Для реализации элементарной симуляции потребуются следующие классы:

* Food – класс, содержащий информацию о еде (об энергии, хранящейся в ней)
* ElementaryEntity – класс, содержащий характеристик существа.
* Field – класс, агрегирующей в себе всю информацию о симуляции (ссылки на сущности, их расположение)

В качестве вспомогательного также вводится класс EmptyEntity, обозначающий отсутствие сущности в клетке, и абстрактный класс Entity, потомками которого являются все вышеперечисленные классы, кроме Field.

Класс Field содержит очередь из elementaryEntity и метод DoBeat(). Метод осуществляет действия одного такта для сущностей из очереди. Вычисляется длина очереди и сохраняется в переменную. После этого извлекается первый объект elementaryEntity, для которого вызывается метод BeatAction(), после чего этот объект помещается в конец очереди или удаляется, если существо погибло. Действия повторяются столько раз, какой была длина очереди в начале работы метода DoBeat(). Длина очереди фиксируется заранее из-за того, что метод BeatAction() может добавлять в очередь новые экземпляры класса elementaryEntity (при создании потомка существа). При этом длина очереди увеличивается, а обработка нового экземпляра, стоящего в конце очереди, должна осуществляться только на следующем такте.

**Класс field**

**Описание класса.**

Класс и его наследники предназначены для хранения всей информации о симуляционном поле (о расположении каждой сущности – класс entity – и о характеристиках каждой ячейки поля) и отображения информации на экран.

Класс является абстрактным, все методы объявлены виртуальными. Предполагается расширение функциональности данного класса в наследниках. В частности, наследники дополняют класс информацией о характеристиках ячеек поля, так как базовый класс хранит исключительно информацию о расположении сущностей, и наследники непосредственно определяют средства отображения (например, отображение с помощью экземпляра класса DataGridView).

Рекомендуется использовать данные матрицы EntityMatrix только для чтения, а её состояние менять через методы класса (для корректного отображения информации на экране).

При работе с координатами в данном классе, предполагается, что координаты – это индексы элемента в матрице EntityMatrix. Классы-наследники могут по-своему определять понятие координат и, соответственно, переопределять методы.

**Конструкторы класса:**

* **public field(int width, int height)** – аргументы задают размеры симуляционного поля.

**Свойства класса**

* **public entity[,] EntityMatrix { get; }** – служит для получения матрицы с расположением всех сущностей на симуляционном поле.

**Методы класса:**

* **public virtual void DoBeat()** – совершает тактовое действие для всех сущностей на поле (вызывает метод BeatAction() класса entity).
* **public virtual void MoveEntity(int oldX, int oldY, int newX, int newY)** – перемещается сущность с координат (oldX, oldY) на координаты (newX, newY). Аргументы метода являются индексами элемента матрицы EntityMatrix. По «старым» координатам создается пустая сущность – класс emptyEntity.
* **public virtual void ClearEntity(int x, int y)** – удаляет сущность по заданным координатам. Координатами являются индексы элемента в матрице EntityMatrix. Вместо удаленной сущности подставляется объект класса emptyEntity.
* **public virtual void AddEntity(entity newEntity)** – добавляет новую сущность на симуляционное поле. Старая сущность при этом удаляется. Координаты сущности (индексы элемента матрицы EntityMatrix) определяются по свойствам Х, Y класса entity.

Класс entity

**Описание класса:**

Абстрактный класс, служащий основой любой сущности для любой симуляции. Наследники определяют характеристики конкретных сущностей и их поведение (в первую очередь, метод BeatAction(), отвечающий за тактовое действие).

Экземпляры классов-наследников хранятся в матрице EntityMatrix класса field. Метод DoBeat() класса field вызывает метод BeatAction() класса entity. Тот, в свою очередь, изменяет данные field в соответствии с поведением сущности.

Под координатами сущности подразумеваются индексы элемента в матрице EntityMatrix класса field. Этот принцип может быть изменен наследниками (при соответствующем переопределении методов класса field).

**Поля класса:**

* **protected int x** – координата х сущности на симуляционном поле.
* **protected int y** – координата y сущности на симуляционном поле.
* **protected field entityBase** – симуляционное поле, которому принадлежит сущность.

**Свойства класса:**

* **public virtual int X { get; set; }** – изменение координаты х.
* **public virtual int Y { get; set; }** – изменение координаты y.
* **public field EntityBase { get; set; }** – установка симуляционного поля для сущности и её размещение. Размещение сущности производится при помощи вызова метода AddEntity класса field. Перед заданием свойства следует убедиться, что значения полей х и y не превышают размеров матрицы EntityMatrix класса field и неотрицательны, так как они используются как индексы.
* **public string Type { get; protected set; }** – строка, представляющая собой краткое описание сущности. Может служить для различения сущностей на симуляционном поле.

**Методы класса:**

* **public abstract void BeatAction()** – тактовое действие сущности. Абстрактный метод. Вызов метода производится из метода DoBeat() класса field.