Objectives

- A few tricks
- Definite Loops

Review

- How do we make our programs interact with a user?

Trick #1: Type Conversion

- You can convert a variable's type
  - Use the type's ***constructor***

Conversion Function/Constructor	Example	Value Returned
int(<number or string>)	int(3.77) int("33")	3 33
float(<number or string>)	float(22)	22.0
str(<any value>)	str(99)	"99"

Example Using Type Conversion

- May want to restrict the type of values that a user enters
- For example, a user's age should be an integer

```
orig_age = input("What is your age? ")
int_age = int(orig_age)
print("Your age is", int_age)
```

Converts age to an integer

Ideally, we'd tell the user that we made a change to their input, but we don't know how to do that yet.

Another Example: Restricting User's Inputs

```
>>> x = 7
>>> yourVal = input("My val is: ")
My val is: x
>>> print(yourVal)
x
```


Another Example: Restricting User's Inputs

```
>>> x = 7
>>> yourVal = input("My val is: ")
My val is: x
>>> print(yourVal)
x
>>> yourVal = eval(input("My val is: "))
My val is: x
>>> print(yourVal)
7
>>> yourVal = int(input("My val is: "))
My val is: x
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10:
'x'
```

What happened here?

Trick #2: Arithmetic Shorthands

- Called **extended assignment operators**
- Increment Operator
  - `x = x + 1` can be written as **`x += 1`**
- Decrement Operator
  - `x = x - 1` can be written as **`x -= 1`**
- Shorthands are similar for `*, /, //` :
  - `amount *= 1.055`
  - `x //= 2`

Review

- How do we make our programs do something multiple times?
- What are the syntax and semantics of the for loop?

Review Solutions

- What will the output from each of these programs be?

```
for x in range(3):
    print("You say 'hello'")
    print("And, I say 'goodbye'...")
```

```
for x in range(3):
   doubled = x * 2
   print(x, "*2 =", doubled)
```


Review Solutions

- What will the output from each of these programs be?

x takes on values 0, 1, 2 in both cases

```
for x in range(3):
    print("You say 'hello'")
    print("And, I say 'goodbye'...")
```

x was not used in loop explicitly-- just used for making loop "go"

```
for x in range(3):
   doubled = x * 2
   print(x, "*2 =", doubled)
```

x was used in loop explicitly

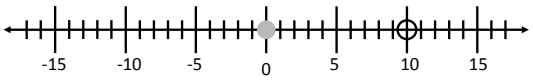
Depends on problem you're solving

`range([start,] stop[, step])`

- 1 argument: `range(stop)`
  - Defaults: `start` = 0, `step` = 1
  - Iterates from 0 to `stop`-1 with `step` size=1
- 2 arguments: `range(start, stop)`
  - Default: `step` = 1
  - Iterates from `start` to `stop`-1 with `step` size=1
- 3 arguments: `range(start, stop, step)`
  - Iterates from `start` to `stop`-1 with `step` size=`step`

`range`

- `range` is a number generator
  - 1 argument: `range(stop)`
  - 2 arguments: `range(start, stop)`
  - 3 arguments: `range(start, stop, step)`

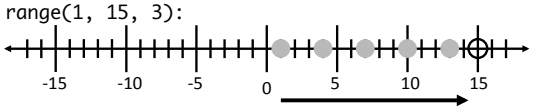


[start, stop)

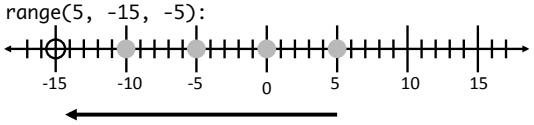
`range(10)`
`range(0,10)`
`range(0,10,1)`

Sequence generated by range

range(1, 15, 3):



range(5, -15, -5):



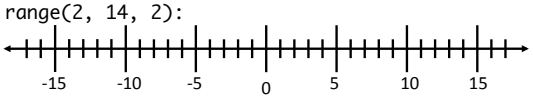
more_range_examples.py

Practice

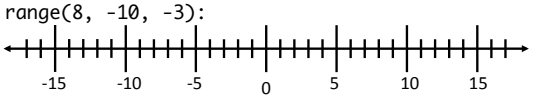
Place these: ● ○

Which direction?

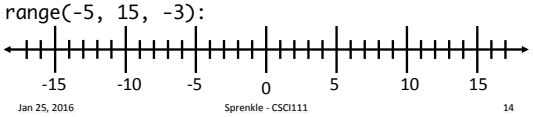
range(2, 14, 2):



range(8, -10, -3):

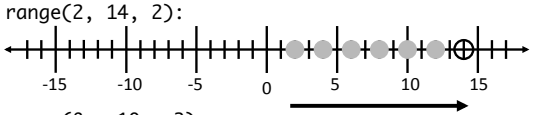


range(-5, 15, -3):

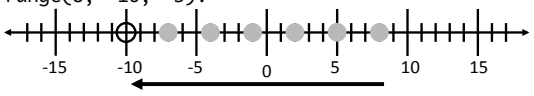


Practice Solution

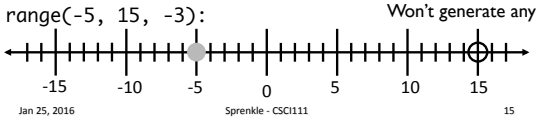
range(2, 14, 2):



range(8, -10, -3):



range(-5, 15, -3): Won't generate any



Practicing for Loops

- Write the Python code to print the following:
  - A)
    1
    2
    3
    4
    5
  - B)
    2
    5
    8
    11
  - C)
    ****
    ****
    ****

What is getting repeated? How many times?

Programming Practice

- Add 5 numbers, inputted by the user
  - After implementing, simulate running on computer

sum5.py

Generalizing Solution: Accumulator Design Pattern

1. Initialize accumulator variable
2. Loop until done
   - Update the value of the accumulator
3. Display result

Programming Practice

- Average 5 numbers inputted by the user

Programming Practice

- Average 5 numbers inputted by the user
- Good example of how to build up to a solution
  - Break down into smaller pieces

This Week

- Lab 2
- Broader Issue: Web Searches