“cls”);

ні

If key = {0}

так

Cout<<”-> Продовжити”

Cout<<” Продовжити”

If key = {1}

ні

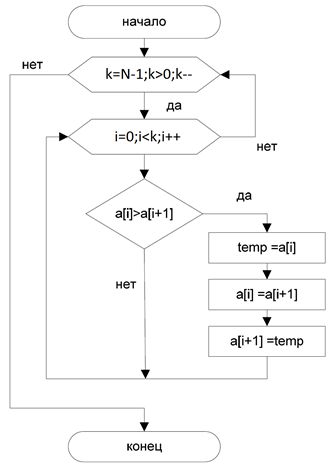
так

Cout<<”-> Вихід”

Cout<<” Вихід”

Кінець

Функція sort:



Функція input\_note:

Початок

while (c != 'y')

pot->uah[0] = valid\_bit(10000, 1);

scanf("%s", &pot->name\_product);

printf("Введите название магазина(Вводите текст): ");

scanf("%s", &pot->name\_shop);

printf("Введите название продукта(Вводите текст): ");

cout << "Введите к-во гривен: ";

cout << "Введите к-во копеек: ";

pot->uah[1] = valid\_bit(10000, 1);

# 

Кінець

# **Додаток Б (відповіді на контрольні запитання)**

1. Список – це послідовність структур, кожна з яких містить посилання, яке зв’язує її з іншою структурою. Для організації списків використовують структури, які складаються з двох частин – інформаційної та допоміжної:

* Інформаційна частина містить інформацію, яка підлягає опрацюванню;
* Додаткова частина містить вказівники на наступну чи попередню структури списку.

Елемент списку називають вузлом. Останній вузол не посилається на наступний елемент, тому його поле посилання має значення NULL. За однозв’язним списком можна рухатися тільки в одному напрямку — від заголовкового (першого) вузла до останнього.

1. Щоб видалити структуру із списку потрібно вказівнику на наступний елемент присвоїти вказівник на наступний наступного.
2. Потрібно виділити пам’ять для елементу, присвоїти значення початку списку вказівнику на наступний.

# **Додаток В (текст програми)**

#include <stdio.h>

#include <conio.h>

#include <windows.h>

#include <stdlib.h>

#include <iostream>

#include <locale.h>

#include <string>

#include <iomanip>

using namespace std;

char ch;

struct PRICE \*head;

int main();

void head\_table();

struct PRICE \*add();

void sort();

struct PRICE \*del();

void find();

typedef struct PRICE

{

char name\_shop[15];

char name\_product[15];

int uah[2];

struct PRICE \*next;

}list;

int menu\_main()

{

HANDLE hStdOut = GetStdHandle(STD\_OUTPUT\_HANDLE);

char key = 0;

int code;

do

{

system("cls");

printf("\t \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\t \* Лабораторная работа №7 \*\n");

printf("\t \* Типы данных, \*\n");

printf("\t \* определенные пользователем \*\n");

printf("\t \* Вариант №13 \*\n");

printf("\t \* Группа KM-63 Артеменко Я.К.\*\n");

printf("\t \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("\n\n");

key = (key + 2) % 2;

SetConsoleTextAttribute(hStdOut, (WORD)((0 << 4) | 8));

printf("\tИНСТРУКЦИЯ!!! ДЛЯ НАВИГАЦИИ ПО МЕНЮ ИСПОЛЬЗУЙТЕ СТРЕЛКИ ВВЕРХ, ВНИЗ И КЛАВИШУ ENTER\n");

printf("\t1. Описать структуру с именем NOTE, которая имеет такие поля:\n");

printf("\t - фамилия, имя; \n");

printf("\t - номер телефона;\n");

printf("\t - день рождения (массив из трех чисел);\n\n");

printf("\t2. Написать программу, которая выполняет такие действия:\n");

printf("\t - ввод с клавиатуры данных у массив из 5 структур типа NOTE;\n");

printf("\t - упорядочения записей за алфавитом; \n");

printf("\t - вывод на экран информации о людях, чьи дни рождения \n");

printf("\t приходятся на месяц, значение которого введено с клавиатуры;\n");

printf("\t если таких людей нет, вывести на экран соответствующее сообщение.\n\n\n");

SetConsoleTextAttribute(hStdOut, (WORD)((0 << 4) | 15));

if (key == 0) cout << "\t -> Продолжить. \n" << endl;

else cout << "\t Продолжить.\n" << endl;

if (key == 1) cout << "\t -> Выход.\n" << endl;

else cout << "\t Выход.\n" << endl;

code = \_getch();

if (code == 224)

{

code = \_getch();

if (code == 80) key++;

if (code == 72) key--;

}

} while (code != 13);

system("cls");

return key;

}

int integer() /\*валідація для цілих чисел\*/ {

char symbol[20];

int z = 0, k, res;

do

{

fflush(stdin); /\*занулення буфера від попередніх записів\*/

scanf("%s", &symbol);

while (symbol[z] >= '0' && symbol[z] <= '9')

z++;

res = atoi(symbol);

if (symbol[z])

printf("Entered symbol is not an integer number! Try again: ");

else

return res;

} while (symbol[z]);

}

int valid\_bit(int a, int b) /\*валідація для бітових операцій на цілі числа (в залежності від потрібних значень)\*/ {

int c;

do

{

c = integer();

if (c < b || c > a)

printf("Too many items! Try again: ");

} while (c < b || c > a);

return c;

}

struct PRICE \*create()

{

list \*pot=NULL, \*pred=NULL;

char c='n';

bool flag = 0;

while (c != 'y')

{

system("cls");

pot = (list\*)malloc(sizeof(list));

if (!flag)

{

head = pred = pot;

flag = 1;

}

printf("Введите название магазина(Вводите текст): ");

scanf("%s", &pot->name\_shop);

printf("Введите название продукта(Вводите текст): ");

scanf("%s", &pot->name\_product);

printf("Введите цену(вводить только числа):\n ");

cout << "Введите к-во гривен: ";

pot->uah[0] = valid\_bit(10000, 1);

cout << "Введите к-во копеек: ";

pot->uah[1] = valid\_bit(10000, 1);

/\*посилання з попереднього запису

на поточний\*/

pred->next = pot;

/\*збереження адреси поточного запису

в полі попереднього\*/

pred = pot;

printf("Закончить? y/n (YES/NO)");

c = \_getch();

};

pot->next = NULL;

return head;

}

int menu()

{

while (1)

{

while (1)

{

system("cls");

printf("Выберите один из вариантов:\n\n");

printf("1. Создать и заполнить список.\n");

printf("2. Показать список.\n");

printf("3. Добавить элемент.\n");

printf("4. Удалить элемент.\n");

printf("5. Отсортировать список.\n");

printf("6. Поиск по списку.\n");

printf("7. Выход в меню.\n\n");

printf("Введите номер пунка:");

ch = \_getch();

printf("\n");

if (ch == '1' || ch == '2' || ch == '3' || ch == '4' || ch == '5' || ch == '6'|| ch=='7')

break;

system("cls");

printf("Вы ввели неправильный номер.\n\n");

}

switch (ch)

{

case '1':

{

head = create();

break;

}

case '2':

{

system("cls");

head\_table();

system("pause");

break;

}

case '3':

{

system("cls");

add();

system("pause");

break;

}

case '4':

system("cls");

del();

system("pause");

break;

case '5':

{

system("cls");

sort();

head\_table();

system("pause");

break;

}

case '6':

{

system("cls");

head\_table();

find();

system("pause");

break;

}

case '7': {main(); break; }

}

}

\_getch();

//free(head);

}

void head\_table()

{

int i = 1;

struct PRICE \*cur=head;

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

printf("| #| Название магазина | название товара | Цена(грн) |\n");

printf("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n");

while (cur)

{

cout <<"|"<<setw(2)<< i++ <<"|"<<setw(21)<<cur->name\_shop <<"|"

<<setw(19)<<cur->name\_product<<"|"<<setw(12)<<cur->uah[0]<<"."<< cur->uah[1] <<"|"<< endl;

cur = cur->next;

}

}

struct PRICE \*add()

{

system("cls");

printf("Виберите, куда вы хотите добавить элемент:\n");

printf("1.В начало.\n");

printf("2.В конец.\n");

printf("3.После введенного номера.\n");

printf("4.После введенного названия магазина.\n");

ch = \_getch();

system("cls");

switch (ch)

{

case '1':

{

list \*pot;

pot = (list\*)malloc(sizeof(list));

printf("Введите название магазина(Вводите текст): ");

scanf("%s", &pot->name\_shop);

printf("Введите название продукта(Вводите текст): ");

scanf("%s", &pot->name\_product);

printf("Введите цену(вводить только числа):\n ");

cout << "Введите к-во гривен: ";

pot->uah[0] = valid\_bit(10000, 1);

cout << "Введите к-во копеек: ";

pot->uah[1] = valid\_bit(10000, 1);

pot->next = head;

head = pot;

return pot;

};

case '2':

{

list \*p;

p = head;

while (p->next != NULL)

{

p = p->next;

}

p->next = (list\*)malloc(sizeof(list));

p = p->next;

printf("Введите название магазина(Вводите текст): ");

scanf("%s", &p->name\_shop);

printf("Введите название продукта(Вводите текст): ");

scanf("%s", &p->name\_product);

printf("Введите цену(вводить только числа):\n ");

cout << "Введите к-во гривен: ";

p->uah[0] = valid\_bit(10000, 1);

cout << "Введите к-во копеек: ";

p->uah[1] = valid\_bit(10000, 1);

p->next = NULL;

return head;

};

case '3':

{

int num;

int i = 1;

list \*p, \*next;

p = head;

head\_table();

cout << "Введите номер: ";

while (getchar() != '\n');

num=valid\_bit(100,1);

while (i < num)

{

p = p->next;

i++;

}

next = p->next;

p->next = (list\*)malloc(sizeof(list));

p = p->next;

system("cls");

printf("Введите название магазина(Вводите текст): ");

scanf("%s", &p->name\_shop);

printf("Введите название продукта(Вводите текст): ");

scanf("%s", &p->name\_product);

printf("Введите цену(вводить только числа):\n ");

cout << "Введите к-во гривен: ";

p->uah[0] = valid\_bit(10000, 1);

cout << "Введите к-во копеек: ";

p->uah[1] = valid\_bit(10000, 1);

p->next = next;

//p->next = p->next->next;

return head;

};

case '4':

{

char name[30];

list \*p, \*next;

p = head;

head\_table();

printf("\n\nВведите название магазина(Вводите текст): ");

while (getchar() != '\n');

gets\_s(name);

while (strcmp(p->name\_shop, name) != 0)

{

p = p->next;

}

next = p->next;

p->next = (list\*)malloc(sizeof(list));

p = p->next;

system("cls");

printf("Введите название магазина(Вводите текст): ");

scanf("%s", &p->name\_shop);

printf("Введите название продукта(Вводите текст): ");

scanf("%s", &p->name\_product);

printf("Введите цену(вводить только числа):\n ");

cout << "Введите к-во гривен: ";

p->uah[0] = valid\_bit(10000, 1);

cout << "Введите к-во копеек: ";

p->uah[0] = valid\_bit(10000, 1);

p->next = next;

return head;

};

}

}

struct PRICE \*del()

{

system("cls");

printf("Виберите, где вы хотите удалить єлемент:\n");

printf("1.В начале.\n");

printf("2.В конце.\n");

printf("3.По введенному номеру.\n");

printf("4.По названию магазина.\n");

ch = \_getch();

system("cls");

switch (ch)

{

case '1':

{

list \*pot;

pot = head;

head = head->next;

free(pot);

return head;

};

case '2':

{

list \*p;

p = head;

while (p->next->next != NULL)

{

p = p->next;

}

p->next = NULL;

return head;

};

case '3':

{

int num;

int i = 1;

list \*p,\*t=NULL;

p = head;

head\_table();

printf("\n\nВведите номер пункта, который хотите удалить: ");

while (getchar() != '\n');

num = valid\_bit(100, 1);

if (num == 1)

{

t = p->next;

free(p);

head = t;

return t;

}

while (i < num - 1)

{

p = p->next;

i++;

}

t = p->next;

p->next = t->next;

free(t);

return head;

};

case '4':

{

char name[30];

list \*p,\*t=NULL;

head\_table();

fflush(stdin);

printf("Введите название магазина, который хотите удалить(Вводите текст): \n");

while (getchar() != '\n');

scanf("%s", name);

p = head;

if (strcmp(head->name\_shop, name) == 0)

{

t = p->next;

free(p);

head = t;

return t;

}

while ((p->next != NULL) && (strcmp(p->next->name\_shop, name) != 0))

p = p->next;

if (p->next != NULL)

{

t = p->next;

p->next = t->next;

free(t);

}

return head;

};

}

}

void sort()

{

PRICE \*p;

for (PRICE \*i = head; i != NULL; i=i->next)

{

for (PRICE \*j = head; j !=NULL; j=j->next)

{

if (strcmp(i->name\_shop, j->name\_shop) < 0)

{

swap(j->name\_shop, i->name\_shop);

swap(j->name\_product, i->name\_product);

swap(j->uah[0], i->uah[0]);

swap(j->uah[1], i->uah[1]);

}

}

}

}

void find()

{

int i = 0;

list \*p;

p = head;

char wher[20];

system("cls");

head\_table();

printf("Введите название магазина, который хотите найти(Вводите текст): ");

while (getchar() != '\n');

gets\_s(wher);

printf("\n");

while (p != NULL)

{

if (strcmp(p->name\_shop, wher) == 0)

{

cout << "|" << setw(2) << ++i << "|" << setw(21) << p->name\_shop << "|"

<< setw(19) << p->name\_product << "|" << setw(12) << p->uah[0] << "." << p->uah[1] << "|" << endl;

}

p = p->next;

}

if (i == 0)

{

printf("Такого магазина нету в списке.");

}

while (1)

{

printf("Выберите один из вариантов\n:");

printf("1. Повторного поиск.\n");

printf("2. Новое заповнение.\n");

printf("3. Возвращение в начальное меню.");

printf("\n\nВведите номер пункта: ");

ch = \_getch();

printf("\n");

if (ch == '1' || ch == '2' || ch == '3')

break;

system("cls");

printf("Вы ввели неправильный номер.\n\n");

}

switch (ch)

{

case '1':

find();

case '2':

menu();

case '3':

main();

}

}

int main()

{

while (1)

{

setlocale(LC\_ALL, "RUS");

int i1 = menu\_main() + 1;

switch (i1)

{

case 1: {menu(); break; }

case 2: {exit(0); }

}

}

}