Collision with platforms

Instead of using the entire bounding box of the player and the platform, I will be using an offset to simply check for the player's feet and the top of the platform to make it more efficient and realistic. If the player's arm is above the platform, but its feet are not, the velocity in the vector Y (gravity) will be applied.

Define the top of player's feet by using sprite.getPosition().y + sprite.getGlobalBounds().height - an offset value, and the end of player's feet by using sprite.getPosition().y + sprite.getGlobalBounds().height. Without the offset.

For the top of the platforms, instead of, again, using the entire sprite to check if only the top of it collides with the player, I will tweak the size of its collision box by creating an offset for the collision. As the top of the bounding box starts from the top left, I use sprite.getPosition().y, and for my offset, I will be using a float value.

Using a Boolean function that takes 2 sprites as a reference(&), check if their bounding boxes intersect:

```
    Get player positions x and y, and its feet position.
    Get platform positions x and y, and the tweaked top collision box.
        If intersects:
            Return true;
        Else:
            Return false;
```

Gravity

Call the function above to perform a collision check between the platform and the player's feet, and use a bool to indicate that the player is on the ground:

```
If the function returns false:
    Constantly apply a float value to the player’s vector.y in the update function.
    Gravity += value;
    Vector.y = gravity;
    On_ground = false;
Else
    Gravity = 0;
    On_ground = true;
```

Movement

Every time A or D is pressed, a vector.x is applied to the player, and to smooth it out, an acceleration and deceleration force is applied to the player's movement by adding velocity to the vector.x or multiplying the vector.x by the deceleration. To avoid the player's infinite speed, I'll use a clamp when the current speed equals the maximum speed or when the abstract value of the vector.x is less than 0.

```
If the movement key is not pressed
    vector.x *= deceleration * delta time.
        If the abstract value of vector.x < 0
            vector.x = 0.
Else
    vector.x += speed_value * acceleration * delta time.
        If the abstract value of vector.x  > max_speed
            Vector.x = max_speed.
```

Jump

Because of the collisions with the platforms, the player's float value for gravity is constantly set to 0. To prevent this, if the jump key is pressed, the Player's sprite is slightly moved a few pixels up, and then the inverse gravity for the jump is applied. The curve will occur because the player's velocity x while not equal to zero.

The vector.y is constantly being updated by the gravity value.

```
    Vector.y += 1.0f * gravity


    If on_ground and Space key pressed:
        Move the player up by a small float value:
            Player.setPosition(current_position_x, -4.f)
        On_ground = false.
        Vector.y = -297.f
```