# Задание

Работа выполняется в парах. Один студент из пары выполняет реализацию классов и предоставляет второму описание доступных (публичных) свойств и методов. Второй студент реализует интерфейс программы для работы с этими классами. В интерфейсе необходимо предусмотреть доступ ко всем возможностям всех классов.

Для второй задачи студенты меняются ролями.

Обмен программами и проверка работы производится через SVN-репозиторий на assembla.com. Все промежуточные версии сохраняются в отдельных папках, с пометками – что было изменено.

Примерный порядок разработки:

|  |  |  |
| --- | --- | --- |
| *№* | *Разработчик классов* | *Разработчик интерфейса* |
|  | Совместное обсуждение задачи. Определение перечня классов, их основных атрибутов и методов. Создание UML-диаграммы классов. |
|  | Создание «скелета» модуля: объявить публичные все классы и методы, объявить их реализацию, но оставить пока пустыми. Передать «скелетную» версию файла коллеге. | Разработать основной интерфейс программы. Определиться со способами ввода-вывода, добавить нужные компоненты. |
|  | Заполнение «скелета» реализацией методов и доступа к полям. По мере обновления проекта новые версии предоставляются коллеге. | Подключение классов к интерфейсу. Реализация методов ввода-вывода. При обнаружении ошибок в реализации классов сообщить о них коллеге. |
|  | Тестирование и отладка. Каждый участник исправляет программный код только в зоне своей ответственности. |

# Assembla

Assembla – популярный ресурс для совместной разработки программ и публикации открытого исходного кода. Assembla предоставляет как платные, так и бесплатные сервисы: хранение файлов в общем для всех разработчиков репозитории, синхронизация и журналирование внесенных изменений, обмен сообщениями и тикетами (на них отмечаются текущие задачи) и т.д.

Бесплатно предоставляется возможность создавать репозитории SVN (subversion), Git и P4. Мы будем использовать SVN (подробнее см. на Wiki или других ресурсах).

Для создания бесплатного репозитория (repository) и совместной работы в нем необходимо:

1. Найти на assembla.com ссылку для создания бесплатного (free) SVN-репозитория (не путать с пробным – trial).
2. Зарегистрироваться.
3. Создать SVN-пространство.
4. Одному участнику команды пригласить в свое пространство коллегу и преподавателя (логин NastasSanna). Все приглашенные участники могут загружать и обновлять файлы в общем хранилище.
5. Скачать и установить один из бесплатных SVN-клиентов (рекомендуется RapidSVN). После установки необходимо создать локальную копию вашего репозитория (указать URL репозитория и локальную директорию для синхронизации; логин, пароль при подключении). Клиент дает возможность скачивать обновленные файлы с сервера и, наоборот, выгружать на сервер.

# Пример

1. Разработать классы для геометрических фигур. Любая фигура (TFigure) имеет название, пару координат на плоскости, методы перемещения, координаты ее нижней, верхней, левой и правой границ, вычисления периметра и площади. Реализовать фигуры: эллипс (TEllipse), окружность (TCircle), кольцо (TRing). Эллипс дополнительно задается двумя радиусами, окружность – одним радиусом, кольцо – радиусами двух составляющих его окружностей.
2. Разработать пошаговую игру «Трава». На двумерном поле, размеченном на клетки, находится земля, семена и трава. Трава рассеивает семена в соседние свободные клетки с землей. Семена могут прорасти в новую траву или погибнуть, став опять землей. Трава также может погибнуть. Размер поля и вероятности засевания семян, роста и гибели задаются пользователем. Начальное расположение травы случайное.

# Варианты

|  |  |  |
| --- | --- | --- |
| Андреюшкина Т. | Кспоян Г. |  |
| Беликова Н. | Крючкова В. |  |
| Виноградов А. | Чугунова Е. |  |
| Груднов Д. | Федорова И. |  |
| Кандалова И. | Мишанин М. |  |
| Киселев К. | Валиев А. |  |
| Сербин И. | Гладких М. |  |

1. Разработать классы для работы с матрицами. Базовый класс – двумерный массив (T2DArray) поддерживает создание массива произвольного размера, добавление и удаление строк и столбцов, заполнение значений элементов. Класс-матрица (TMatrix) должен поддерживать операцию транспонирования, а также действия с матрицами: умножение матрицы на число, сложение матриц, умножение матриц. Класс – квадратная матрица (TSquareMatrix) должен поддерживать проверку на симметричность, создание единичной матрицы.
2. Разработать пошаговую игру «Зомби и жители». На двумерном поле находятся зомби, жители и препятствия.

Зомби и жители могут перемещаться, за 1 ход на любую соседнюю свободную клетку, в том числе по диагонали.

Если зомби и житель оказались в соседних клетках, то они сражаются. Если вокруг зомби жителей больше, чем зомби (считая его самого), то зомби гарантировано погибает. Если зомби и жителей поровну, или жителей меньше, то зомби погибает с некоторой вероятностью (чем больше жителей, тем выше вероятность). Если вокруг жителя есть зомби, то он заражается с некоторой вероятностью (тем выше, чем больше по соседству зомби). При сражении движение на соседние клетки не происходит.

Пользователь задает размер поля, количество зомби, жителей и препятствий. Игра завершается, если не осталось зомби или жителей.

1. Разработать класс TPair, описывающий пару целых чисел (x, y). Сравнение пар чисел производится по правилам: пара (x1, y1) > (x2, y2), если x1 > x2 или, если x1 = x2 и y1 > y2; пары чисел равны, если x1 = x2 и y1 = y2; в противном случае считается, что (x1, y1) < (x2, y2). Сложение и вычитание производится по правилам:

(x1, y1) + (x2, y2) = (x1 + x2, y1 + y2)

(x1, y1) – (x2, y2) = (x1 – x2, y1 – y2)

Создать два класса-наследника: двумерный вектор (T2DVektor) и денежная сумма в рублях и копейках (TMoney). Переопределить операции сложения и вычитания. Для вектора добавить вычисление его длины.

1. Разработать пошаговую игру «Амебы и вирус». На двумерном поле, размеченном на клетки, находятся амебы и лекарство. Изначально в пространстве есть какое-то количество любым образом размещённых амёб и лекарства. Какая-то часть амёб заражена вирусом.

Амёбы двигаются. Амёба может переместиться на любую свободную из восьми соседних клеток. Если рядом есть лекарство, то амёба перемещается на одну из этих клеток (правило выбора задать самостоятельно). Если нет, то амёба перемещается случайным образом, в том числе может остаться на месте. Если все соседние клетки заняты другими амёбами, то движение не осуществляется.

Заражение / Выздоровление. Если здоровая амёба соседствует с заражённой, то может заразиться вирусом с некоторой вероятностью. Вероятность тем выше, чем больше заражённых соседок. И тем ниже, чем больше лекарства носит с собой амёба. Каждая заражённая амёба имеет некоторый шанс выздороветь. Шанс тем выше, чем больше лекарства носит амёба.

Гибель / Деление. Заражённые амёбы могут погибнуть с некоторой вероятностью. На месте гибели амебы остаётся все ее лекарство, если оно у нее было, или пустое место. Здоровая амеба может поделиться на две амёбы. Для этого рядом с ней должна быть клетка, не занятая другими амёбами. При делении лекарство делится пополам.

Количество лекарства в амёбах в конце хода уменьшается.

Игра прекращается, если все амебы умерли или выздоровели. Пользователь задает размер игрового поля, количество лекарства, больных и здоровых амеб в начале игры.

1. Создать абстрактный класс – математическую функцию (TMathFunction) с поддержкой вычисления значения функции в точке x, построения ее графика в заданном диапазоне значений [xmin; xmax], вычисление значения производной в точке x. Создать иерархию наследников: полином (TPoly), парабола (TParabola), гипербола (THyperbola).
2. Разработать пошаговую игру «Инь-Ян». На двумерном поле расположены ячейки, каждая из которых может быть пустой (мертвой), живой ячейкой Инь и живой ячейкой Ян. Соседними считаются 8 ячеек.

Рождение. Если у пустой ячейки ровно три соседа (живых) и они не все одинаковые, то в ней рождается Ян, когда среди соседей только один Ян, или Инь, когда среди соседей только один Инь.

Гибель от перенаселения (одиночества). Живая ячейка, имеющая больше четырех (меньше двух) соседей, умирает от перенаселения (от одиночества);

Гибель в неравном противостоянии. Если у живой ячейки ровно четыре соседа, из которых большинство – противоположного типа – ячейка умирает.

Игра заканчивается, если вселенная вырождается (остается только Инь или только Ян, или только мертвые клетки). Пользователь задает размер поля и количество ячеек Инь и Ян в начале игры.

1. Создать абстрактный класс-прогрессию (TSeries) с абстрактными методами вычисления i-того члена прогрессии и суммы i первых членов. Создать классы-наследники: арифметическую (TLinear) и геометрическую (TExponential) прогрессию. Добавить методы создания прогрессий по значениям первых двух членов. Добавить прогрессию-сумму (TSumSeries), члены которой вычисляются как суммы членов двух других прогрессий (произвольных).
2. Разработать пошаговую игру «Сахарный мир». На двумерном поле находятся инфузории и сахар. Инфузории могут перемещаться в поисках пищи, тратя на каждый шаг энергию. Если в соседних клетках есть сахар, инфузория перемещается в любую из них и съедает сахар, пополняя уровень энергии. Если сахара нет, она перемещается на любую соседнюю свободную клетку. Инфузории не могут перемещаться по диагонали. Если у инфузории не осталось энергии, она погибает.

Если запас энергии у инфузории выше определенного порога и рядом есть свободная ячейка, инфузория делится. При этом одна инфузория перемещается на свободную клетку, другая остается на месте. Запас энергии делится пополам.

На каждом шаге на свободных клетках может появиться новый сахар. Количество сахара в каждой клетке случайно.

Игра заканчивается, если все инфузории погибнут. Пользователь задает размер поля, количество инфузорий в начале и вероятность появления сахара.

1. Разработать класс-множество (TSet), которое может содержать произвольное число однотипных элементов. Множество должно поддерживать добавление и удаление элементов, определение числа элементов, операции с множествами: объединение, пересечение, а также проверку двух множеств на равенство. Реализовать наследников: множество целых чисел (TIntSet), множество вещественных чисел (TRealSet) и множество строк текста (TStringSet).
2. Разработать пошаговую игру «Элементы». Двумерное поле полностью заполнено тремя элементами: водой, огнем и воздухом. Начальное размещение элементов случайное. У каждой клетки 4 соседа.

Если в соседних от огня клетках нет воздуха, огонь гаснет (становится воздухом).

Если в соседних с водой клетках не менее 2 огней и меньше 2 клеток воды, вода испаряется (становится воздухом).

Если в соседних с воздухом клетках огня больше, чем воды, он загорается (становится огнем).

Если в соседних с воздухом клетках воды больше, чем огня, его затапливает (становится водой).

Игра заканчивается, если один из элементов исчезает с поля. Пользователь задает начальный размер поля, количество огня и воды (остальное – воздух).

1. Разработать класс TTriad, описывающий тройку целых чисел (x, y, z). Сравнение двух троек производится по правилам: (x1, y1, z1) < (x2, y2, z2), если x1 < x2 и y1 < y2 и z1 < z2; тройки равны, если x1 = x2 и y1 = y2 и z1 = z2; в противном случае считается, что (x1, y1, z1) > (x2, y2, z2). Сложение и вычитание производится по правилам:

(x1, y1,z1) + (x2, y2,z2) = (x1 + x2, y1 + y2, z1 + z2)

(x1, y1,z1) – (x2, y2,z2) = (x1 – x2, y1 – y2, z1 + z2)

Создать два класса-наследника: трехмерный вектор (T3DVektor) и дата, содержащая день, месяц и год (TDate). Переопределить для них операции сложения и вычитания. Для вектора добавить вычисление его длины.

1. Разработать пошаговую игру «Живое и мертвое». На двумерном поле находятся живые и мертвые клетки. Клетка имеет восемь соседей. Распределение живых клеток в начале игры называется первым поколением. Каждое следующее поколение рассчитывается на основе предыдущего по таким правилам:

пустая (мёртвая) клетка ровно с тремя живыми клетками-соседями оживает;

если у живой клетки есть две или три живые соседки, то эта клетка продолжает жить; в противном случае (если соседок меньше двух или больше трёх) клетка умирает (от «одиночества» или от «перенаселённости»).

Игра прекращается, если на поле не останется живых клеток. Пользователь задает размер поля и количество живых клеток в начале игры.

1. Создать классы для описания металлов (TMetal) и сплавов (TAlloy), являющиеся наследниками класса вещество (TSubstance). У вещества задается название, плотность и температура плавления. Металл задается обозначением химического элемента, его номером и атомной массой. Сплав задается списком входящих в него металлов с указанием количества (массы). Металлы в составе не должны повторяться. У сплава можно вычислить общую массу и долю каждого металла в общей массе. К сплаву можно добавить металл или другой сплав, в результате получится новый сплав, в котором будут присутствовать все металлы из первых двух. Из состава сплава можно полностью удалить один металл, если он не последний.
2. Разработать пошаговую игру «Гном и золото». На двумерном игровом поле находятся случайным образом сгенерированные клетки с пустой породой и золотом. В начале игры в центре поля на очищенной клетке находится управляемый игроком гном.

На каждом ходу пользователь выбирает направление движения гнома (на соседнюю клетку, двигаться по диагонали нельзя). Переход на очищенную клетку бесплатный, на рытье пустой породы или золота тратится 1 ед. золота. При добыче золота гном получает от 3 до 10 ед. золота.

Игра завершается, если собрано все золото. Пользователь задает размер поля, количество пустой породы (не менее половины поля), золота на поле и золота у гнома в начале игры.