ECS 10

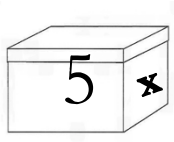
1/16

## Announcements

- Second program due tomorrow night.

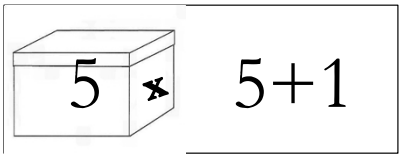
## Variables

```
x = 5
x = x+1
```



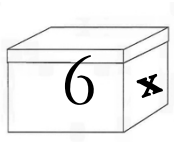

## Re-assigning variable

```
x = 5
x = x+1
```




## Re-assigning variable

```
x = 5
x = x+1
```




## Boolean expressions

```
today == "y"
```

- Values of Boolean expressions are either True or False
- We now have four data types (and four kinds of expressions):

  integer, floating point, string and Boolean.

## Boolean algebra

- Named after George Boole (1815-1864)
- Main idea: you can write down logic as mathematical formulas.
- His book: *An Investigation into the Laws of Thought*
- Computers do lots of logical as well as numerical and string computation.

## True and False

- These are the only two possible Boolean data items.
- NOT descriptions of expressions; actual data.
- Maybe better names would be Truth and Falsehood, so we think of them as **things**.
- We can store True and False in variables, just like other data.

## Boolean variable

- Have value **True** or **False**

```
correct = (answer == "y")
if correct:
        score = score+1
```

## Not!

- The not command changes True to False and False to True

```
if not canBeInt(guess):
    print("That is not a valid guess.")
```

## not

- not is a Boolean operator
- The value of a Boolean operation is Boolean

```
>>> s = 5
>>> s > 10
False
>>> not s > 10
True
```

## and and or

- and and or are Boolean operators

```
>>> 5+6
11
>>> (2<3) and (1<3)
True
```

## and

```
>>> True and True
True
>>> True and False
False
>>> False and True
False
>>> False and False
False
```

## or

```
>>> True or True
True
>>> True or False
True
>>> False or True
True
>>> False or False
False
```

Memorize this!

## Booleans simplify this program

```
if today == "y":
    if yesterday == "y":
        print "Doin' good!"
    else:
        print "Try harder!"
else:
    print "Try harder!"
```

## Using Booleans

```
if today == "y“ and yesterday == “y”:
    print "Doin' good!"
else:
    print "Try harder!"
```

## Program as a function

```
def main():
    name = input("Enter name: ")
    print("Hi,",name,"!")

main()
```

- Works exactly the same as:

```
name = input("Enter name: ")
print("Hi,",name,"!")
```

## Function definition

```
def main():
    name = input("Enter name: ")
    print("Hi,",name,"!")

main()
```

- This part defines the function main().
- The block under the def statement is executed whenever the program calls main().

## Calling the function

```
def main():
    name = input("Enter name: ")
    print("Hi,",name,"!")

main()
```

- The program itself is just this one statement, which calls a function called main() .
- Here main has no input, so the parentheses are empty.

## The return statement

- The return statement stops a function early.
- If the program gets to a return statement, the function is over.
- This lets you skip the rest of the function.

- How can you use this in the quiz program?