## Branching

Branching is a fundamental part of programming. It makes decision to take one path or another. Through the `if` statement, your programs can branch to a section of code or skip taking any action.

```
If condition:
   statement
   statement
elif condition:
   statement
else:
   statement
```

https://github.com/JasonWang19/cacc_python/blob/master/week3/Conditional.py

## Blocks

By indenting the line, it becomes a *block*. A block is one or more consecutive lines indented by the same amount. Indenting sets lines off not only visually, but logically too. Together, they form a single unit.

## Loop

Repeatedly do something until some condition doesn't meet

```
while condition:
   statement
   statement
```

https://github.com/JasonWang19/cacc_python/blob/master/week3/Loop.py

## What Value Treated As False

short general explanation: Almost everything is converted to `True`. The only values among built-in types that are interpreted as False are

- The Boolean value `False` itself
- Any numerical value equal to 0 (0, 0.0 but not 2 or -3.1)
- The special value `None`
- Any empty sequence or collection, including the empty string (`''`, but not `'0'` or `'hi'` or `'False')` and the empty list (`[]`, but not `[1,2, 3]` or `[0]`)

## Break and continue

You can use `break` and `continue` in any loop you create. They aren't just restricted for use in intentional infinite loops.

## Compound Conditions

and, or, ^, not

## Practices

1. The code should request user to input an integer, if the integer is odd, print 10 odd integers after the input integer, if even, print 10 even integers after the input. Implement in at least two ways.
2. Same problem as 1, but if input is even, print 5 even integers.
3. Write a program that flips a coin 100 times and then tells you the number of heads and tails. (need to refer to https://docs.python.org/3/library/random.html
   `random.`**`randint`**(*a*, *b*))