№ 3 Проектирование типов. Классы.

Задание

1) Определить класс, указанный в варианте, содержащий:

- Не менее трех конструкторов (с параметрами и без, а также с параметрами по умолчанию);
- **статический конструктор** (конструктор типа);
- определите **закрытый** конструктор; предложите варианты его вызова;
- поле - **только для чтения** (например, для каждого экземпляра сделайте поле только для чтения ID - равно некоторому уникальному номеру (хэшу) вычисляемому автоматически на основе инициализаторов объекта);
- поле- константу;
- **свойства** (get, set) – для всех поле класса (поля класса должны быть закрытыми); Для одного из свойств ограните доступ по set
- в одном из методов класса для работы с аргументами используйте **ref - и out-параметры**.
- создайте в классе **статическое поле**, хранящее количество созданных объектов (инкрементируется в конструкторе) и **статический метод** вывода информации о классе.
- сделайте касс **partial**
- переопределяете методы класса Object: Equals, для сравнения объектов, GetHashCode; для алгоритма вычисления хэша руководствуйтесь стандартными рекомендациями, ToString – вывода строки –информации об объекте.

2) Создайте несколько объектов вашего типа. Выполните вызов конструкторов, свойств, методов, сравнение объекты, проверьте тип созданного объекта и т.п.

3) Создайте массив объектов вашего типа. И выполните задание, выделенное курсивом в таблице.
4) Создайте и выведите анонимный тип (по образцу вашего класса).
5) Ответьте на вопросы, приведенные ниже

Вариант 10	Построить класс Булев вектор (**BoolVector**). Реализовать методы для выполнения поразрядных конъюнкции, дизъюнкции и отрицания векторов, а также подсчета числа единиц и нулей в векторе. *Создать массив объектов. Вывести:* *a) вектора с заданным числом единиц/нулей;* *b) определить и вывести равные вектора.*

№ 4 Перегрузка операций, методы расширения и вложенные типы

Задание

1) Создать заданный в варианте класс. Определить в классе необходимые методы, конструкторы, индексаторы и заданные перегруженные операции. Написать программу тестирования, в которой проверяется использование перегруженных операций.
2) Добавьте в свой класс вложенный объект Owner, который содержит Id, имя и организацию создателя. Проинициализируйте его
3) Добавьте в свой класс вложенный класс Date (дата создания). Проинициализируйте

4) Создайте статический класс MathOperation, содержащий 3 метода для работы с вашим классом (по варианту п.1): поиск максимального, минимального, подсчет количества элементов.
5) Добавьте к классу MathOperation методы расширения для типа string и вашего типа из задания№1. См. задание по вариантам.

Вариант 10	Класс - список **List**. Дополнительно перегрузить следующие операции: + - добавить элемент в начало (item+list); -- - удалить первый элемент из списка (--list); != - проверка на неравенство; * - объединение двух списков. Методы расширения: 1) Подсчет количества слов с заглавной буквы 2) Проверка на повторяющиеся элементы в списке

№ 5 Наследование, полиморфизм, абстрактные классы и интерфейсы

Задание

1) Определить иерархию и композицию классов (в соответствии с вариантом), реализовать классы. Если необходимо расширьте по своему усмотрению иерархию для выполнения всех пунктов л.р.

   Каждый класс должен иметь отражающее смысл название и информативный состав. При кодировании должны быть использованы соглашения об оформлении кода code convention.

   В одном из классов переопределите все методы, унаследованные от Object.
2) В проекте должны быть **интерфейсы и абстрактный класс**(ы). Использовать виртуальные методы и переопределение.
3) Сделайте один из классов герметизированным **(бесплодным)**.
4) Добавьте в интерфейсы или интерфейс + абстрактный класс **одноименные методы**. Дайте в наследуемом классе им разную реализацию и вызовите эти методы.

5) Написать демонстрационную программу, в которой создаются объекты различных классов. Поработать с объектами **через ссылки на абстрактные классы и интерфейсы**. В этом случае для идентификации типов объектов использовать **операторы is или as.**

6) Во всех классах (иерархии) переопределить метод ToString(), который выводит информацию о типе объекта и его текущих значениях. Создайте дополнительный класс Printer с полиморфным методом iAmPrinting(SomeAbstractClassorInterface someobj). Формальным параметром метода должна быть ссылка на абстрактный класс или наиболее общий интерфейс в вашей иерархии классов. В методе iAmPrinting определите тип объекта и вызовите ToString(). В демонстрационной программе создайте массив, содержащий ссылки на разнотипные объекты ваших классов по иерархии, а также объект класса Printer и последовательно вызовите его метод iAmPrinting со всеми ссылками в качестве аргументов.

Вариант 10	Квитанция, Накладная, Документ, Чек, Дата, Организация.

№ 6 Структуры, перечисления, классы контейнеры и контроллеры

Задание

1) К предыдущей лабораторной работе (л.р. 5) добавьте к существующим классам **перечисление** и **структуру**.

2) Один из классов сделайте **partial** и разместите его в разных файлах.

3) Определить **класс-Контейнер** (указан в вариантах жирным шрифтом) для хранения разных типов объектов (в пределах иерархии) в виде списка или массива (использовать абстрактный тип данных). Класс-контейнер должен содержать методы get и set для управления списком/массивом, методы для добавления и удаления объектов в список/массив, метод для вывода списка на консоль.

4) Определить управляющий **класс-Контроллер**, который управляет объектом- Контейнером и реализовать в нем запросы по варианту. При необходимости используйте стандартные интерфейсы (IComparable, ICloneable,….)

Вариант 10	Собрать **Бухгалтерию**. *Найти суммарную стоимость продукции заданного наименования по всем накладным*, *количество чеков.* *Вывести две документы за указанный период времени.*

№ 7 Обработка исключений

Задание

Дополнить предыдущую лабораторную работу № 6.

Создать иерархию классов исключений (собственных) – 3 типа и более. Сделать наследование пользовательских типов исключений от стандартных классов .Net (например Exception).

Сгенерировать и обработать как минимум пять различных исключительных ситуаций на основе своих и стандартных исключений. Например, не позволять при инициализации объектов передавать неверные данные, обрабатывать ошибки при работе с памятью и ошибки работы с файлами, деление на ноль, неверный индекс, нулевой указатель и т. д.

В конце поставить универсальный обработчик catch.

Обработку исключений вынести в main. При обработке выводить специфическую информацию о месте, диагностику и причине исключения. Последним должен быть блок, который отлавливает все исключения (finally).
Добавьте код в одной из функций макрос Assert. Объясните что он проверяет, как будет выполняться программа в случае не выполнения условия.
Объясните назначение Assert.
Пример:

```
int[] aa= null;
            Debug.Assert(aa !=null, "Values array cannot be null");
```

Не забудьте подключить `using  System.Diagnostics;`

№ 8 Обобщения

Задание

1. Создайте **обобщенный интерфейс** с операциями добавить, удалить, просмотреть.

2. Возьмите за основу лабораторную № 4 «Перегрузка операций» и сделайте из нее **обобщенный тип (класс) CollectionType<T>**, в который вложите обобщённую коллекцию. Наследуйте в обобщенном классе интерфейс из п.1. Реализуйте необходимые методы. Добавьте **обработку исключений** с finally. Наложите какое-либо **ограничение** на обобщение.

3. Проверьте использование обобщения для стандартных типов данных (в качестве стандартных типов использовать целые, вещественные и т.д.). Определить пользовательский класс, который будет использоваться в качестве параметра обобщения. Для пользовательского типа взять класс из лабораторной №5 «Наследование».

Дополнительно:

Добавьте методы сохранения объекта (объектов) обобщённого типа CollectionType<T> в файл и чтения из него.

№9 Делегаты, события и лямбда выражения

Задание

1. Используя делегаты (множественные) и события промоделируйте ситуации, приведенные в таблице ниже. Можете добавить новые типы (классы), если существующих недостаточно. При реализации методов везде где возможно использовать лямбда-выражения.

2. Создайте пять методов пользовательской обработки строки (например, удаление знаков препинания, добавление символов, замена на заглавные, удаление лишних пробелов и т.п.). Используя стандартные типы делегатов (Action, Func) организуйте алгоритм последовательной обработки строки написанными вами методами.

9, 10	Создать класс *Программист* с событиями *Удалить* и *Мутировать*. В main создать некоторое количество объектов (списков). Подпишите объекты на события произвольным образом. Реакция на события может быть следующая: удаление первого/последнего элемента списка, случайное перемещение строк и т.п. Проверить результаты изменения состояния объектов после наступления событий, возможно не однократном

№ 10 Коллекции

Задание

1. Создать *необобщенную* коллекцию **ArrayList.**
    a. Заполните ее 5-ю случайными целыми числами
    b. Добавьте к ней строку
    c. Добавьте объект типа Student
    d. Удалите заданный элемент
    e. Выведите количество элементов и коллекцию на консоль.
    f. Выполните поиск в коллекции значения
2. Создать **обобщенную коллекцию** в соответствии с вариантом задания и заполнить ее данными, тип которых определяется вариантом задания (колонка – *первый тип)*.
    a. Вывести коллекцию на консоль
    b. Удалите из коллекции n последовательных элементов
    c. Добавьте другие элементы (используйте все возможные методы добавления для вашего типа коллекции).
    d. Создайте *вторую коллекцию* (см. таблицу) и заполните ее данными из первой коллекции.
    e. Выведите вторую коллекцию на консоль. В случае не совпадения количества параметров (например, *LinkedList<T>* и *Dictionary<Tkey, TValue>*), при нехватке - генерируйте ключи, в случае избыточности – оставляйте *TValue*.
    f. Найдите во второй коллекции заданное значение.

№	Первая коллекция	Первый тип	Вторая коллекция
1	Stack<T>	char	List<T>
2	Queue<T>	int	Dictionary<Tkey, TValue>
3	HashSet<T>	long	LinkedList<T>
4	List<T>	float	Stack<T>
5	Dictionary<Tkey, TValue>	double	Queue<T>
6	LinkedList<T>	char	HashSet<T>
7	SortedDictionary<TKey, TValue>	string	List<T>
8	SortedList<TKey, TValue>	long	Stack<T>
9	SortedSet<T>	float	Queue<T>
10	Dictionary<Tkey, TValue>	int	HashSet<T>
11	SortedList<TKey, TValue>	char	List<T>
12	Stack<T>	double	LinkedList<T>
13	Queue<T>	int	SortedDictionary<TKey, TValue>
14	HashSet<T>	long	SortedList<TKey, TValue>
15	List<T>	long	SortedSet<T>
16	Dictionary<Tkey, TValue>	string	Stack<T>
17	LinkedList<T>	double	SortedList<TKey, TValue>

18	SortedDictionary<TKey, TValue>	char	LinkedList<T>
19	SortedList<TKey, TValue>	int	Stack<T>
20	Stack<T>	int	SortedList<TKey, TValue>
21	SortedSet<T>	int	Dictionary<Tkey, TValue>
22	Dictionary<Tkey, TValue>	long	List<T>
23	SortedList<TKey, TValue>	float	Stack<T>
24	List<T>	char	Dictionary<Tkey, TValue>
25	Dictionary<Tkey, TValue>	bool	List<T>
26	LinkedList<T>	bool	Dictionary<Tkey, TValue>

3. Повторите задание п.2 для **пользовательского типа данных** (в качестве типа T возьмите любой свой класс из лабораторной *№5 (Наследование*….). Не забывайте о необходимости реализации интерфейсов (IComparable, ICompare,….). При выводе коллекции используйте цикл foreach.
4. Создайте объект *наблюдаемой коллекции* **ObservableCollection<T>**. Создайте произвольный метод и зарегистрируйте его на событие CollectionChange. Напишите демонстрацию с добавлением и удалением элементов. В качестве типа *T* используйте свой класс из лабораторной *№5 Наследование*….

№ 11 LINQ to Object

Задание

1. Задайте массив типа string, содержащий 12 месяцев (June, July, May, December, January ….). Используя LINQ to Object напишите запрос выбирающий последовательность месяцев с длиной строки равной n, запрос возвращающий только летние и зимние месяцы, запрос вывода месяцев в алфавитном порядке, запрос считающий месяцы содержащие букву «u» и длиной имени не менее 4-х..

2. Создайте коллекцию List<T> и параметризируйте ее типом (классом) из лабораторной №3 (при необходимости реализуйте нужные интерфейсы).

3. На основе LINQ сформируйте следующие запросы по вариантам. При необходимости добавьте в класс T (тип параметра) свойства.

4. Придумайте и напишите свой собственный запрос, в котором было бы не менее 5 операторов из разных категорий: условия, проекций, упорядочивания, группировки, агрегирования, кванторов и разиения.

5. Придумайте запрос с оператором Join

Вариант 10	*вектора с заданным числом единиц/нулей;* *определить и вывести равные вектора в коллекции* *максимальный вектор* *первый вектор с n единицами* *упорядоченный вектор по числу единиц*

№ 12 Рефлексия

Задание

1. Для изучения .NET Reflection API допишите класс *Рефлектор*, который будет содержать методы выполняющие следующие действия:

    a. выводит всё содержимое класса в текстовый файл (принимает в качестве параметра имя класса);

    b. извлекает все общедоступные публичные методы класса (принимает в качестве параметра имя класса);

    c. получает информацию о полях и свойствах класса;

    d. получает все реализованные классом интерфейсы;

    e. выводит по имени класса имена методов, которые содержат заданный (пользователем) тип параметра (имя класса передается в качестве аргумента);

    f. вызывает некоторый метод класса, при этом значения для его параметров необходимо прочитать из текстового файла (имя класса и имя метода передаются в качестве аргументов).

Продемонстрируйте работу «Рефлектора» для исследования типов на созданных вами классах не менее двух (предыдущие лабораторные работы) и стандартных классах .Net.

№ 13 Работа с потоковыми классами и файловой системой

Задание

Каждый класс в данном проекте должен начинаться (Префикс) с ваших инициалов ФИО (AVF, JK,….). Предусмотреть обработку ошибок.

4. Создать класс XXXLog. Он должен отвечать за работу с текстовым файлом xxxlogfile.txt. в который записываются все действия пользователя и соответственно методами записи в текстовый файл, чтения, поиска нужной информации.
    a. Используя данный класс выполните запись всех последующих действиях пользователя с указанием действия, детальной информации (имя файла, путь) и времени (дата/время)
5. Создать класс XXXDiskInfo с методами для вывода информации о
    a. свободном месте на диске
    b. Файловой системе
    c. Для каждого существующего диска - имя, объем, доступный объем, метка тома.
    d. Продемонстрируйте работу класса
6. Создать класс XXXFileInfo с методами для вывода информации о конкретном файле
    a. Полный путь
    b. Размер, расширение, имя
    c. Время создания
    d. Продемонстрируйте работу класса
7. Создать класс XXXDirInfo с методами для вывода информации о конкретном директории
    a. Количестве файлов
    b. Время создания
    c. Количестве поддиректориев
    d. Список родительских директориев
    e. Продемонстрируйте работу класса

8. Создать класс XXXFileManager. Набор методов определите самостоятельно. С его помощью выполнить следующие действия:
    a. Прочитать список файлов и папок заданного диска. Создать директорий XXXInspect, создать текстовый файл xxxdirinfo.txt и сохранить туда информацию. Создать копию файла и переименовать его. Удалить первоначальный файл.
    b. Создать еще один директорий XXXFiles. Скопировать в него все файлы с заданным расширением из заданного пользователем директория. Переместить XXXFiles в XXXInspect.
    c. Сделайте архив из файлов директория XXXFiles. Разархивируйте его в другой директорий.
9. Найдите и выведите сохраненную информацию в файле xxxlogfile.txt о действиях пользователя за определенный день/ диапазон времени/по ключевому слову. Посчитайте количество записей в нем. Удалите часть информации, оставьте только записи за текущий час.

№ 14 Сериализация

Задание

10. Из лабораторной №5 выберите класс с наследованием и/или композицией/агрегацией для сериализации. Выполните сериализацию/десериализацию объекта используя
    a. бинарный,
    b. SOAP,
    c. JSON,
    d. XML формат.

    * *Усложненное задание:*
    Для сериализации выберите класс-контейнер из лабораторной № 6.
    При записи в xml формате используйте некоторые свойства класса как атрибуты.

11. Создайте коллекцию (массив) объектов и выполните сериализацию/десериализацию.
    * *Усложненное задание:*
    Создайте клиент и сервер на синхронных сокетах.
    Нужно сериализованные данные(объект) отправить по сокету и десериализовать на стороне клиента.

12. Используя XPath напишите два селектора для вашего XML документа.
13. Используя Linq to XML (или Linq to JSON) создайте новый xml (json) - документ и напишите несколько запросов.

№ 15 Работа с потоками выполнения

Задание

1. Определите и выведите на консоль/в файл все запущенные процессы:id, имя, приоритет, время запуска, текущее состояние, сколько всего времени использовал процессор и т.д.
2. Исследуйте текущий домен вашего приложения: имя, детали конфигурации, все сборки, загруженные в домен. Создайте новый домен. Загрузите туда сборку. Выгрузите домен.
3. Создайте в отдельном потоке следующую задачу расчета (можно сделать sleep для задержки) и записи в файл и на консоль простых чисел от 1 до n (задает пользователь). Вызовите методы управления потоком (запуск, приостановка, возобновление и тд.) Во время выполнения выведите информацию о статусе потока, имени, приоритете, числовой идентификатор и т.д.
4. Создайте два потока. Первый выводит четные числа, второй нечетные до n и записывают их в общий файл и на консоль. Скорость расчета чисел у потоков – разная.
    a. Поменяйте приоритет одного из потоков.
    b. Используя средства синхронизации организуйте работу потоков, таким образом, чтобы
        i. выводились сначала четные, потом нечетные числа
        ii. последовательно выводились одно четное, другое нечетное.
5. Придумайте и реализуйте повторяющуюся задачу на основе класса Timer

Дополнительно (по желанию)

1. *На складе имеются товары (файл с записями). Создайте три потока - машины, каждая машина имеет свою скорость загрузки/разгрузки. Разгрузите склад. Обеспечьте последовательный доступ складу (только одна машина может загружаться единовременно)*
2. *Создайте пул ресурсов видеоканалов (класс) которых изначально меньше чем клиентов (класс), которые хотят ими воспользоваться. Каждый клиент получает доступ к каналу, причем пользоваться можно только одним каналом. Если все каналы заняты, то клиент ждет заданное время и по его истечении уходит не получив услуги. (используйте средства синхронизации - семафор)*

№ 16 Платформа параллельных вычислений

Задание

6. Используя TPL создайте длительную по времени задачу (на основе Task) на выбор:

   - ✓ поиск простых чисел (желательно взять «решето Эратосфена»),
   - ✓ перемножение матриц,
   - ✓ умножение вектора размера 100000 на число,
   - ✓ создание множества Мандельброта
   - ✓ или другой алгоритм.

   a. Выведите идентификатор текущей задачи, проверьте во время выполнения – завершена ли задача и выведите ее статус.
   b. Оцените производительность выполнения используя объект `Stopwatch` на нескольких прогонах.

   Дополнительно:
   Для сравнения реализуйте последовательный алгоритм.

7. Реализуйте второй вариант этой же задачи с токеном отмены CancellationToken и отмените задачу.
8. Создайте три задачи с возвратом результата и используйте их для выполнения четвертой задачи. Например, расчет по формуле.
9. Создайте задачу продолжения (continuation task) в двух вариантах:
   a. С ContinueWith - планировка на основе завершения множества предшествующих задач
   b. На основе объекта ожидания и методов GetAwaiter(),GetResult();
10. Используя Класс Parallel распараллельте вычисления циклов For(), ForEach(). Например, на выбор: обработку (преобразования)последовательности, генерация нескольких массивов по 1000000 элементов, быстрая сортировка последовательности, обработка текстов (удаление, замена). Оцените производительность по сравнению с обычными циклами
11. Используя Parallel.Invoke() распараллельте выполнение блока операторов.
12. Используя Класс BlockingCollection реализуйте следующую задачу:

    Есть 5 поставщиков бытовой техники, они завозят уникальные товары на склад (каждый по одному) и 10 покупателей – покупают все подряд, если товара нет - уходят. В вашей задаче: спрос превышает предложение. Изначально склад пустой. У каждого поставщика своя скорость завоза товара. Каждый раз при изменении состоянии склада выводите наименования товаров на складе.

13. Используя async и await организуйте асинхронное выполнение любого метода.