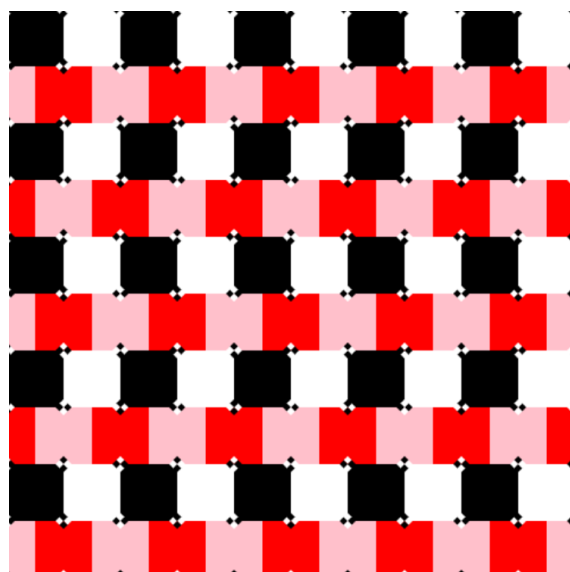


錯視プログラミング②

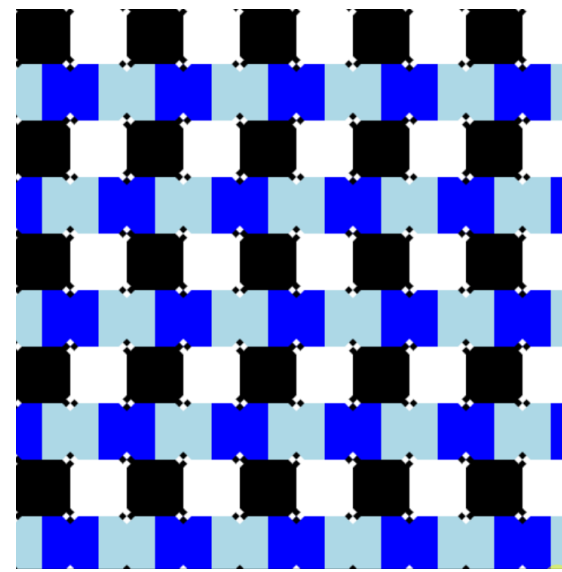
今回やること

- キー入力を使って錯視の色や大きさを変えられるようになる

スペースを押しているとき



スペースを押していないとき



繰り返しの考え方

- 描画エリアは1秒に30回繰り返し実行されている

描画エリア

```
setMainColor("lightblue", "blue");  
setSubColor("black", "white");  
drawIllusion(50, 50, 10);
```

繰り返しの考え方

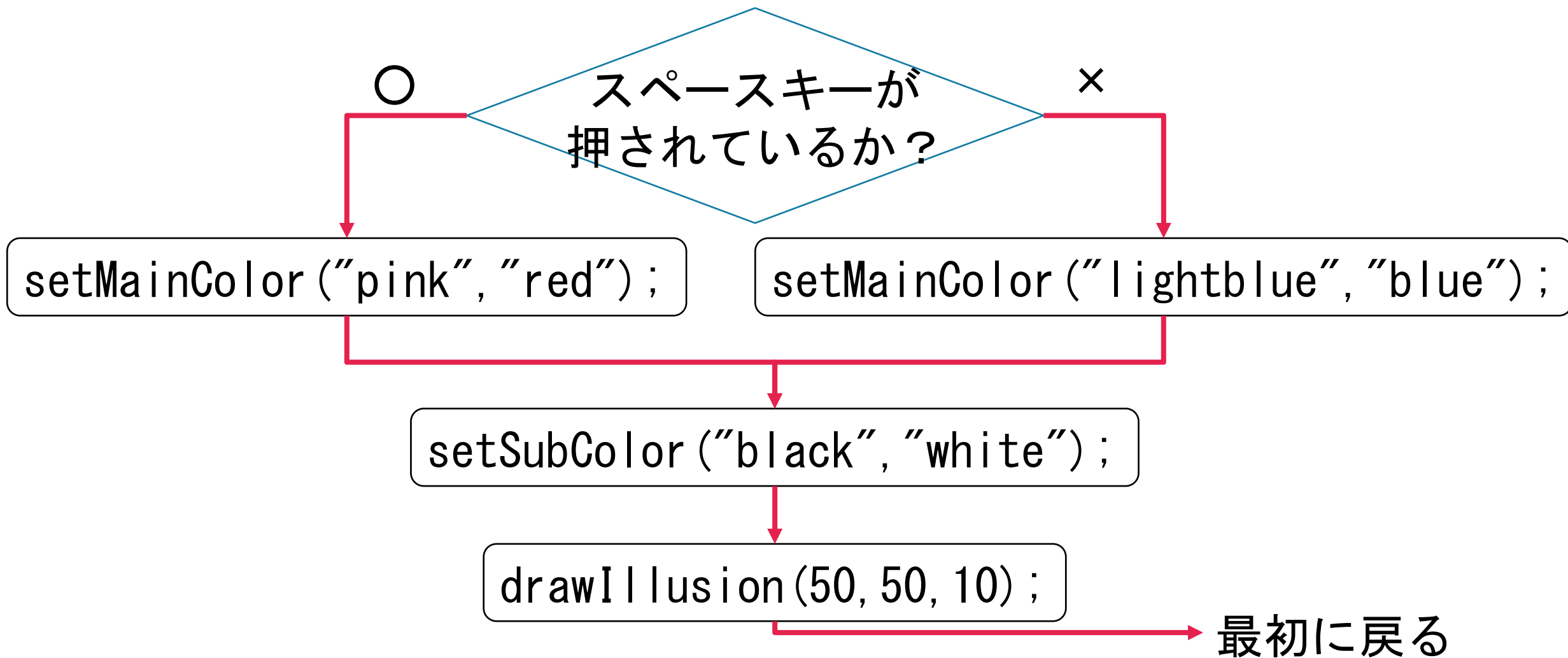
```
setMainColor ("pink", "red");
```

```
setSubColor ("black", "white");
```

```
drawIllusion (50, 50, 10);
```

最初に戻る

分岐の考え方



分岐の考え方(文章)

- ① もしスペースキーが押されているなら、
setMainColor("pink", "red")を実行する。
そうでないのであれば、
setMainColor("lightblue", "blue")を実行する。
- ② setSubColor("black", "white")を実行する。
- ③ drawIllusion(50, 50, 10);を実行する。
- ④ 最初に戻る。

分岐の考え方(コード)

描画エリア

```
if (isPressed(32)) {  
    setMainColor("pink", "red");  
}  
else {  
    setMainColor("lightblue", "blue");  
}  
setSubColor("black", "white");  
drawIllusion(50, 50, 10);
```


(応用) 大きさを変えてみる

- 変数(データを入れる箱)を使って考える

変数定義エリア

```
let x=10;
```

xという名前の箱に10という値を入れる

(応用) 変数の使い方①

変数定義エリア

```
let x=10;
```

描画エリア

```
setMainColor("lightblue", "blue");  
setSubColor("black", "white");  
drawIllusion(50, 50, x);
```

- 変数定義エリアのxの値を使ってdrawIllusionを実行することができる

(応用) 変数の使い方②

変数定義エリア

```
let x=10;
```

描画エリア

```
setMainColor("lightblue", "blue");  
setSubColor("black", "white");  
drawIllusion(50, 50, x);  
x=x+0.1;
```

- 描画エリアが繰り返し実行されるたびにxの値が増えていく
→ アニメーションが作れる！

(応用) 大きさを変えてみる

変数定義エリア

```
let x=10;
```

※drawIllusionの前には
setMainColorとsetSubColorを
忘れずに記述する
(スライドでは省略している)

描画エリア

```
if(isPressed(38)) {  
    x=x+0.1;  
}  
if(isPressed(40)) {  
    x=x-0.1;  
}  
drawIllusion(50, 50, x);
```