

This method determines the winner for a rock, paper, scissors game.

Reduce the lines of code. I can reduce it to 10 lines.

```
public int winner (char p1, char p2)
{
    //returns 1 if player 1 wins, 2 if player 2 wins and 0 if no one wins
    //p1 and p2 hold 'r','p','s' to represent player 1 and 2's choices
1   if (p1 == 'r' && p2 == 'r')
2       return 0;
3   else if (p1 == 'r' && p2 == 's')
4       return 1;
5   else if (p1 == 'r' && p2 == 'p')
6       return 2;
7   else if (p1 == 'p' && p2 == 'r')
8       return 1;
9   else if (p1 == 'p' && p2 == 's')
10      return 2;
11  else if (p1 == 'p' && p2 == 'p')
12      return 0;
13  else if (p1 == 's' && p2 == 'r')
14      return 2;
15  else if (p1 == 's' && p2 == 's')
16      return 0;
17  else if (p1 == 's' && p2 == 'p')
18      return 1;
}
```

Also, see if you can recall what ORATE stands for.

**What
does
ORATE
stand for?**

Organization
Reuseability
Abstraction
Testing
Extensibility



Yesterday, we used methods to cut out repeated code. What three ORATE principles did we apply?

- ① Reuseability
(Organization)
- ② Testing, Extensibility
- ③



Today, we are going to use methods to help with the other two pieces of ORATE.

Organization
Abstraction

What type is each kind of method?

1.
 return type parameters

public boolean **win()** B

Returns true if certain conditions are met

2.
 Returns true if the word matches the answer

public boolean **winRiddle** (**String** word) { D }

3.
 Toasts either player 1 or 2, based on who wins

public void **rockPaperScissorsWin**(**char** p1, **char** p2) C

4.
 Toasts either X or O, based on who wins

public void **win()** A

A: No return,
No Parameters

B: Return,
No Parameters

C: No return,
Parameters

D: Return,
Parameters

Call each method.

parameterType parameter = value;
returnType variable = **methodName** (parameter);

1.

public boolean win()

boolean ans = win();



another way to call it:

if (win())



2.

public boolean winRiddle (**String** word){

String word = input.getText().toString();

3. boolean ans = winRiddle(word);

public void rockPaperScissorsWin(**char** p1, **char** p2)

char p1 = 'r'; char p2 = 'p';

4. rockPaperScissorsWin(p1, p2);

public void win()

win();

Identify the method name, return type, parameter type, parameter name, method signature.

What sort of method is rollDice?

	No Parameters	Some Parameters
No Return Type	A	C
Return Type	B	D

RT MN PT PN ← signature

```
public int rollDice (ImageView d) {  
    int num = (int) (Math.random() * 6) + 1;  
    if (num==1)  
        d.setImageResource (R.drawable.one);  
    else if (num==2)  
        d.setImageResource (R.drawable.two);  
    else if (num==3)  
        d.setImageResource (R.drawable.three);  
    else if (num==4)  
        d.setImageResource (R.drawable.four);  
    else if (num==5)  
        d.setImageResource (R.drawable.five);  
    else  
        d.setImageResource (R.drawable.six);  
    return num;  
}
```

Call this
method in the
button's
onClick

```
public int rollDice(ImageView d) {  
    parameterType parameter = value;  
    returnType variable = methodName (parameter);
```

```
int d1=1; ←  
int d2=6; ←
```

Green is how to
switch it for a second dice

```
public void dice1Click(View view) {  
    ImageView d = (ImageView) findViewById(R.id.dice1);  
    d1=rollDice(d);  
    if (d1==d2) {  
        Toast.makeText(getApplicationContext(), "You win!", Toast.LENGTH_SHORT).show();  
    }  
}
```

Let's look at this method:

3. First, code the win methods. Then, using the method signatures, call the methods.
(a) Method:

```
public boolean win(){  
    //returns true if 5 points (global variable), false otherwise  
    if (points >= 5)  
        return true;  
    else  
        return false;  
}
```

Call:

if (win()) boolean ans = win();
if (ans == true)
 Toast.makeText(getApplicationContext(), "Win!",
 Toast.LENGTH_SHORT).show();

So how do we reduce the lines of code on this method?

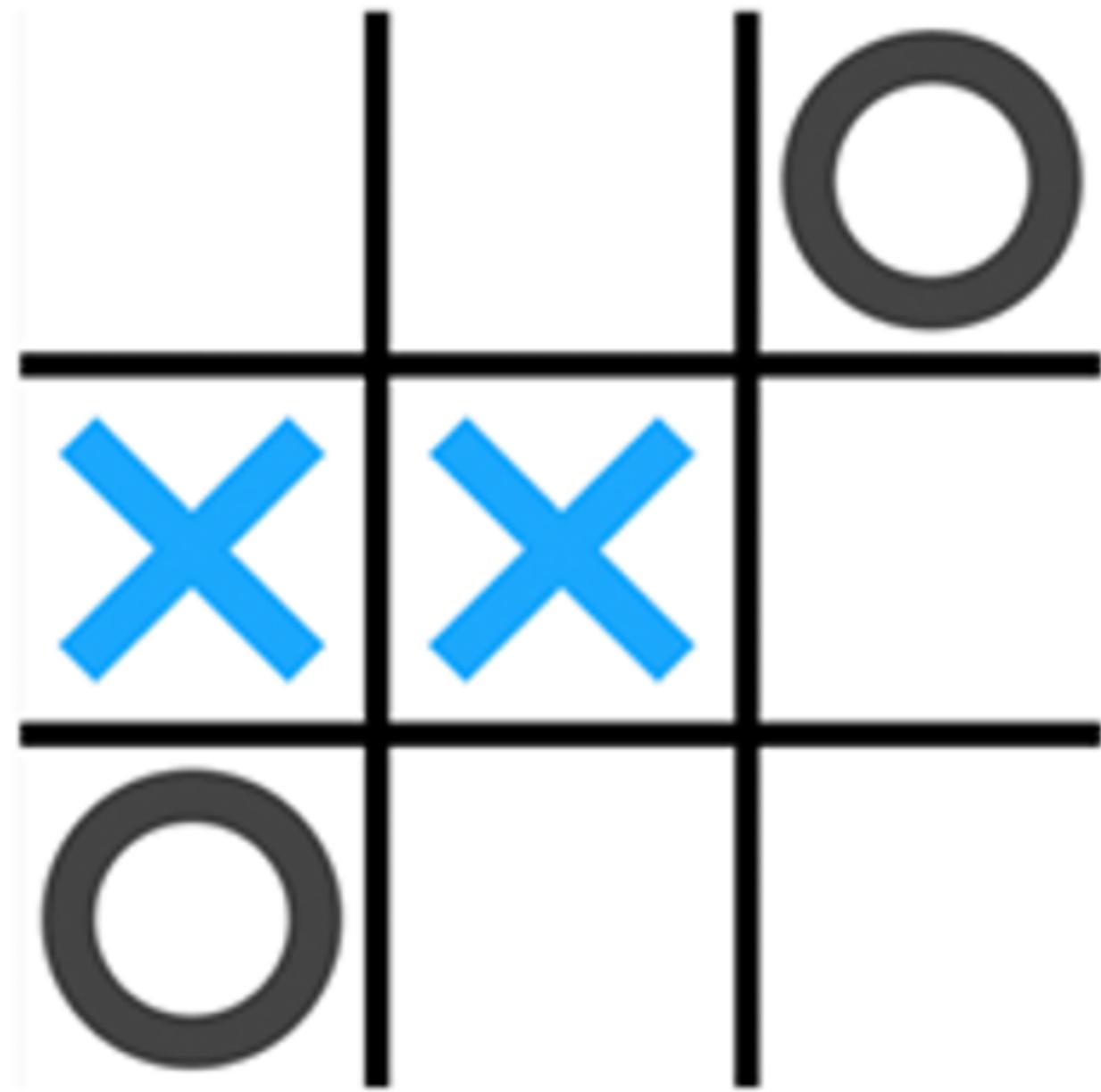
```
public int winner (char p1, char p2)
{
    //returns 1 if player 1 wins, 2 if player 2 wins and 0 if no one wins
    //p1 and p2 hold 'r','p','s' to represent player 1 and 2's choices
1   if (p1 == 'r' && p2 == 'r') ← if (p1 == p2)
2       return 0;           return 0;
3   else if (p1 == 'r' && p2 == 's')
4       return 1;
5   else if (p1 == 'r' && p2 == 'p')
6       return 2;
7   else if (p1 == 'p' && p2 == 'r')
8       return 1;
9   else if (p1 == 'p' && p2 == 's')
10    return 2;
11  else if (p1 == 'p' && p2 == 'p') ←
12      return 0;
13  else if (p1 == 's' && p2 == 'r')
14      return 2;
15  else if (p1 == 's' && p2 == 's') ←
16      return 0;
17  else if (p1 == 's' && p2 == 'p')
18      return 1;
}
```

else
return 2;

4. This method takes two chars, one for each player and using Rock, Paper, Scissors rules, displays a message for the winner. Fill in the blanks to make the method functional.

```
public void rockPaperScissorsWin(char p1, char p2) {  
    int winner = 2;  
    if (p1 == 'r' && p2 == 'S')  
        winner = 1;  
    else if (p1 == 's' && p2 == 'P')  
        winner = 1;  
    else if (p1 == 'p' && p2 == 'R')  
        winner = 1;  
    else if (p1 == p2) } {the tie  
        winner = 0;  
    if (winner == 1) {  
        Toast.makeText(getApplicationContext(), "Player 1 Wins", Toast.LENGTH_SHORT).show();  
    } else if (winner == 2) {  
        Toast.makeText(getApplicationContext(), "Player 2 Wins", Toast.LENGTH_SHORT).show();  
    } else {  
        Toast.makeText(getApplicationContext(), "Tie", Toast.LENGTH_SHORT).show();  
    }  
}
```





We need
8 Boolean
expressions
to check
a win

Tic Tac Toe
Win

[0][0]	[0][1]	[0][2]
[1][0]	[1][1]	[1][2]
[2][0]	[2][1]	[2][2]

board[0][0] == board[0][1]

```

public void win() {
    //board [x][y] holds 0 if empty; 1 if X holds square, 2 if O holds square
    int winner = 0;
    if (board[0][0] == board[0][1] && board[0][0] == board[0][2] && board[0][0] != 0)
        winner = board[0][0];
    else if (board[1][0] == board[1][1] && board[1][0] == board[1][2] && board[1][0] != 0)
        winner = board[1][0];
    else if (board[2][0] == board[2][1] && board[2][0] == board[2][2] && board[2][0] != 0)
        winner = board[2][0];
    else if (board[0][0] == board[1][0] && board[0][0] == board[2][0] && board[0][0] != 0)
        winner = board[0][0];
    else if (board[0][1] == board[1][1] && board[0][1] == board[2][1] && board[0][1] != 0)
        winner = board[0][1];
    else if (board[0][2] == board[1][2] && board[0][2] == board[2][2] && board[0][2] != 0)
        winner = board[0][2];
    else if (board[0][0] == board[1][1] && board[0][1] == board[2][0] && board[0][0] != 0)
        winner = board[1][1];
    else if (board[0][2] == board[1][1] && board[0][1] == board[2][2] && board[0][2] != 0)
        winner = board[0][2];
    //cat's game
    else if (board[0][0] != 0 && board[0][1] != 0 && board[0][2] != 0 &&
            board[1][0] != 0 && board[1][1] != 0 && board[1][2] != 0 &&
            board[2][0] != 0 && board[2][1] != 0 && board[2][2] != 0)
        winner = 3;
    if (winner == 1) {
        Toast.makeText(getApplicationContext(), "X wins", Toast.LENGTH_SHORT).show();
    } else if (winner == 2) {
        Toast.makeText(getApplicationContext(), "O wins", Toast.LENGTH_SHORT).show();
    } else if (winner == 3) {
        Toast.makeText(getApplicationContext(), "Cat's game", Toast.LENGTH_SHORT).show();
    }
}

```

[0][0]	[0][1]	[0][2]
[1][0]	[1][1]	[1][2]
[2][0]	[2][1]	[2][2]