

Задачи:

1. **bash 1:** Написать bash-скрипт, который печатает на экран Hello world. Используйте команду echo.
2. **bash 2:** Написать bash-скрипт, который создаёт папку test и файл test.txt в котором будет написано "Testing". Используйте команды mkdir, touch и cat.
3. **Раздельная компиляция:** Создайте файлы hello.h и hello.cpp. Эта мини-библиотека должна содержать 1 функцию hello(), которая печатает на экран Hello world. Создайте файл main.cpp и вызовите из этого файла эту функцию. Скомпилируйте оба исполняемых файла отдельными вызовами gcc. Создайте bash-скрипт, который будет компилировать оба файла, скомпилируйте оба исполняемых файла отдельными вызовами gcc.
4. **Раздельная компиляция. Make-файл:** Сделайте то же самое, только с помощью make-файла.

Работа с библиотекой SFML:

Простейший пример:

```
#include <SFML/Graphics.hpp>

int main()
{
    // создаём окно
    sf::RenderWindow window(sf::VideoMode(800, 600), "My window");
    // цикл, который будет работать пока открыто окно
    while (window.isOpen())
    {
        // проверяем все события, связанные с окном, произошедшие со времени прошедшей итерации
        sf::Event event;
        while (window.pollEvent(event))
        {
            // проверяем на закрытие окна
            if (event.type == sf::Event::Closed)
                window.close();
        }
        // очистить окно черным цветом
        window.clear(sf::Color::Black);

        // рисуем тут...
        // window.draw(...);

        // конец текущего кадра
        window.display();
    }
}
```

1. **Компиляция:** Скомпилируйте программу, приведенную выше. Помните, что на этапе линковки нужно указать опции -lsfml-graphics -lsfml-window -lsfml-system.
2. **Рисуем круг:** Нарисовать синий круг радиуса 100 в центре экрана.

```
sf::CircleShape shape(50); // создаём экземпляр круга с помощью конструктора
shape.setFillColor(sf::Color(100, 250, 50)); // задаём цвет круга
shape.setPosition(200, 400); // задаём положение круга
window.draw(shape)
```

3. **Анимация 1:** Нарисуйте круг радиуса 10,двигающийся с постоянной скоростью. Используйте метод `move(dx, dy)`:

```
shape.move(5, 10); // задаём положение круга
```

4. **Анимация 2:** Нарисуйте круг радиуса 10,двигающийся по окружности.

5. **События 1:**

Пример обработки события нажатия клавиши:

```
if (event.type == sf::Event::KeyPressed)
    if (event.key.code == sf::Keyboard::Escape)
    {
        std::cout << "the escape key was pressed" << std::endl;
    }
```

Дописать программу так, чтобы анимация работала только тогда, когда зажата клавиша M.

6. **События 2:**

Пример обработки события нажатия кнопки мыши:

```
if (event.type == sf::Event::MouseButtonPressed)
{
    if (event.mouseButton.button == sf::Mouse::Right)
    {
        std::cout << "the right button was pressed" << std::endl;
        std::cout << "mouse x: " << event.mouseButton.x << std::endl;
        std::cout << "mouse y: " << event.mouseButton.y << std::endl;
    }
}
```

Дописать программу так, чтобы при нажатии клавиши мыши кружок перемещался в на место курсора.

7. **Класс Ball:**

Создать класс `Ball` с полями `x`, `y`, `vx`, `vy`, `radius`. В функции `main()` создать вектор из экземпляров класса `Ball`. При нажатии на клавишу мыши должен создаваться новый экземпляр класса в соответствующем месте. Скорость задаётся случайным образом(но не делайте её очень большой). Радиус равен 5. Все экземпляры должны правильно отрисовываться.

8. **Граничные условия**

Добавьте стенки, так чтобы шарики не улетали за пределы экрана.

9. **Задача N тел**

Добавьте гравитационное взаимодействие между шариками. Считайте что масса всех шариков равна 1.

10. **Задача N тел с массой**

Добавьте разную массу шарикам. При создании шарика масса должна задаваться случайным образом(но не делайте массу слишком большой либо слишком маленькой!).

11. **Солнечная система**

Смоделируйте солнечную систему в 2D (масштабы можно не соблюдать).

12. **Создайте игру pong**

Смоделируйте солнечную систему в 2D (масштабы можно не соблюдать). Для рисования прямоугольника:

```
sf::RectangleShape rectangle(sf::Vector2f(120, 50));
```