В ходе выполнения домашнего задания я углубил свои знания языка программирования C++.

- Изучил порядок создания и разрушения объектов;

- Узнал судьбу копий объектов при передаче их параметрами по значению;

- Изучил порядок создания и разрушения составных объектов и их предков или полей;

- Узнал как размещать статические поля класса;

- Изучил правила работы с динамическими данными;

- Научился создавать и использовать шаблоны.

Ответы на вопросы:

1. Передача по значению, присваивание;
2. Сначала создаются все «родители» и свойства класса, затем уже сам составной классе. Свойства создаются по очереди, в зависимости от порядка объявления в составном объекте;
3. Интерфейс IDisposable в C# объявляет один единственный метод Dispose, в котором должно происходить освобождение неуправляемых ресурсов (в основном это различные сетевые подключения, работа с файлами);
4. Dispose будет вызван, но объект не будет уничтожен;
5. Для этого нужно явно указать наследование.

Пример: Child(int n):Test(n)

1. Для этого после конструктора нужно написать “: base(args[])”.

Пример:

public Child(int n) : base(n)

1. Для этого нужно явно указать вызов необходимого конструктора.

Пример:

Test m\_objTest;

AggregateT() : m\_objTest(9)

{

cout << "Created AggregateT" << endl;

}

Здесь мы явно вызываем конструктор члена класса с параметром.

1. В .Net, как и в C++, ссылка указывает на какой-либо объект, хранящийся в памяти. Но в .Net (в том числе unsafe) мы не можем создать ссылку на ячейку памяти (только указатель) (т.е. мы не можем сделать следующее: int a = 8; int& re = a). Таким образом, в .Net ссылка может указывать только на конкретный объект, в C++ - на любую ячейку памяти.
2. Для этого перед объявлением нужно прописать

«template <typename T>». Далее тип T можно будет использовать в контексте класса. В качестве T можно передать любой класс.