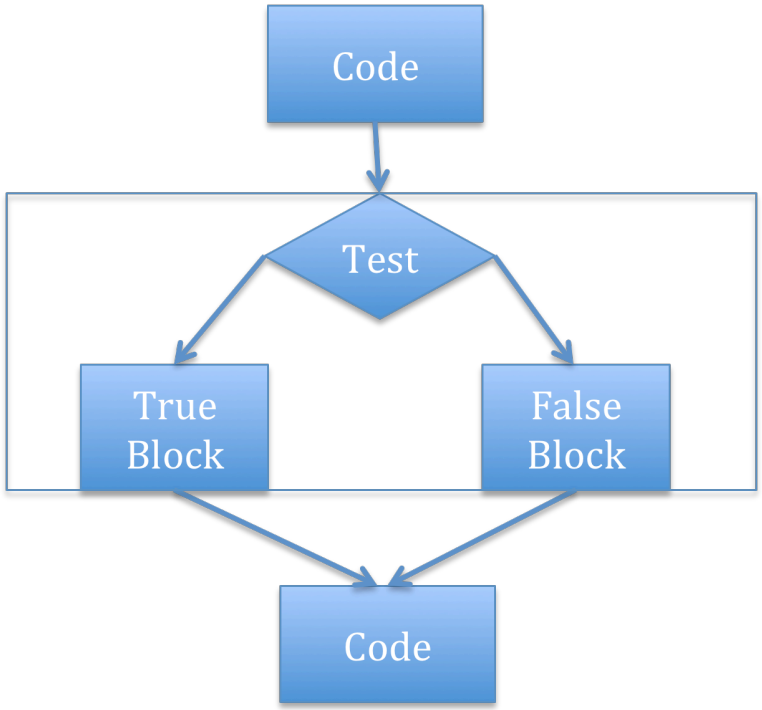
Branching programs

- The simplest branching statement is a **conditional**
  - A test (expression that evaluates to `True` or `False`)
  - A block of code to execute if the test is `True`
  - An optional block of code to execute if the test is `False`



A simple example

```
x = int(raw_input('Enter an integer: '))
if x%2 == 0:
    print(‘’)
    print('Even')
else:
    print(‘’)
    print('Odd')
print(’Done with conditional')
```

indentation.

Some observations

- The expression `x%2 == 0` evaluates to `True` when the remainder of `x` divided by `2` is 0
- Note that `==` is used for comparison, since `=` is reserved for assignment
- The indentation is important – each indented set of expressions denotes a block of instructions
  - For example, if the last statement were indented, it would be executed as part of the `else` block of code
- Note how this indentation provides a visual structure that reflects the semantic structure of the program

We can have nested conditionals

```
if x%2 == 0:
    if x%3 == 0:
        print('Divisible by 2 and 3')
    else:
        print('Divisible by 2 and not by 3')
elif x%3 == 0:
    print('Divisible by 3 and not by 2')
```

else, 一定 elif 在前

And we can use compound Booleans

```
if x < y and x < z:
    print('x is least’)
elif y < z:
    print('y is least’)
else:
    print('z is least’)
```

What have we added?

- Branching programs allow us to make choices and do different things
- But still the case that at most, each statement gets executed once.
- So maximum time to run the program depends only on the length of the program
- These programs run in **constant time**