Семинар #1: Библиотека SFML. Домашнее задание.

Все задачи нужно решить обязательно с использование только одной сторонней библиотеки – SFML.

Задача 1. Цикл

Напишите программу, которая будет рисовать круг, движущийся с постоянной скоростью по горизонтали слева вправо. После того, как круг выходит за границы экрана с правой стороны, он должен появиться с левой стороны.

Задача 2. Вращающийся квадрат

Напишите программу, которая рисует вращающийся квадрат. Квадрат должен вращаться вокруг своего центра, а не вокруг одной из своих вершин. Используйте метод `setOrigin`.

Задача 3. Движение по кругу

Напишите программу, которая будет рисовать круг, двигающийся по окружности.

Задача 4. Привет мир!

Напишите программу, которая будет рисовать на экран фразу `Привет Мир!`.

Задача 5. Координаты мыши

Напишите программу, которая будет рисовать на экране текст, в котором содержатся координаты мыши в системе координат пикселей и в системе координат окна (то есть на экран должны выводиться 4 числа). При изменении положения курсора мыши, текст должен меняться. Используйте функцию `std::to_string` для конвертации в строки.

Задача 6. Движение с помощью стрелочек

Напишите программу, которая будет рисовать кружок на экране. Кружок должен перемещаться влево, вправо, вниз и вверх при нажатии на соответствующие клавиши стрелочек.

Задача 7. Притяжение к мыши

Напишите программу, которая будет рисовать кружок на экране. При зажатии левой кнопки мыши (ЛКМ), кружок должен начать двигаться к курсору мыши. Решите задачу в двух вариантах:

a. Кружок должен двигаться к курсору мыши с постоянной скоростью. Если ЛКМ не зажата, то кружок должен оставаться на месте.

b. Кружок должен двигаться к курсору мыши с постоянным ускорением. Если ЛКМ не зажата, то кружок должен продолжать двигаться с постоянной скоростью.

Программа должна работать корректно при изменении размера окна. Используйте метод `mapPixelToCoords`.

Задача 8. Броуновское движение

Напишите программу, которая будет моделировать броуновское движение частиц. Представьте каждую частицу в виде шарика, который перемещается на случайное небольшое смещение в случайные моменты времени.

Для случайных чисел используйте функцию:

```
float getRandomFloat(float min, float max)
{
    static std::random_device rd;
    static std::mt19937 gen(rd());
    std::uniform_real_distribution<float> d(min, max);
    return d(gen);
}
```

Задача 9. Задача n тел

В двумерном пространстве находятся n шариков с различными массами и различными электрическими зарядами. Напишите программу, которая будет моделировать движение таких шариков.

- Шарики действуют друг на друга силой Кулона:

$$F = \frac{q_1 \cdot q_2}{R}$$

  Где $q_{1,2}$ – заряды шариков, R – расстояние между шариками. Обратите внимание, что сила взаимодействия обратно пропорциональна первой степени расстояния, а не второй. Это правильная формула для силы Кулона в двух измерениях.

- Считайте, что силы гравитации между шариками пренебрежимо малы по сравнения с элекрическими силами. Их можно не учитывать.

- Столкновения шариков друг с другом можно тоже не учитывать. Шарики взаимодействуют только посредством силы Кулона.

- На границах окна поставьте стенки. Шарики должны упруго отскакивать от стенок.

- Расчёт ускорений, скоростей и положений всех шариков проводите с шагом Δt. В этой задаче можно считать, что Δt постоянна и равна $1/fps$. Где значение количества кадров в секунду (fps) задаётся с помощью метода `setFramerateLimit`.

- Если два шарика подойдут слишком близко друг к другу, то сила взаимодействия может стать очень большой. Это приведёт к большой погрешности в вычислениях из-за того, что Δt больше характерного времени изменения силы взаимодействия между шариками. В результате этого шарики начнут чрезмерно быстро двигаться. Чтобы исправить этот баг просто сделайте так, чтобы сила взаимодействия шариков была равна нулю, если расстояние между шариками меньше некоторой величины.

- Начальные значения масс, зарядов, положений шариков задайте случайным образом из некоторых диапазонов. Начальные значения скоростей равны нулю.

- Цвет шарика должен зависит от его заряда. Положительно заряженные шарики рисуйте красным цветом, а отрицательно зарженные – синим.

- (*****) Добавьте возможно добавлять шарики, используя мышь. При нажатии левой кнопки мыши, в том месте, где находится курсор, должен создаваться шарик с маленькой массой и с отрицательным зарядом. При нажатии правой кнопки мыши должен создаваться шарик с очень большой массой и положительным зарядом.

Необязательные задачи

Задача 1. Случайные перемещения

Напишите программу, которая бы рисовала круг, который бы перемещался в случайное место на экране каждые 2 секунды. При этом, все остальное должно работать как прежде, то есть функцию `sf::sleep` использовать не получится. Используйте классы `sf::Time` и `sf::Clock`.

Задача 2. Изменение цвета

Напишите программу, которая бы рисовала круг, цвет которого зависил бы от положения мыши на экране. Красная компонента цвета круга должна меняться от 0 до 255 при изменении x-компоненты курсора мыши. Синяя компонента должна меняться от 0 до 255 при изменении y-компоненты курсора мыши.

Задача 3. Столкновение курсора с кругом

Напишите программу, которая бы рисовала круг, цвет которого изменялся бы при наведении на него мышью. Если курсор мыши находится вне круга, то круг должен рисоваться зелёным цветом. Если курсор мыши находится внутри круга, то круг должен рисоваться красным цветом.

Задача 4. Select Move Delete

В папке `select_move_delete/` содержится заготовка исходного кода для этого задания. В этой программе есть несколько объектов(кругов), которые можно выделять. Выделение происходит по нажатию левой клавиши мыши. Зажав клавишу `Ctrl` можно выделить несколько объектов. Также в программе реализован прямоугольник выделения (но он пока не выбирает объекты). Левой кнопкой мыши с зажатой клавишей левый `Alt` можно создать круг случайного размера. Добавьте следующие возможности в программу:

- Задание случайного цвета. При нажатии клавиши пробел цвет всех выделенных шаров должен меняться на случайный. Для этого понадобится добавить поле `color` в класс `Ball`.
- Выделение объектов с помощью прямоугольника выделения. Прямоугольник выделения должен рисоваться только если нажатие мыши произошло вне кругов. Все объекты, полностью находящиеся внутри прямоугольника выделения, на момент отпускания левой кнопки мыши должны выделяться. Объекты должны выделяться во время изменения прямоугольника, а не только при отпускании кнопки мыши.
- Перемещение всех выделенных объектов при зажатии левой клавиши мыши и её движении. Перемещаться должны все выделенные объекты параллельно (также как перемещаются несколько выделенных значков на рабочем столе). Прямоугольник выделения при этом рисоваться не должен.
- При нажатии клавиши `Delete`, все выделенные объекты должны удаляться. Чтобы удалить элемент из `std::list` используйте итераторы и метод `erase`. При удалении элемента `std::list` нужно внимательно следить за тем, чтобы не испортить итераторы. Этот момент был более рассмотрен в семинаре #4.
- В папке `../classroom_tasks_solutions/context_menu/` содержится реализация контекстного меню – класс `ContextMenu`. Используйте этот класс чтобы добавить контекстное меню в программу. Оно должно открываться при нажатии правой кнопки мыши. Добавьте следующие варианты в меню:
  - `Delete` - при нажатии на выделенные объект правой кнопкой мыши и выборе этого варианта все выделенные объекты должны удаляться.
  - `Create` - при нажатии на любое место и выборе этого варианта должнен создаваться новый случайный круг.
  - `Random Color` – все выделенные объекты должны окрашиваться в случайный цвет.
  - `Increase` – все выделенные объекты должны увеличиваться на 25 процентов.
  - `Decrease` – все выделенные объекты должны уменьшаться на 25 процентов.
- **Copy Paste Cut** Добавьте возможность копирования, вставки и вырезания объектов. Вызов этих операций должен происходить в контекстном меню или с помощью комбинаций клавиш `Ctrl-C`, `Ctrl-V` и `Ctrl-X`.