МІНІСТЕРСТВО ОСВІТИ І НАУКИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ “ХПІ”

Кафедра “Обчислювальна техніка та програмування”

Розрахункове завдання з дисципліни

«Основи програмування ч.2»

Пояснювальна записка

ЄСПД ГОСТ 19.404–79(СТЗВО – ХПІ – 06.06-2021 ССОНП)

КІТ.120А.14-01 90 01-1 -ЛЗ

Виконала:

студентка групи КІТ-120А

Клименко Станіслава Олександрівна

Перевірив:

Давидов В'ячеслав Вадимович

Харків 2021

Розрахункове завдання

Тема: *Розробка інформаційно-довідкової системи*

Мета: *Закріпити отримані знання з дисципліни «Програмування» шляхом виконання типового комплексного завдання.*

1. Призначення та галузь застосування

Розроблена інформаційно-довідкова система, являє собою колекцію взуття та методи роботи з нею. Згідно заданого завдання колекція має методи: пошуку ортопедичного взуття брендів Найк і Пума, пошук найдешевших бігових кросівок, пошук чобіт розміром більш за 39 та сортування списку за вказаним користувачем критерієм і напрямком. Також є методи роботи зі списком, які дають змогу: видалити задане користувачем взуття зі списку, або весь список, додати взуття до списку, замінити чи отримати взуття по індексу.

Створену інформаційно-довідкову систему можна застосовувати в галузі роботи з взуттям, наприклад інтернет-магазин, або веб-каталог.

2. Постановка завдання до розробки

1.1 Загальне завдання

1. З розділу ”Розрахункове завдання / Індивідуальні завдання”, відповідно до варіанта завдання, обрати прикладну галузь;
2. Для прикладної галузі розробити розгалужену ієрархію класів, що описана у завдані та складається з одного базового класу та двох спадкоємців. Класи повинні мати перевантажені оператори введення-виведення даних та порівняння;

3. Розробити клас-список `List.[h/cpp]`, що буде включати до себе масив (STL-колекцію) вказівників до базового класу. А також базові методи роботи з списком: а) очистка списку б) відображення списку в) додання/видалення/отримання/оновлення елементу;

4. Розробити клас-контроллер `Controller.[h/cpp]`, що буде включати колекцію розроблених класів, та наступні методи роботи з цією колекцією: а) читання даних з файлу та їх запис у контейнер (STL-контейнер); б) запис даних з контейнера у файл; в) сортування елементів у контейнері за вказаними критеріями: поле та напрям сортування, які задаються користувачем з клавіатури; г) пошук елементів за вказаними критеріями (три критерія, що присутні у кожному варіанті);

5. Розробити клас `Menu.[h/cpp]`, який має відображати діалогове меню для демонстрації реалізованих функцій класу контролера;

6. Оформити схеми алгоритмів функцій класів контролера (за необхідністю), тесту-контролера та діалогового меню;

7. Оформити документацію: пояснювальну записку.

Додаткові вимоги на оцінку «відмінно»:

- виконати перевірку вхідних даних за допомогою регулярних виразів.
- критерій для пошуку та сортування задавати у вигляді функтора;
- розробити клас-тестер контролеру `ControllerTest.cpp`, основною метою якого буде перевірка коректності роботи класу-контролера.

1.2 Індивідуальне завдання

– Варіант 14. "Взуття"

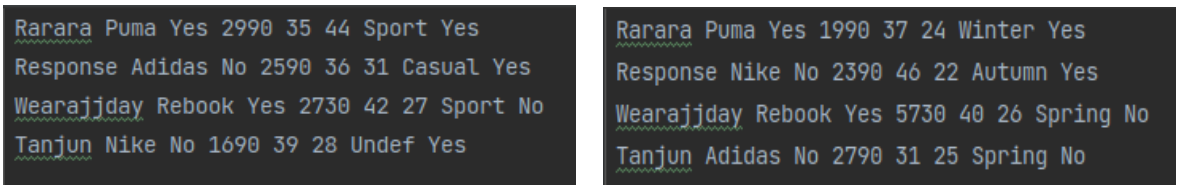
- Поля базового класу:
    - Чи є ортопедичним (наприклад: так, ні)
    - Назва моделі (наприклад: SuperStar, AirForse)
    - Ціна, USD (наприклад: 300, 1200)
    - Розмір (структура, що містить розмір та довжину устілки)
    - Бренд (один з переліку: adidas, puma, nike, rebook)
- Спадкоємець 1 – Кросівки. Додаткові поля:
    - Чи є біговими (наприклад: так, ні)
    - Призначення (один з переліку: sport, casual, undef)
- Спадкоємець 2 – Чоботи. Додаткові поля:
    - Сезон (один з переліку: зима, осінь, весна)
    - Наявність антиковзаючої підошви (наприклад: так, ні)
- Методи роботи з колекцією:
    1. Знайти ортопедичне взуття брендів найк і пума.
    2. Знайти найдешевші бігові кросівки.
    3. Знайти чоботи з розміром більше 39.

3. Опис вхідних та вихідних даних

3.1 Опис вхідних даних

Під час запуску програма зчитує дані з файлу за шляхом «Sneakers.txt» та «Boots.txt». В файл повинен містити наступні параметри: перший, це строка, яка позначає назву моделі вхідного об’єкту, наступний строка, яка позначає назву бренда вхідного об’єкту, потім строка “Yes / No” що

відповідає за те, что є взуття ортопедичним. Наступна цифра відповідає за ціну взуття. Потім цифра, що відповідає за розмір устілки ,а наступна цифра відповідає за довжину устілки. Наступна строка відповідає за призначення взуття(у кросівок) та за сезон(у чобіт). Остання трока “Yes / No”, відповідає за те чи є кросівки біговими, та чи наявна антиковзаюча підошва у чобіт. Приклад файлу з вхідними даними продемонстровані на рисунку 1.



а б

Рисунок 1 – Приклад вхідного файлу (а - кросівки, б - чоботи)

3.2 Опис вихідних даних

Вихідні дані записуються у файл розташований за шляхом «result.txt», в тому ж порядку, в якому задані днні у списку. Приклад файлу з вихідними даними продемонстровані на рисунку 2.

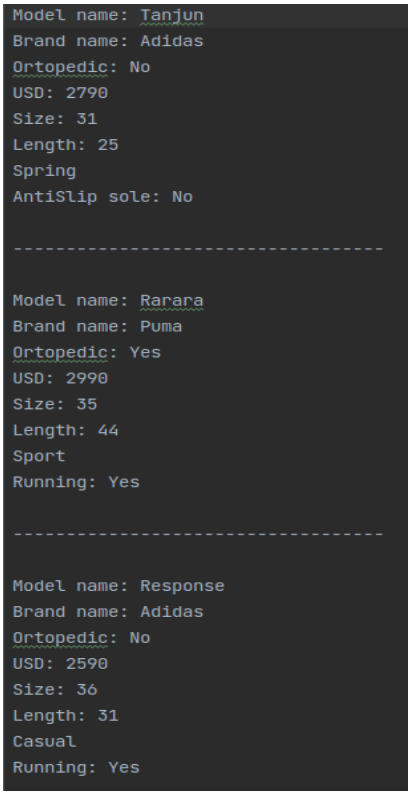


Рисунок 2 – Приклад вихідного файлу

4. Опис складу технічних та програмних засобів

4.1 Функціональне призначення

Програма виводить меню можливих дій з колекцією, та в залежності від отриманих від користувача даних виконує методи із загального та індивідуально завдань.

4.2 Опис логічної структури програми

Головна функція(`main()`). Створює список, та викликає функцію Run(), яка у свою чергу викликає функції `ReadFromFileSneakers(list);`

`ReadFromFileBoots(list);` що записують дані з файлу у контейнер.

. Функція Run() містить у собі оператор try catch, у якому викликається Menu(). Там від користувача запрошується номер дії. І в залежності від отриманого результату функція `CallingSelectedFunction(action, list);`викликає необхідну функцію.

Метод демонстрації вмісту контейнеру (`showAll()`). Виводить вміст усього контейнеру на екран.(Реалізація методу знаходиться в додатку Ж)

Метод запису данних до файлу (`SaveInFile()`). Записує данні з контейнеру до файлу.

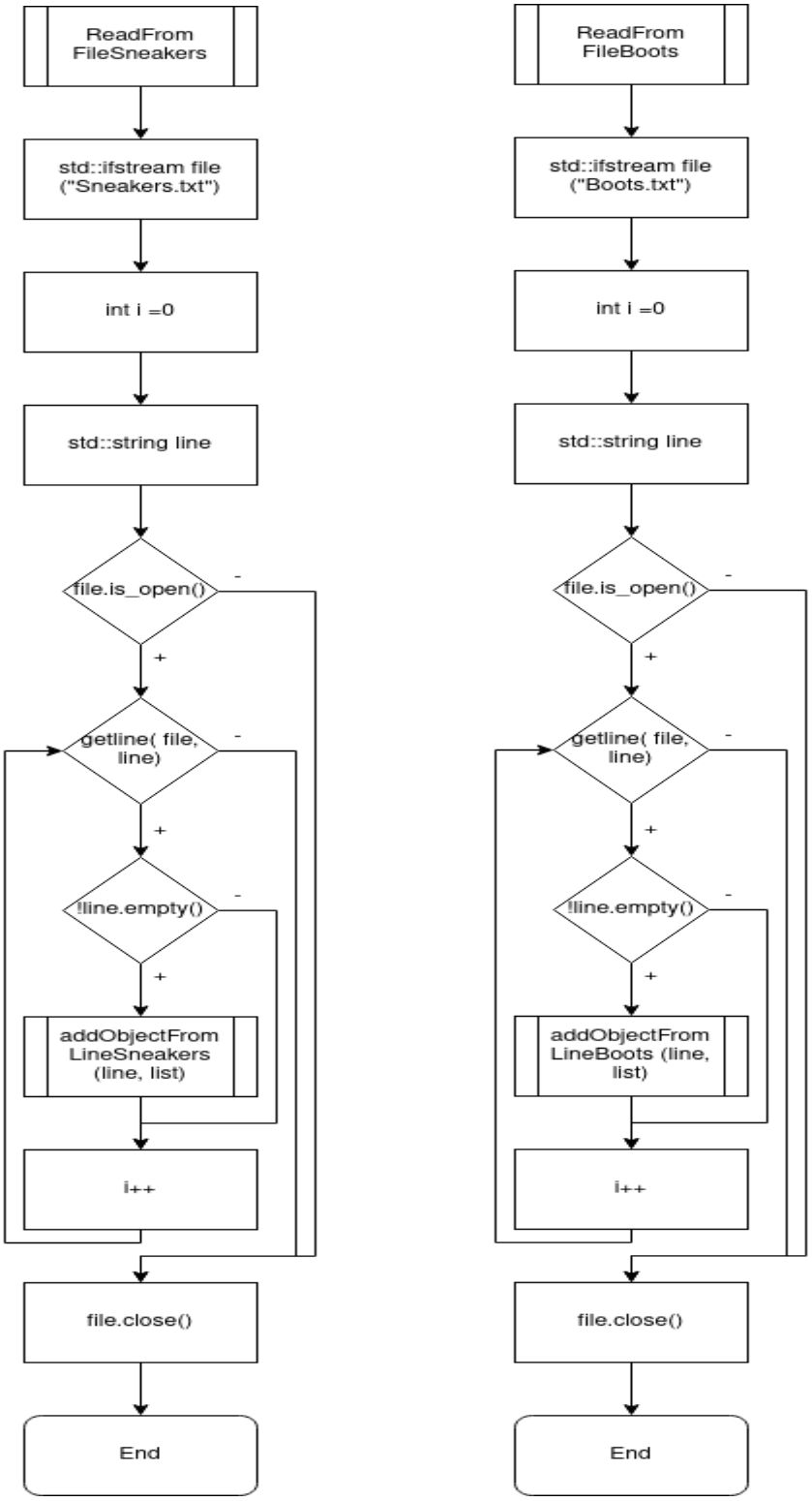
Метод пошуку чобіт з розміром бішьше 39 (`More39()`). (Реалізація методу знаходиться в додатку Б).

Метод пошуку найдешевших бігових кросівок (`CheapRunningShoes()`). (Реалізація методу знаходиться в додатку Г).

Метод пошуку ортопедичного взуття Найк і Пума (`NikePuma()`). (Реалізація методу знаходиться в додатку В).

Метод пошуку по трьом критеріям (`SaveInFile()`). (Реалізація методу знаходиться в додатку З).

Метод зчитування данних з файлу (`ReadFromFileSneakers()` і `ReadFromFileBoots()`). Зчитує данні з файлу та записує їх у контейнер. Схема алгоритму методу подана на рисунку 3.



а б

Рисунок 3 – Схеми алгоритмів методів (а - `ReadFromFileSneakers`, б - `ReadFromFileBoot`)

Метод сортування вмісту контейнера (`Sort(int criteria)`). Сортує данні відповідно критерію ціни. Якщо флаг дорівнює 1, то сортується від меньшого до більшого, якщо дорівнює 0, то навпаки.Схема алгоритму методу подана на рисунку 5. (Реалізація методу знаходиться в додатку А)

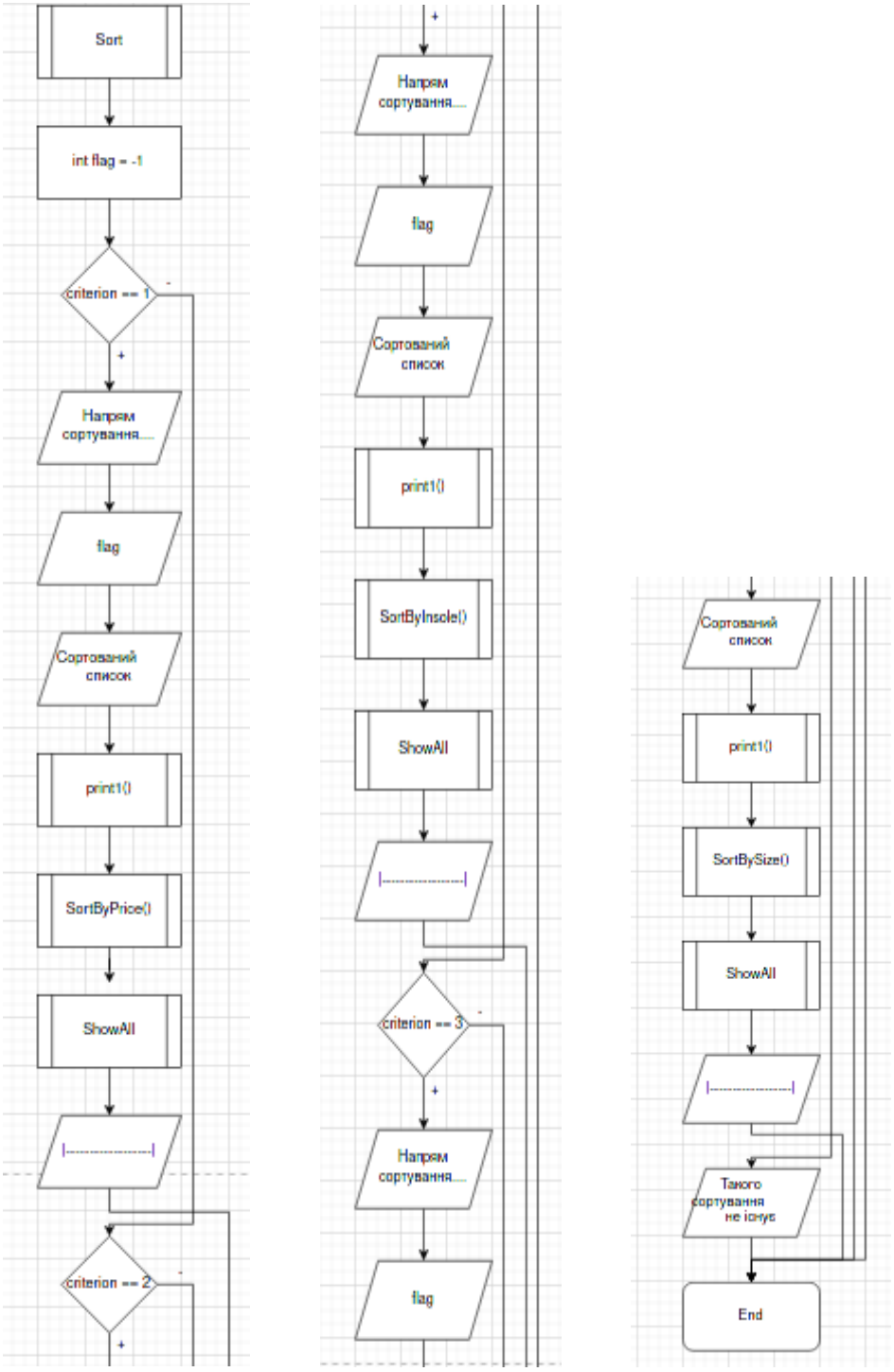


Рисунок 5 — Схема алгоритму методу `Sort`

Метод додання взуття до контейнера (`addShoes(Shoes *shoes)`). Додає взуття до контейнеру, використовуючи вже існуючий об’єкт. (Реалізація методу знаходиться в додатку Е).

Метод видалення ланки за індексом (`deleteElement(int index)`). Видаляє ланку з контейнеру за індексом. Схема алгоритму методу подана на рисунку

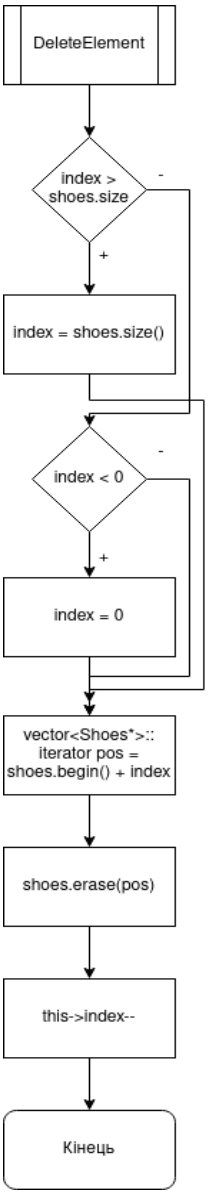


Рисунок 6 — Схема алгоритму методу `deleteElement`

Метод очищення всього списку (`deleteAll()`). Очищає інформацію з усього списку. Схема алгоритму методу подана на рисунку 7.

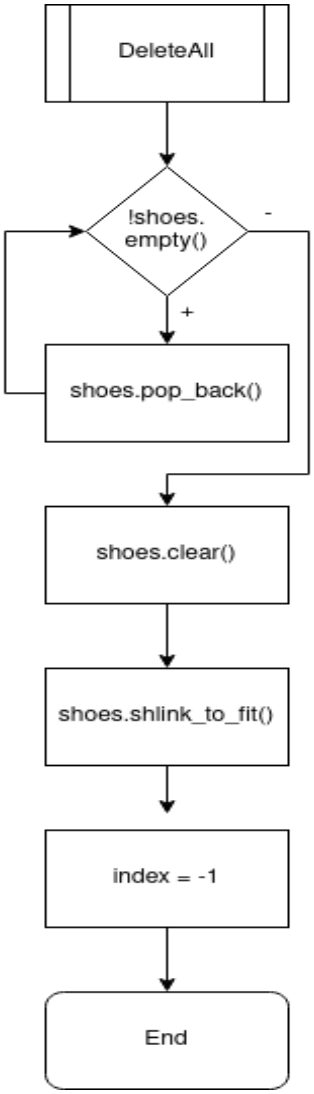


Рисунок 7 — Схема алгоритму методу `deleteAll`

4.3 Структура проекту

```
├── RGZ
    ├── doc
        └── Звіт
    ├── cmake-build-debug
        ├── Sneakers.txt
        ├── Boots.txt
        └── result.txt
    ├── CMakelist.txt
```

```
├── Container.cpp
├── Container.h
├── Controller.cpp
├── Controller.h
├── main.cpp
├── Menu.cpp
├── Menu.h
├── Shoes.cpp
└── Shoes.h
```

4.4 Варіанти використання

Для демонстрації результатів використовується IDE Clion. Нижче наводиться послідовність дій запуску програми.

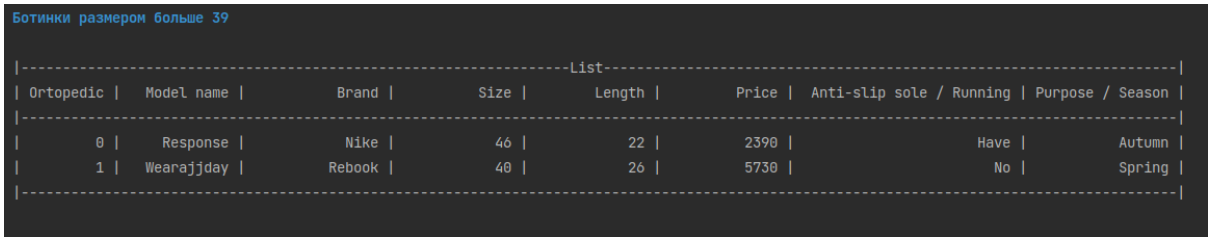
Крок 1 (рисунок 8). Виконаємо методи пошуку.

```
Ортопедическая обувь брендов Найк и Пума

|-------------------------------------------------------------List-------------------------------------------------------------|
| Ortopedic |   Model name |          Brand |          Size |        Length |         Price |  Anti-slip sole / Running | Purpose / Season |
|-------------------------------------------------------------------------------------------------------------------------------|
|         1 |       Rarara |           Puma |            37 |            24 |          1990 |                      Have |           Winter |
|         1 |       Rarara |           Puma |            35 |            44 |          2990 |                   Running |           Sport  |
|-------------------------------------------------------------------------------------------------------------------------------|
```

а

```
Самые дешёвые беговые кроссовки

|-------------------------------------------------------------List-------------------------------------------------------------|
| Ortopedic |   Model name |          Brand |          Size |        Length |         Price |  Anti-slip sole / Running | Purpose / Season |
|-------------------------------------------------------------------------------------------------------------------------------|
|         0 |       Tanjun |           Nike |            39 |            28 |          1690 |                   Running |           Undef  |
|-------------------------------------------------------------------------------------------------------------------------------|
```

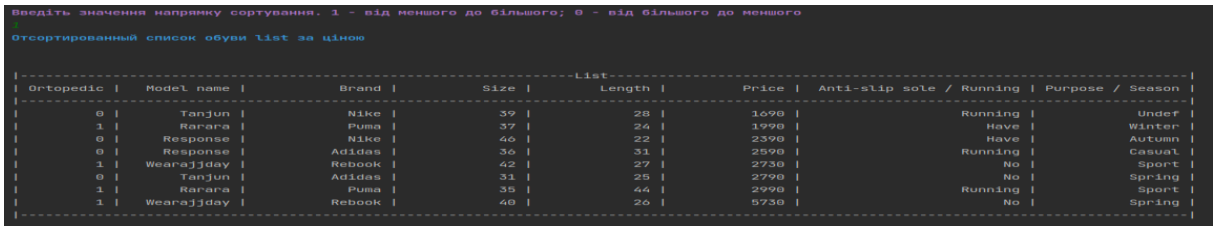
б



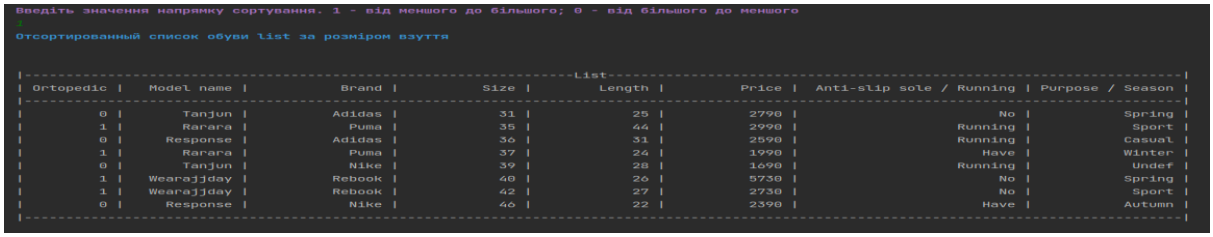
в

Рисунок 8 — результати роботи методів пошуку(а - пошук ортопедичного взуття брендів найк і пума, б - пошук найдешевших бігових кросівок, в - пошук чоботів з розміром більше 39)

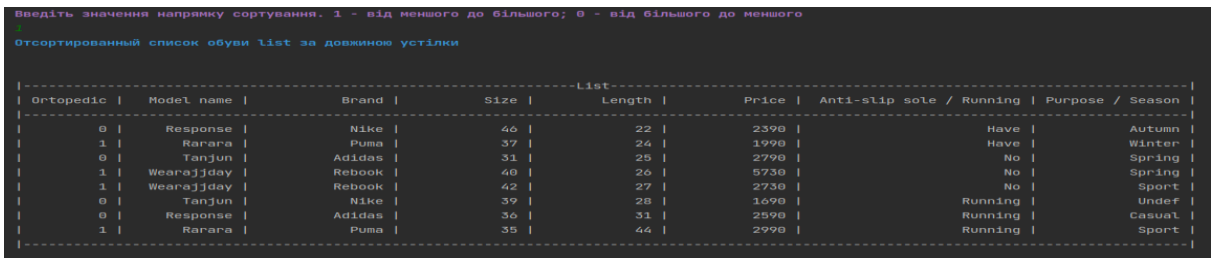
Крок 2 (рисунок 9). Виконаємо метод сортування.



а



б



в

Рисунок 9 — результат роботи методу сортування(а - за ціною, б - за розміром взуття (устілки), в - за довжиною устілки)

Висновки

Виконуючи розрахункове завдання я закріпила отримані знання з дисципліни «Програмування» та отримала практичні навички шляхом виконання типового комплексного завдання.

Додаток А. Реалізація методу Sort(int criterion)

```
void Controller::Sort(int criterion) {

   int flag = -1;

   if(criterion == 1){

       cout << "\33[1:35mВведіть значення напрямку сортування. 1 - від меншого до більшого; 0
- від більшого до меншого\33[0m";

       cin >> flag;

       cout << endl;

       cout << "\33[1:34mОтсортированный список обуви list за ціною\33[0m\n";

       print1();

       SortByPrice(flag);

       ShowAll();

       cout << "|-----------------------------------------------------------------------------
-------------------------------------------------------------|"<<endl;

       cout << endl;

   }else if(criterion == 2){

       cout << "\33[1:35mВведіть значення напрямку сортування. 1 - від меншого до більшого; 0
- від більшого до меншого\33[0m";

       cin >> flag;

       cout << endl;

       cout << "\33[1:34mОтсортированный список обуви list за довжиною устілки\33[0m\n";

       print1();

       SortByInsole(true);

       ShowAll();

       cout << "|-----------------------------------------------------------------------------
-------------------------------------------------------------|"<<endl;

       cout << endl;

   }else if(criterion == 3){

       cout << "\33[1:35mВведіть значення напрямку сортування. 1 - від меншого до більшого; 0
- від більшого до меншого\33[0m";

       cin >> flag;

       cout << endl;

       cout << "\33[1:34mОтсортированный список обуви list за розміром взуття\33[0m\n";
```

```
        print1();

        SortBySize(true);

        ShowAll();

        cout << "|-------------------------------------------------------------------------------
-------------------------------------------------------------|"<<endl;

        cout << endl;

    } else cout << "\33[1:31mТакого сортування не існує\33[0m";

}
```

Додаток Б. Реалізація методу More39()

```
vector<Boots*> Controller::More39() {

    vector<Boots*> res;

bool flag = true;

auto temp1 = shoes.getShoes();

auto iter = temp1.begin();

while (true) {

    iter = std::find_if(iter, temp1.end(), predG);

    if(*iter == NULL){

        break;

    }

    flag = false;

    res.push_back((Boots*)*iter);

    ((Boots*)*iter)->Print();

    cout << endl;

    iter++;

}

if (flag){

    cout << "ERROR: Nothing found" << endl;

}

return res;

}
```

Додаток В. Реалізація методу NikeRuma()

```
vector<Shoes*>  Controller::NikePuma() {
   vector<Shoes*> temp;
bool flag = true;
auto temp1 = shoes.getShoes();
auto iter = temp1.begin();
while (true) {
   iter = std::find_if(iter, temp1.end(), predM);
   if(*iter == NULL){
       break;
   }
   flag = false;
   ((Sneakers*)*iter)->Print();
   cout << endl;
   temp.push_back((Sneakers*)*iter);
   iter++;
}
if (flag){
   cout << "ERROR: Nothing found" << endl;
}
return temp;
}
```

Додаток Г. Реалізація методу CheapRunningShoes()

```
vector<Sneakers*>  Controller::CheapRunningShoes(){
   vector<Sneakers *> res;
bool flag = true;
auto temp1 = shoes.getShoes();
auto iter = temp1.begin();
while (true) {
```

```
    iter = std::find_if(iter, temp1.end(), predS);
    if (*iter == NULL) {
        cout << "ERROR: Nothing found" << endl;
    }
    flag = false;
    ((Sneakers *) *iter)->Print();
    cout << endl;
    res.push_back((Sneakers *) *iter);
    iter++;
    //}
    if (flag) {
        cout << "ERROR: Nothing found" << endl;
    }
    return res;
}
}
```

Додаток Е. Реалізація методу addShoes

```
void Controller::addShoes(Shoes *shoes) {
    index += 1;
    if (index >= 255) {
        return;
    }
    this->shoes.push_back(shoes->clone());
}
```

Додаток Ж. Реалізація методу ShowAll()

```
void Controller::ShowAll(){
    for (int i = 0; i <= this->index; ++i) {
        shoes[i]->Print();
        cout << endl;
```

```
    }
}
```

Додаток 3. Реалізація методу SaveInFile

```
void Controller::SaveInFile() {
    auto temp = shoes.getShoes();
FILE* myfile = fopen("result.txt", "w");
for (int i = 0; i < temp.size(); ++i) {
    temp[i]->SaveInFile(myfile);
}
fclose(myfile);
}
```