|  |  |
| --- | --- |
| Институт (факультет) | Институт информационных технологий |
| Кафедра | Кафедра математического и программного обеспечения ЭВМ |

# РАЗРАБОТКА АБСТРАКТНЫХ ТИПОВ ДАННЫХ

|  |  |
| --- | --- |
| **Дисциплина:** | ООП |
| **Темы:** | Инкапсуляция; классы; поля и компонентные функции; конструкторы и деструкторы класса |

**Среда разработки:** Microsoft Visual Studio

**Язык программирования:** C++

**Тип проекта:** Консольное приложение

**Задание на лабораторную работу №2**

**ТРЕБОВАНИЯ К РАЗРАБОТКЕ**

1. Каждый класс должен быть оформлен в отдельных файлах: заголовочный (.h) и файл с кодом (.cpp).
2. Запрещается использовать обработку исключительных ситуаций и генерировать исключения.
3. Придерживайтесь принципа DRY (Don’t repeat yourself).
4. Обязательно наличие комментариев.

**ЦЕЛЬ РАБОТЫ**

Изучить способы определения классов, правила доступа к элементам; приобрести практические навыки работы с объектами класса; изучить принципы и механизмы создания абстрактных типов данных.

**ЗАДАНИЯ**

1. *часов*
2. Разработайте АТД – класс, обеспечивающий хранение объектов класса из ЛР1, согласно варианту задания (см. раздел прил. 1).
3. Интерфейс класса может содержать конструкторы и должен содержать деструктор.
4. Для описания элементов контейнера использовать вложенные структуры.
5. Интерфейс класса должен содержать функции:
   1. Добавления;
   2. Удаления;
   3. Вывода содержимого;
   4. Обработки (сортировка, поиск по условию);
   5. Получения данных контейнера в виде массива.

**ПРИМЕЧАНИЯ**

* У класса-контейнера не должно быть прямого доступа к полям класса предметной области (ЛР1).

**КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Каковы особенности синтаксиса и семантики конструкторов и деструкторов?

2. В чем заключаются особенности применения конструкторов по умолчанию?

3. В каких случаях вызывается копирующий конструктор?

4. При каких условиях конструктор может использоваться для преобразования типа?

5. В каких ситуациях необходимы инициализаторы конструктора, синтаксис и семантика их использования?

**Приложение 1**

**Варианты:**

1. Статический вектор\*

2. Динамический вектор\*

3. Статическая матрица\*\*

4. Динамическая матрица\*\*

5. Линейный односвязный список\*

6. Линейный двусвязный список\*

7. Стек\*

8. Очередь\*

9. Закольцованный список\*

10. Двоичное дерево\*

\* - обязательно с использованием структуры и указателей.

\*\* - разрешается использовать двумерный динамический массив.