```
import pygame
import time
import random
import math
pygame.init()

WIDTH, HEIGHT = 800, 600

win = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Fireworks!")

FPS = 60

COLORS = [
    (255, 0, 0),
    (0, 255, 0),
    (0, 0, 255),
    (0, 255, 255),
    (255, 165, 0),
    (255, 255, 255),
    (230, 230, 250),
    (255, 192, 203)
]


class Projectile:
    WIDTH = 5
    HEIGHT = 10
    ALPHA_DECREMENT = 3

    def __init__(self, x, y, x_vel, y_vel, color):
        self.x = x
        self.y = y
        self.x_vel = x_vel
        self.y_vel = y_vel
        self.color = color
        self.alpha = 255

    def move(self):
        self.x += self.x_vel
        self.y += self.y_vel
        self.alpha = max(0, self.alpha - self.ALPHA_DECREMENT)

    def draw(self, win):
        self.draw_rect_alpha(win, self.color + (self.alpha,),
                             (self.x, self.y, self.WIDTH, self.HEIGHT))

    @staticmethod
    def draw_rect_alpha(surface, color, rect):
        shape_surf = pygame.Surface(pygame.Rect(rect).size,
pygame.SRCALPHA)
        pygame.draw.rect(shape_surf, color, shape_surf.get_rect())
        surface.blit(shape_surf, rect)


class Firework:
    RADIUS = 10
    MAX_PROJECTILES = 50
```

```python
    MIN_PROJECTILES = 25
    PROJECTILE_VEL = 4

    def __init__(self, x, y, y_vel, explode_height, color):
        self.x = x
        self.y = y
        self.y_vel = y_vel
        self.explode_height = explode_height
        self.color = color
        self.projectiles = []
        self.exploded = False

    def explode(self):
        self.exploded = True
        num_projectiles = random.randrange(
            self.MIN_PROJECTILES, self.MAX_PROJECTILES)

        if random.randint(0, 1) == 0:
            self.create_circular_projectiles(num_projectiles)
        else:
            self.create_star_projectiles()

    def create_circular_projectiles(self, num_projectiles):
        angle_dif = math.pi*2 / num_projectiles
        current_angle = 0
        vel = random.randrange(self.PROJECTILE_VEL - 1,
                               self.PROJECTILE_VEL + 1)
        for _ in range(num_projectiles):
            x_vel = math.sin(current_angle) * vel
            y_vel = math.cos(current_angle) * vel
            color = random.choice(COLORS)
            self.projectiles.append(Projectile(
                self.x, self.y, x_vel, y_vel, color))
            current_angle += angle_dif

    def create_star_projectiles(self):
        angle_diff = math.pi/4
        current_angle = 0
        num_projectiles = 32
        for i in range(1, num_projectiles + 1):
            vel = self.PROJECTILE_VEL + (i % (num_projectiles / 8))
            x_vel = math.sin(current_angle) * vel
            y_vel = math.cos(current_angle) * vel
            color = random.choice(COLORS)
            self.projectiles.append(Projectile(
                self.x, self.y, x_vel, y_vel, color))
            if i % (num_projectiles / 8) == 0:
                current_angle += angle_diff

    def move(self, max_width, max_height):
        if not self.exploded:
            self.y += self.y_vel
            if self.y <= self.explode_height:
                self.explode()

        projectiles_to_remove = []
        for projectile in self.projectiles:
            projectile.move()
            if projectile.x >= max_width or projectile.x < 0:
                projectiles_to_remove.append(projectile)
            elif projectile.y >= max_height or projectile.y < 0:
                projectiles_to_remove.append(projectile)

        for projectile in projectiles_to_remove:
            self.projectiles.remove(projectile)

    def draw(self, win):
        if not self.exploded:
            pygame.draw.circle(win, self.color, (self.x, self.y),
self.RADIUS)

        for projectile in self.projectiles:
            projectile.draw(win)


class Launcher:
    WIDTH = 20
    HEIGHT = 20
    COLOR = 'grey'

    def __init__(self, x, y, frequency):
        self.x = x
        self.y = y
        self.frequency = frequency  # ms
        self.start_time = time.time()
        self.fireworks = []

    def draw(self, win):
        pygame.draw.rect(
            win, self.COLOR, (self.x, self.y, self.WIDTH, self.HEIGHT))

        for firework in self.fireworks:
            firework.draw(win)

    def launch(self):
        color = random.choice(COLORS)
        explode_height = random.randrange(50, 400)
        firework = Firework(self.x + self.WIDTH/2,
                            self.y, -5, explode_height, color)
        self.fireworks.append(firework)

    def loop(self, max_width, max_height):
        current_time = time.time()
        time_elapsed = current_time - self.start_time

        if time_elapsed * 1000 >= self.frequency:
            self.start_time = current_time
            self.launch()

        fireworks_to_remove = []
        for firework in self.fireworks:
            firework.move(max_width, max_height)
            if firework.exploded and len(firework.projectiles) == 0:
                fireworks_to_remove.append(firework)
        for firework in fireworks_to_remove:
            self.fireworks.remove(firework)


def draw(launchers):
    win.fill("black")

    for launcher in launchers:
        launcher.draw(win)

    pygame.display.update()


def main():
    run = True
    clock = pygame.time.Clock()

    launchers = [Launcher(100, HEIGHT - Launcher.HEIGHT, 3000),
Launcher(300, HEIGHT - Launcher.HEIGHT, 4000),
                 Launcher(500, HEIGHT - Launcher.HEIGHT, 2000),
Launcher(700, HEIGHT - Launcher.HEIGHT, 5000)]

    while run:
        clock.tick(FPS)

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                run = False
                break

        for launcher in launchers:
            launcher.loop(WIDTH, HEIGHT)

        draw(launchers)

    pygame.quit()
    quit()


if __name__ == '__main__':
    main()
```