Домашнее задание 8

Космический корабль

Вам предстоит разработать аналог игры Астероиды. В разрабатываемой версии игрок управляет космическим кораблём с помощью 4х кнопок: 2 кнопки для поворота (по часовой и против часовой стрелке), третья кнопка для придания кораблю скорости, и ещё одна позволяет стрелять ракетами в препятствия. По экрану летают большие астероиды в случайном направлении со случайной скоростью. Цель игры состоит в том, чтобы разрушить все астероиды до столкновения с космическим кораблём. В аркадной версии большие астероиды при попадании в них ракетой распадаются на мелкие части, которые так же могут повредить корабль и которые следует уничтожать. Периодически появляется летающая тарелка, которая стремится уничтожить летающий корабль.

Разработка игры будет разбита на два домашних задания. В первом вам необходимо разработать прототип игры, включающий летающий космический корабль, один астероид и одну ракету. Шаблон программы содержит необходимый код и графику для того, чтобы игра выглядела современнее.

Рекомендованный порядок работы над заданием

Процесс создания прототипа игры следует разделить на четыре этапа.

Этап 1. Космический корабль

На этом этапе необходимо реализовать схему управления космическим кораблём, для чего потребуется работать с классом Spaceship (космический корабль) и обработчиком нажатия на клавиатуру.

Корабль должен реагировать на нажатие кнопок следующим образом:

- стрелки влево и вправо контролируют ориентацию корабля в пространстве. При нажатии стрелки влево, корабль должен поворачиваться против часовой стрелки. При нажатии стрелки вправо – по часовой. Если не нажата ни одна из этих кнопок, корабль не должен менять свою ориентацию в пространстве. Вам потребуется подобрать разумную угловую скорость для поворота корабля.

- стрелка вверх контролирует работу двигателя космического корабля. Двигатель включается при нажатии кнопки вверх и выключается, если кнопка вверх отпущена. Во время включённого двигателя следует рисовать огонь у турбин корабля. Если двигатель не включен, то огня быть не должно.

- когда двигатель включён, корабль должен ускоряться в направлении «прямо по курсу». Вектор направления может быть рассчитан в зависимости от угла корабля с помощью функции *angle_to_vector*. Следует поэкспериментировать с масштабированием компонент вектора ускорения для определения разумного ускорения корабля.

- следует помнить, что во время движения корабль ускоряется в прямом направлении, но при этом двигается в зависимости от вектора скорости. Эти два направления могут не совпадать. Возможность ускориться в направлении, отличном от текущего направления движения, является отличительной чертой игры Астероиды.

- необходимо учитывать силу трения при движении корабля. Для этого при обновлении скорость всегда нужно умножать на константу, меньшую 1, что будет тормозить корабль. После остановки двигателя корабль через некоторое время должен остановиться.

Рекомендуется организовать работу над управлением кораблём в следующем порядке.

1. Отредактируйте метод рисования для класса Ship для вывода изображения корабля (без огня у турбин) вместо рисования кружочка. Метод должен учитывать позицию и угол корабля. Следует помнить, что угол должен считаться в радианах, а не в градусах. Если метод рисования корабля вызывается из обработчика рисования, то корабль должен отображаться на экране.
   Поэкспериментируйте с различными позициями и углами корабля, чтобы убедиться, что рисование работает корректно.
2. Разработайте первоначальную версию метода обновления позиции корабля. Она должна обновлять позицию корабля, основываясь на его скорости.
   Если метод обновления вызывается из обработчика рисования, корабль должен начать двигаться. Проверьте движение корабля с различными начальными скоростями.
3. Модифицируйте метод обновления для корабля так, чтобы изменять его угол ориентации в соответствии с текущим значением угловой скорости.
4. Научите корабль двигаться в зависимости от нажатия на правую/левую стрелки.
   Для этого добавьте методы в класс Ship для увеличения и уменьшения угловой скорости на фиксированную величину (подберите по вашему усмотрению).
   Добавьте обработчик нажатия и отпускания кнопок, который должен реагировать на кнопки вверх/вниз и вызывать соответствующие методы класса Ship.
   Проверьте, что корабль двигается, как задумано.
5. Модифицируйте обработчик клавиатуры для включения/выключения огня у турбин корабля. Добавьте метод к классу Ship, отвечающий за включение и выключение двигателя. Состояние двигателя можно передать с помощью аргумента булевского типа.
6. Модифицируйте метод рисования корабля для рисования огня, когда двигатель включен.
7. Модифицируйте метод включения двигателя корабля для проигрывания звука работающих турбин, когда двигатель включен. При выключении двигателя перематывайте звук в начало.
8. Добавьте код к методу обновления позиции корабля для учёта ускорения при включённом двигателе.
   Используйте вспомогательную функцию *angle_to_vector* для получения вектора ориентации корабля по углу поворота. Этот вектор является вектором ускорения корабля. Обновите вектор скорости с учётом вектора ускорения. Вам потребуется умножить вектор ускорения на небольшое число, чтобы корабль не слишком сильно ускорялся.
9. Далее модифицируйте метод обновления корабля таким образом, чтобы при пересечении им края экрана корабль появлялся на противоположном крае (используйте вычисления по модулю).
10. На настоящий момент корабль никогда не замедляется. Добавьте трение к методу обновления корабля. Для этого при каждом обновлении умножайте каждую из компонент скорости на число, меньшее 1.

Таким образом, на текущий момент корабль умеет летать по экрану. Подберите значения констант так, чтобы управление было удобным для игрока.

Этап 2. Астероиды

Для создания астероидов будет использован класс Sprite. Обратите внимание, что метод обновления позиции для спрайта очень напоминает метод обновления для корабля. Главное отличие состоит в том, что скорость корабля и поворот контролируется с помощью клавиш, в то время как спрайты движутся в соответствии с параметрами, установленными случайно при создании. Выходя за границы экрана, астероиды должны появляться с противоположной его стороны так же, как и корабль.

В шаблоне при старте игры создаётся один астероид с нулевой скоростью (переменная *a_rock)*. Вместо этого следует создавать астероид, заменяя *a_rock* каждую секунду в обработчике таймера.

В следующем задании игра будет работать с несколькими астероидами. В этой версии астероиды не таранят корабль, соприкасаясь с ним.

Работу над астероидом выполняйте в следующем порядке.

1. Завершите реализацию класса Sprite.
   Модифицируйте обработчик рисования, чтобы он рисовал изображение астероида.
   Реализуйте метод обновления, чтобы спрайт двигался и крутился. Астероиды не ускоряются и не имеют трения, так что этот метод обновления должен быть проще, чем у корабля.
   Проверьте правильность работы, подавая на вход разные начальные параметры и убедитесь, что всё работает правильно.
2. Модифицируйте обработчик таймера *rock_spawner*. Устанавливайте в качестве значения *a_rock* новый астероид каждый такт. Не забудьте объявить *a_rock* как глобальную переменную в обработчике таймера.
   При создании астероида случайно генерируйте скорость, положение и угловую скорость. Вам потребуется подобрать рамки для случайных величин, чтобы в игру было интересно играть. Удостоверьтесь, что астероиды крутятся и двигаются во всех направлениях.

Этап 3. Ракеты

Ракеты также будут реализованы на основе класса Sprite, как и астероиды. Ракеты всегда имеют нулевую угловую скорость. У них есть время жизни (они должны исчезнуть по прошествии времени, иначе ракеты будут повсюду), но пока эта функция не будет реализована. На данный момент, необходимо сделать так, чтобы можно было запустить 1 ракету, которая не разрушает астероиды. Взрывы будут добавлены позже.

Ракета должна вылетать по нажатию на пробел, а не по таймеру, как астероиды. И она так же должна, вылетая за границы экрана появляться с другой стороны.

Реализуйте ракету в следующем порядке.

1. Добавьте метод *shoot* к классу корабль. Этот метод должен выпускать новую ракету (на данный момент необходимо поменять существующую ракету в переменной *a_missile).* Начальная позиция ракеты должна совпадать с координатой кончика пушки корабля. Её скорость является суммой скорости корабля и нескольких векторов направления.
2. Модифицируйте обработчик нажатия на кнопку для вызова метода shoot, если нажата кнопка пробела.
3. Убедитесь, что звук ракет воспроизводится во время выстрела.

Этап 4. Интерфейс пользователя

Интерфейс пользователя игры показывает количество оставшихся жизней и очки (результат). Их подсчёт отложите до следующего задания. Добавьте код к обработчику рисования для рисования показателей на холсте. Используйте глобальные переменные *lives* и *score* для хранения оставшихся жизней и очков.

Дополнительные расширения

Добавьте второй слой анимированного фона, который будет находиться под существующим и двигаться медленней. Возможно, стоит разделить астероиды в этих слоях по размерам – в более быстром (и близком) слое – покрупнее, в более медленном (дальнем) – помельче.

Оценка задания

Программа отображает изображение корабля	-	5 %.
Корабль летит по прямой, если двигатель выключен	-	5 %.
Корабль вращается при нажатии на клавиши ← и →	-	10 %.
Ориентация корабля не зависит от его скорости	-	5 %.
При нажатии клавиши ↑ программа рисует корабль с огнём двигателей	-	5 %.
Программа воспроизводит звук двигателя при нажатой клавише ↑	-	5 %.
Корабль ускоряется в направлении своей ориентации при нажатой клавише ↑	-	5 %.
При пересечении границы экрана корабль появляется с другой стороны	-	5 %.
Корабль останавливается, если двигатели выключены	-	5 %.
Программа отображает изображение астероида	-	5 %.
Астероид движется прямолинейно с постоянной скоростью	-	5 %.
Астероид пересоздаётся каждую секунду	-	5 %.
Астероид создаётся со случайными позицией, скоростью и угловой скоростью	-	5 %.
При нажатии на пробел выпускается ракета	-	5 %.
Ракета создаётся на конце пушки корабля	-	5 %.
Скорость ракеты определяется на основе скорости и направления корабля	-	5 %.
При запуске ракеты программа проигрывает соответствующий звук	-	5 %.
Программа выводит число жизней в левом верхнем углу экрана	-	5 %.
Программа выводит очки в правом верхнем углу экрана	-	5 %.

Дополнительные баллы

Второй слой анимированного фона	-	15 %.

Срок сдачи задания.

Для подгруппы, занимающейся по четвергам: 30 октября 2014;

для подгруппы, занимающейся по вторникам: 3 ноября 2014.

В случае сдачи задания с опозданием, полученные за него баллы будут уменьшены:

при задержке на 1 неделю: баллы умножаются на 0.9;

при задержке на 2 недели: баллы умножаются на 0.75;

при задержке на 3 и более недель: баллы умножаются на 0.65.