Федеральное государственное автономное образовательное учреждение высшего образования «**Институт промышленного менеджмента, экономики и торговли»**

Факультет \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Курсовая работа по программированию**

Преподаватель: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2022г

1. Введение

В первом семестре в дисциплине «программирование» были изучены основы языка программирования С++ (синтаксис, инициализация, циклы и условия, функции, работа с указателями и ссылками(здесь можешь чтот добавить, я не знаю вашей программы)). Данная курсовая работа посвящена последним в списке перечисленных – указатели и ссылки.

Указатель – переменная, значением которой является адрес на ячейку памяти. Указатели часто используются в языках С и С++ по трем основым причинам:

* выделить новые объеты в куче
* передача функции другим функциям
* итерация элементов в массивах и других структурах данных

Благодаря грамотной работе с указателями время исполнения программы значительно уменьшается. Также в С++ существуют ссылки, которые тоже ссылаются на ячейку в памяти. Основное отличие от указателей – после инициализации переменной ссылку нельзя перенаправить на другой объект или присвоить значение *null.* Ссылки делятся на два типа:

* *lvalue* – ссылаются на именнованную переменную
* *rvalue* – ссылаются на временный объект

Ссылки объявляются таким образом:

*[описатели класса хранения] [cv-qualifiers] описатель типа [ms-modifier] декларатор [=выражение];*

Также существует упрощенный способ объявления ссылки, который используется гораздо чаще:

*[описатели класса хранения] [cv-qualifiers] type-specifiers [& или &&] [cv-qualifiers] identifier [=expression];*

Ссылка объявляется по следующей последовательности:

Спецификаторы объявления:

* Необязательный спецификатор хранения
* Необязательный *const* или *volatile* квалификаторы
* Спецификатор типа: имя типа

2. Декларатор:

* Оператор & или &&
* Необязательный *const* или *volatile* квалификаторы
* Идентификатор

3. Необязательный инициализатор

Более сложные формы декларатора для указателя на массивы и функции применяются к ссылкам на те же массивы и функции.

2. Постановка задачи

Для грамотного выполнения работы, проект был разделен на два этапа. В каждом из которых были поставлены определенные задачи:

Этап #1

1. Создание структуры *BOOK,* описывающую книгу(автор, заглавие, год издания, цена, категорию)
2. Создание дополнительных функций для работы с объектом структуры. Вывод полей объекта в консоль через printf, ввод для формирования полей структуры через scanf.
3. Реализация проверки на «дурака», чтобы пользователь не столкнулся с ошибками исполнения программы при неверно введенном значении(проверка года издания [0;2022], цена [0; +*inf*), категорию пользователь может выбрать только из предложенных {«manga», «roman», «classic», «fantasy»})

Этап #2

1. Создание динамического массива типа *BOOK* – картотека + динамическое добавление объекта в конец массива.
2. Добавление дополнительных функций для работы с картотекой(создание «меню», где будет выводится перечень возможных функций(распечатать содержимое картотеки, добавить новую книгу, удалить существующую(ие) книге, записать содержимое картотеки в файл, считать содержимое из файла в картотеку), выход из «меню»)
3. Создание сортировки картотеки по одному из полей объекта структуры и вывод отсортированной картотеки.