For Loop:

When we did the dumb 8 queens, we used 8 nested for loops.

If we need to find all possible combination for 2 columns, we have to use 2 nested for loops.
- For each value in first column we have another set of values for the second column.
For Loop:

If we need to find all possible combination for 3 columns, we have to use 3 nested for loops.

If we need to find all possible combination for 4 columns, we have to use 4 nested for loops.

If we need to find all possible combination for n columns, we have to use n nested for loops.
Recursion:

What can we do if we need to work with $n$ nested for loops?

- Recursion
Recursion - For Loop:

Base Case:
- Only one column

for (int i = 0; i < size; i++){
    array[size - 1] = i;
}

Why is size - 1? Why is not 0?
Recursion - For Loop:

```c
for (int i = 0; i < size; i++){
    array[column] = i;
    loop_rec(array, size, column + 1);
}
```
Recursion - For Loop:

```cpp
void print(int a[], int size){
    for ( int i = 0; i < size; i++)
        cout << a[ i ] << ((i == size - 1)? "\n" : " " );
}
void loop_rec(int array[ ], int size, int column = 0) {
    if ( size == column + 1) {
        for (int i = 0; i < size; i++)
            array[size - 1] = i;
        print(array, size);
        return;
    }
    for (int i = 0; i < size; i++) {
        array[column] = i;
        loop_rec(array, size, column + 1);
    }
}
```
Conditional operator:

Select between two options based upon a condition. Unlike if, it can do it inline.

Ex:

```c
int x = (1==0) ? 1 : 0;
```

--

Page 8
Conditional operator:

```cpp
for ( int c = 0; c < size; c++)
{
    cout << array[c] << " ";
}
cout << endl;

for ( int c = 0; c < size; c++)
    cout << array[c] << ( (c == size - 1) ? "\n": " ");
```
Conditional operator:

Pass in a select value to function

bool check = true;

func(check?!check:check);

func(check?check:check+1);