AUTHORS

- David Vajcenfeld (david.vajcenfeld@mail.utoronto.ca)
- Lorna Licollari (lorna.licollari@mail.utoronto.ca)

A FEW PASSES

```python
def test(car: Car, track: Track):
    max_velocity_pass(track, car)
    max_acceleration_pass(track, car)
    done = False
    while not done:
        done = pit_stop_pass(track, car)
        max_velocity_pass(track, car)
        max_acceleration_pass(track, car)
    return calc_points_time(track.points), track
```

FORWARD PASS: MAX VELOCITY

```python
def max_velocity_pass(track: Track, car: Car):
    """
    Calculates the maximum velocity in one forward pass.
    """

    def calc(prev: Point, this: Point):
        this.max_velocity = 0 if this.is_pit_stop \
            else min(car.top_speed,
                     calc_velocity(prev.max_velocity, car.acceleration),
                     calc_max_velocity(this.radius, car.handling),
                     calc_max_velocity(prev.radius, car.handling))

    track.points[0].max_velocity = 0
    for first, second in pairwise(track.points):
        # Find maximum possible velocities at each point.
        calc(first, second)
```

BACKWARDS PASS: MAX ACCELERATION

```python
def max_acceleration_pass(track: Track, car: Car):
    """
    Calculates the maximum acceleration (and corrects maximum velocity if necessary) in
    one backwards pass.
    """

    def calc(this: Point, nxt: Point):
        # Find maximum acceleration.
        this.max_acceleration = (nxt.max_velocity ** 2 - this.max_velocity ** 2) / 2
        if this.max_acceleration > car.acceleration:
            # Can't speed up enough, so correct next max velocity.
            nxt.max_velocity = calc_velocity(this.max_velocity, this.max_acceleration)
            if nxt.next:
                calc(this.next, nxt.next)
        if this.max_acceleration < -1 * car.braking:
            # Can't slow down enough, so correct previous max velocity.
            this.max_acceleration = -1 * car.braking
            this.max_velocity = calc_velocity(nxt.max_velocity, car.braking)
        nxt.max_velocity = calc_velocity(this.max_velocity, this.max_acceleration)

    track.points[-1].max_acceleration = 0
    for second, first in pairwise(track.points[::-1]):
        calc(first, second)
```

FORWARD PASS: PIT STOPS

```python
def pit_stop_pass(track: Track, car: Car):
    """
    Looks for first point where gas or tires run out in one forward pass,
    and add a pit stop at or slightly before it.
    """
    track.points[0].gas_usage = calc_gas_usage(track.points[0].max_acceleration)
    track.points[0].tire_wear = calc_tire_wear(track.points[0].max_acceleration)
    for prev, this in pairwise(track.points):
        this.gas_usage = prev.gas_usage + calc_gas_usage(this.max_acceleration)
        this.tire_wear = prev.tire_wear + calc_tire_wear(this.max_acceleration)
        if this.is_pit_stop:
            this.gas_usage = 0
            this.tire_wear = 0
            continue
        if this.gas_usage > car.gas_capacity:
            add_pit_stop(track, this.i)
            return False
        if this.tire_wear > car.tire_durability:
            add_pit_stop(track, this.i)
            return False
    return True
```

TEST A CAR

```python
def test_car(car):
    total_score = 0
    for track_num in range(1, 8 + 1):
        time, track = test(car, read_track_n(track_num))
        track.points = map(
            lambda p: Point(acceleration=p.max_acceleration, is_pit_stop=p.is_pit_stop),
            track.points)
        total_score += multipliers[track_num] * time
    return total_score
```