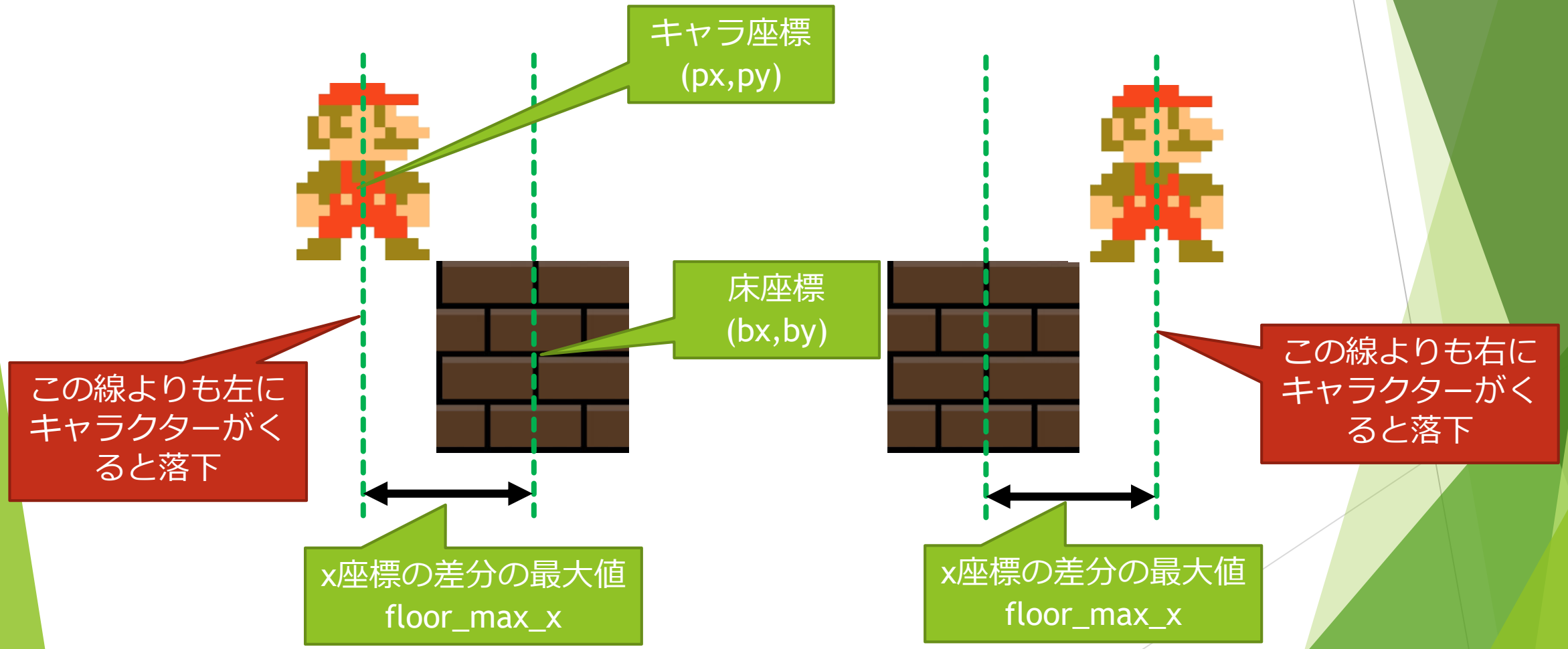
説明のキャラクター座標はオブジェクトの中心で説明

２Ｄアクション
アルゴリズム①

地面着地判定

説明のキャラクター座標はオブジェクトの中心で説明

x方向の着地判定について



説明のキャラクター座標はオブジェクトの中心で説明

x方向に関する判定

- 着地する為には、キャラクターと床のx座標の差分が一定範囲内の必要がある。
- 条件をまとめると・・・・

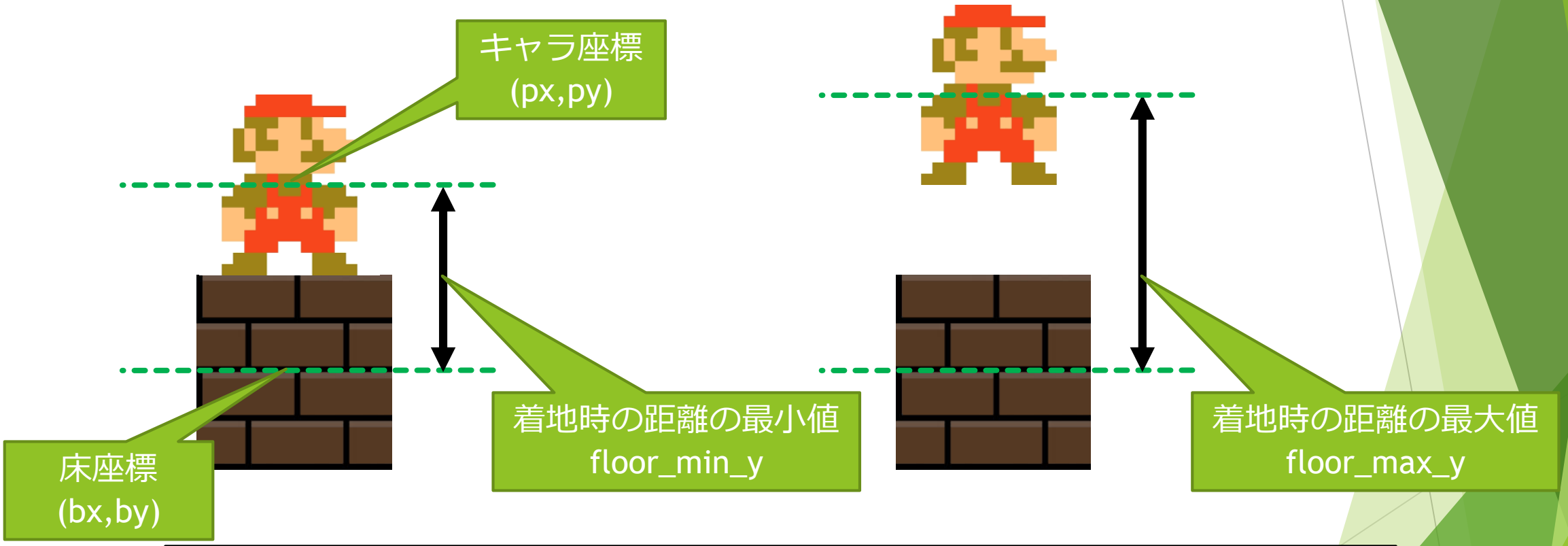
```
bx - x < floor_max_x  &&  x - bx < floor_max_x
```

または、

```
abs(bx - x) < floor_max_x
```

説明のキャラクター座標はオブジェクトの中心で説明

y方向に関する判定




```
bx - y > floor_min_y  &&  by - y <= floor_max_y
```

・floor_max_yが遠すぎると、磁石みたいになるので注意する。
・floor_max_yの値は、落下速度と位置の調整が必要。

説明のキャラクター座標はオブジェクトの中心で説明

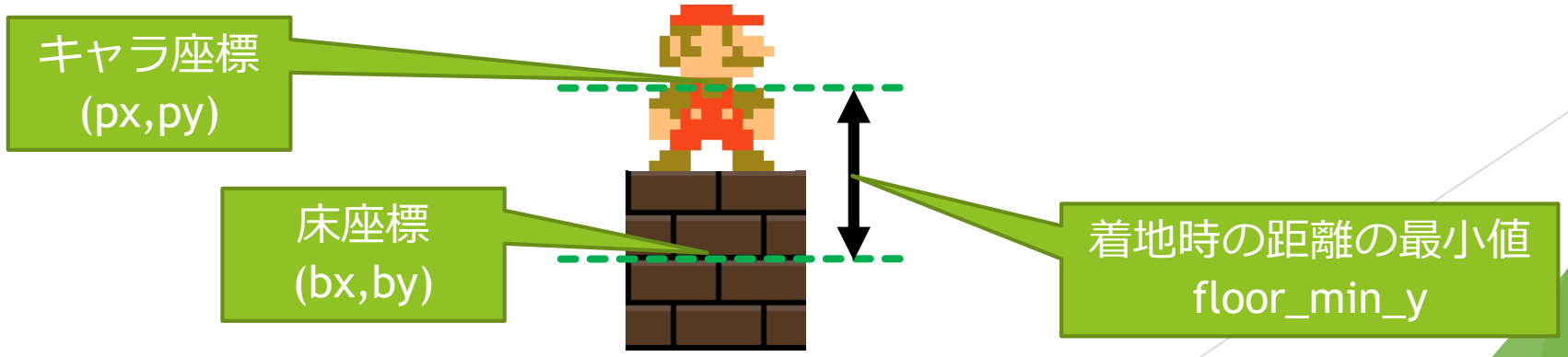
落下スピードと位置の調整　＆　着地

- floor_max_yの値は、落下スピードとの関係を考慮する。
- 突き抜けを防ぐ場合は、落下スピードの大きさをmin_y～max_y以下に制御する必要がある。

```
by - y > floor_min_y && by - y <=floor_max_y
```

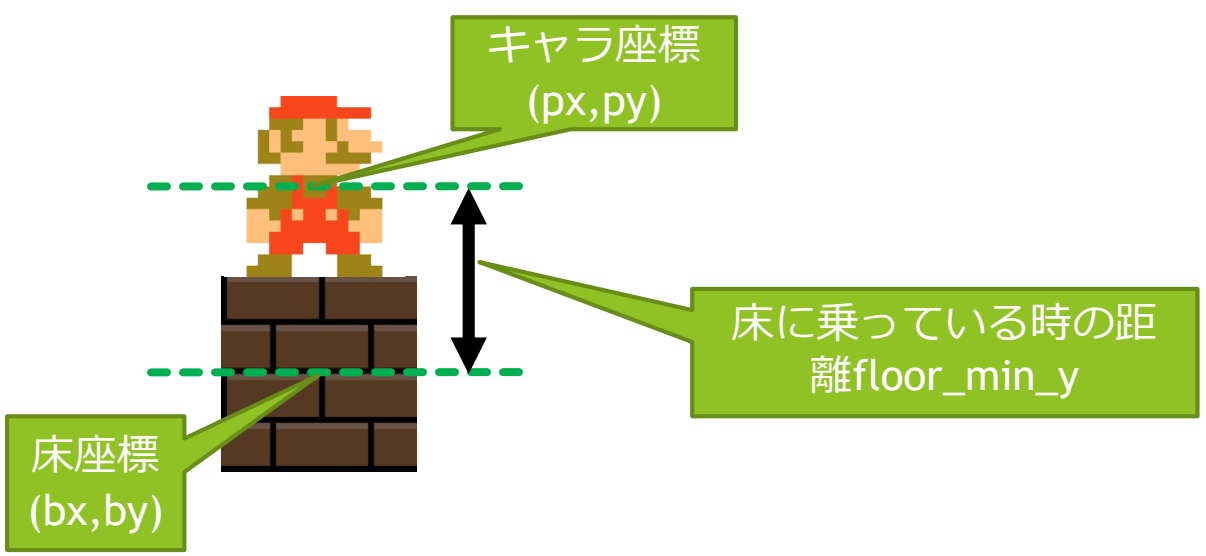
- 着地後の距離は、キャラ座標を(px,py)、ブロックを(bx,by)とすると

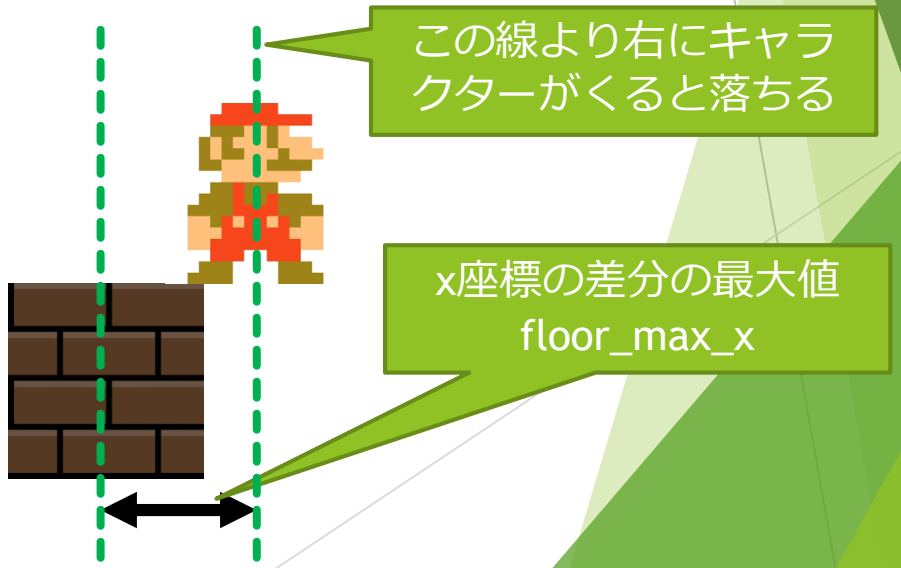
```
py = by - floor_min_y
```



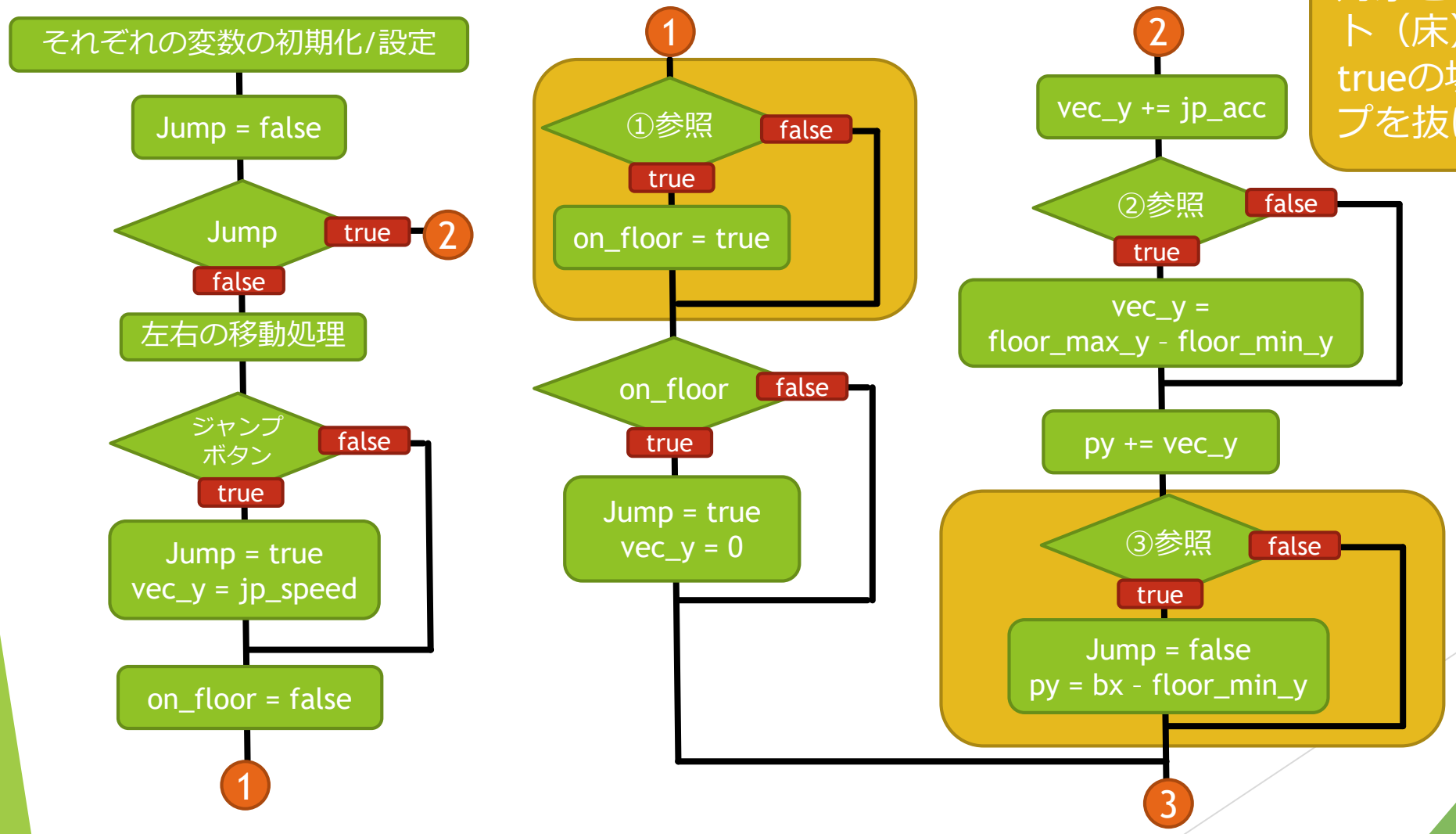
床の端の判定処理

- キャラクターが床に乗っているための条件から考える
- x方向　→　**abs(bx - x) < floor_max_x**　　一定範囲内
- y方向　→　**by - y == floor_min_y**　　差分が一定値
- 上記の二つが成立している場合に、床に乗っているという判定になる。





処理の流れ



対象となっているオブジェクト（床）全てと判定させて、trueの場合は、breakしてループを抜けさせる。

参照類

①
```
abs(bx - px) < floor_max_x &&
by - py == floor_min_y
```

②
```
vec_y > floor_max_y - floor_min_y
```

③
```
abs(bx - px) < floor_max_x &&
by - py > floor_min_y &&
by - py <= floor max_y
```

変数

プレイヤー(px,py)
プレイヤーの移動速度(vec_x,vec_y)
ブロック（床）(bx,by)
ジャンプの初速度 jp_speed
ジャンプの加速度 jp_acc
ジャンプ状態のフラグ Jump
床に乗っているか状態のフラグ on_floor

floor_max_x floor_min_x
floor_max_y floor_min_y
については、各自で調整