CS Bridge, Lecture 4

For Loops Deconstructed

For Loop Redux

Create a counting variable i

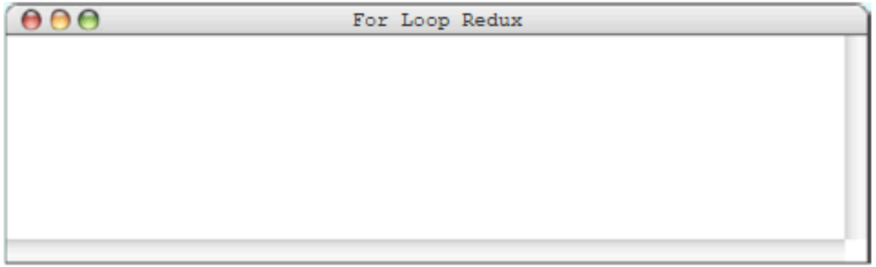
Which takes on the values 0 to 99 one at a time

```
for i in range(100):
    print("Python rocks socks!")
```

For Loop Redux

range(3) -> 0, 1, 2

```
for i in range(3):
   print(“Python rocks socks!”)
```

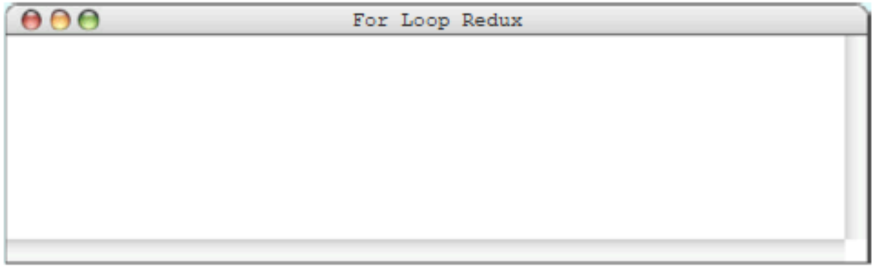


For Loop Redux

i $\boxed{0}$

range(3) -> 0, 1, 2

```
for i in range(3):
   print("Python rocks socks!")
```

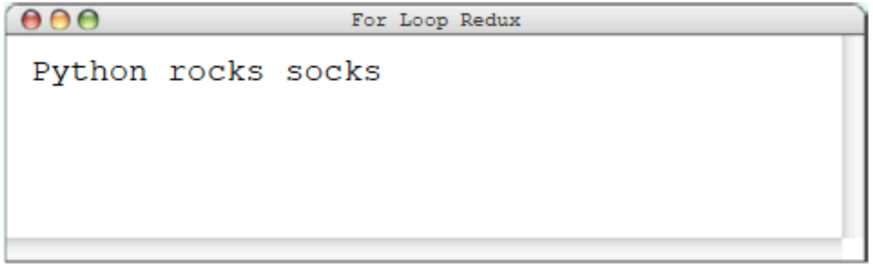


For Loop Redux

```
i  0
```

range(3) -> 0, 1, 2

```
for i in range(3):
    print("Python rocks socks!")
```



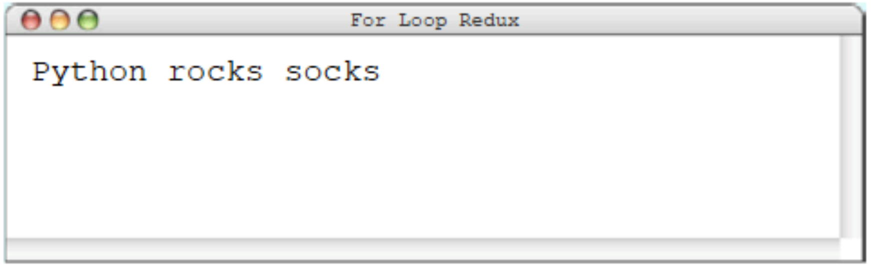
```
Python rocks socks
```

For Loop Redux

```
i  1
```

range(3) -> 0, 1, 2

```
for i in range(3):
    print("Python rocks socks!")
```



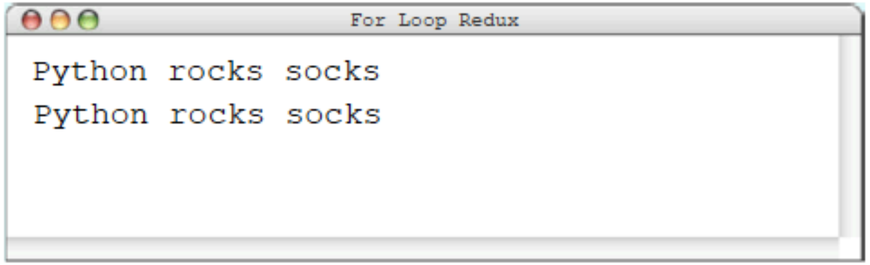
```
Python rocks socks
```

For Loop Redux

i $\boxed{1}$

range(3) -> 0, 1, 2

```
for i in range(3):
  print("Python rocks socks!")
```



For Loop Redux

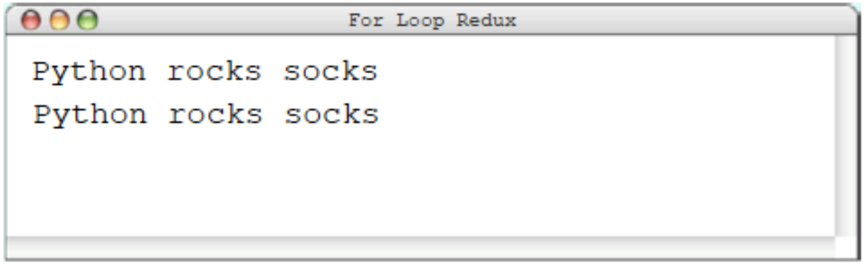
```
Python rocks socks
Python rocks socks
```

For Loop Redux

i $\boxed{2}$

range(3) -> 0, 1, 2

```
for i in range(3):
    print("Python rocks socks!")
```



For Loop Redux

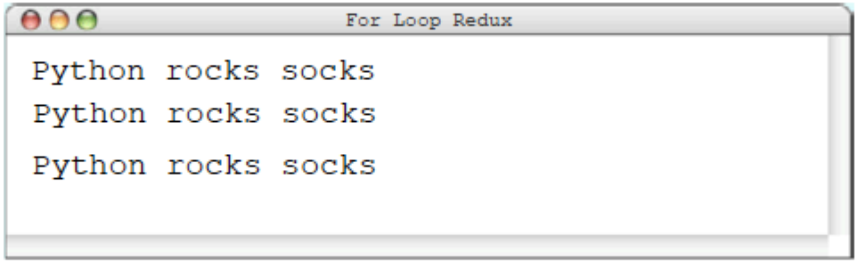
```
Python rocks socks
Python rocks socks
```

For Loop Redux

i $\boxed{2}$

range(3) -> 0, 1, 2

```
for i in range(3):
  print("Python rocks socks!")
```


For Loop Redux

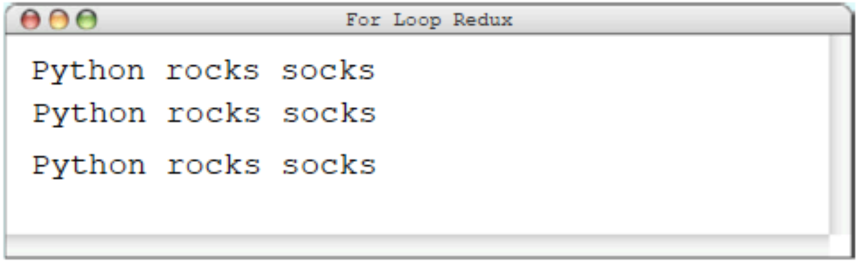
```
Python rocks socks
Python rocks socks
Python rocks socks
```

For Loop Redux

i 2

range(3) -> 0, 1, 2

```
for i in range(3):
    print(“Python rocks socks!”)
```


For Loop Redux

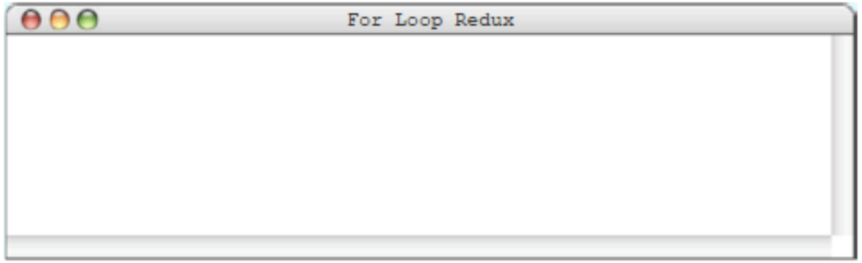
```
Python rocks socks
Python rocks socks
Python rocks socks
```

END OF FOR LOOP

WE CAN USE THE FOR LOOP VARIABLE

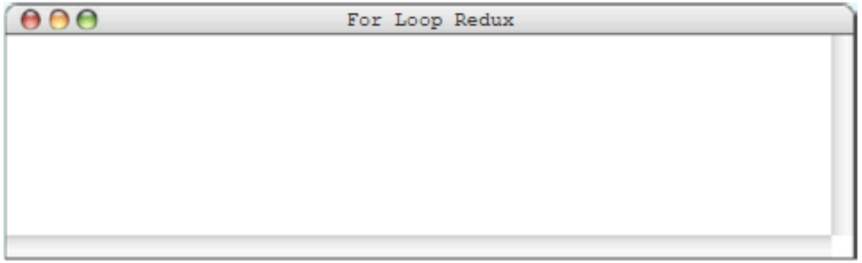
Printing Even Numbers

```
for i in range(3):
   print(i * 2)
```



Printing Even Numbers

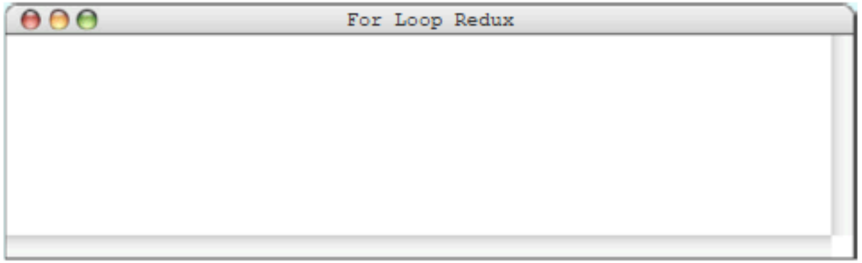
```
for i in range(3):
    print(i * 2)
```



Printing Even Numbers

i $\boxed{0}$

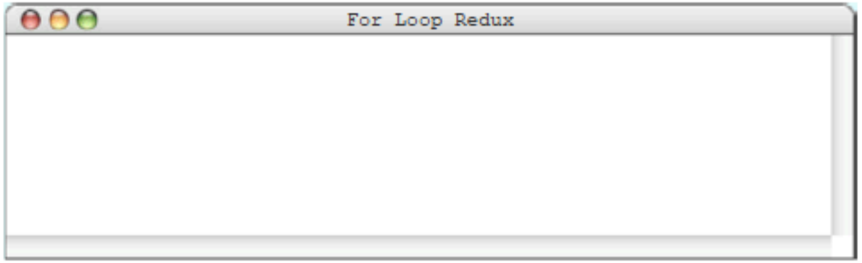
```
for i in range(3):
    print(i * 2)
```



Printing Even Numbers

```
i  0
```

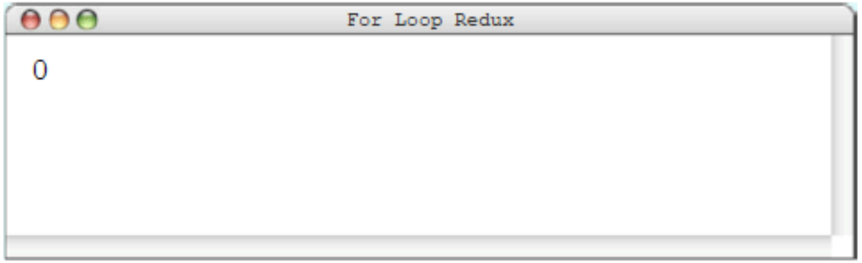
```
for i in range(3):
    print(i * 2)
```



Printing Even Numbers

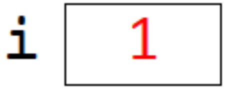
i $\boxed{0}$

```
for i in range(3):
    print(i * 2)
```

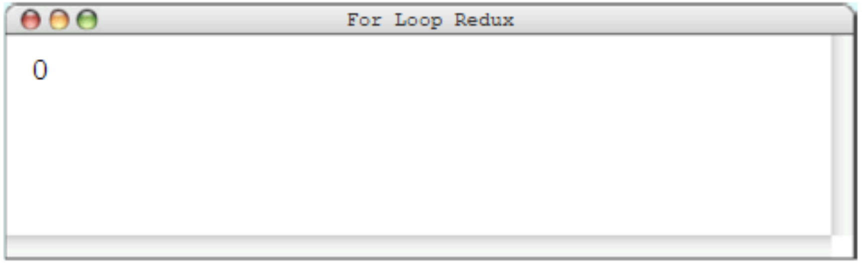


```
0
```

Printing Even Numbers



```
for i in range(3):
    print(i * 2)
```

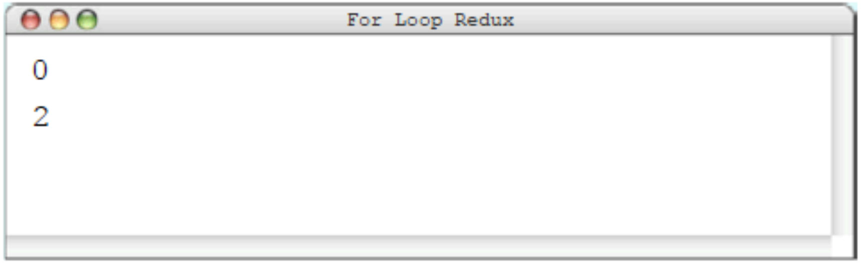


```
0
```

Printing Even Numbers

```
i  1
```

```
for i in range(3):
    print(i * 2)
```

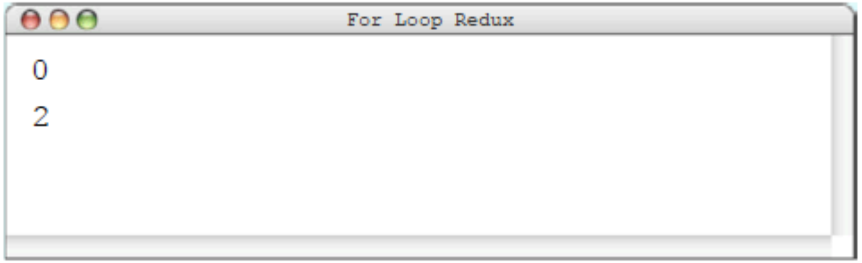


```
0
2
```

Printing Even Numbers

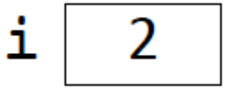
i 2

```
for i in range(3):
    print(i * 2)
```

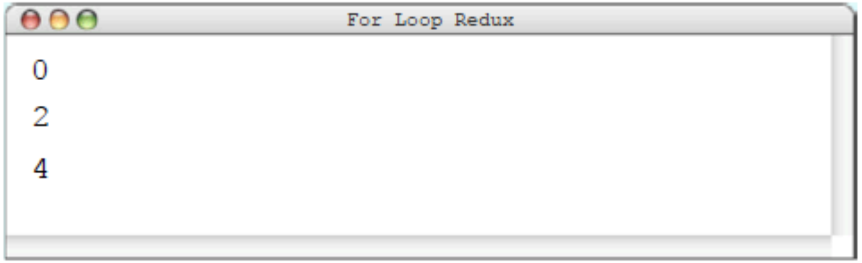


```
0
2
```

Printing Even Numbers



```
for i in range(3):
    print(i * 2)
```



```
0
2
4
```

Printing even numbers

```
# our solution
for i in range(3):
   print(i * 2)
```

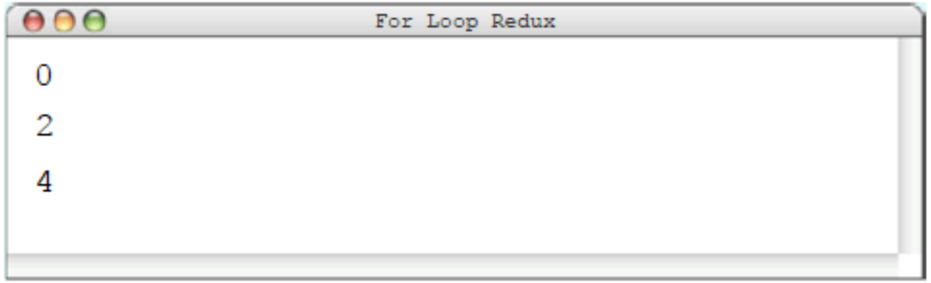
0, 1, 2

Start at 0

Stop before 6

```
# equivalently
for i in range(0, 6, 2):
   print(i)
```

Skip by 2 each time

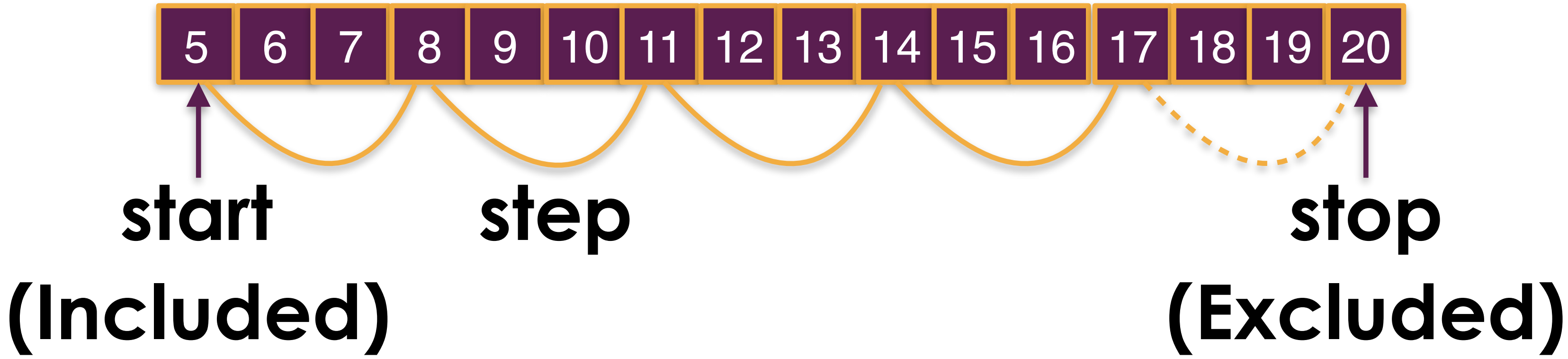


```
0
2
4
```

More on range()

range(start, stop, step)

range(5, 20, 3)



More on range()

Experiments with range(start, stop, step)

Excerpt from Lecture6/range_example.py

```
for i in range(...):
    print(i, end=' ')
```

```
range(5)            ->    0 1 2 3 4
range(5, 10)        ->    5 6 7 8 9
range(6, 15, 3)     ->    6 9 12
range(15, 6, -3)    ->   15 12 9
```

For loop exercises

Let’s write a program that outputs all numbers divisible by 5 in a user defined range (a minimum number and a maximum number)

If the minimum value entered is larger than the maximum value entered, your program should swap them and use.

Sample run:

```
Specify the minimum value:63
Specify the maximum value:39
Your minimum value was bigger than max value
I'll swap them for you
Min-value:39, max-value: 63
40 is divisible by 5
45 is divisible by 5
50 is divisible by 5
55 is divisible by 5
60 is divisible by 5
```

Divisors and prime numbers

Write a program that outputs divisors of all numbers in range [50, 60]. The program should print “is a prime number" if there are no divisor found except 1 and the number itself. Expected output:

```
50:  2  5  10  25
51:  3  17
52:  2  4  13  26
53:  is a prime number
54:  2  3  6  9  18  27
55:  5  11
56:  2  4  7  8  14  28
57:  3  19
58:  2  29
59:  is a prime number
60:  2  3  4  5  6  10  12  15  20  30
```

Creating number combinations

```
def main():
    for i in range(2):
        for j in range(2):
            for k in range(2):
                for m in range(2):
                    print(str(i) + str(j) + str(k) + str(m))
```

```
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
```

Keep the balance

I have a factory that runs with 100 people.

Some people get paid 500 units/month, some 100 units/month, and some 5 units/month.

I pay 10000 units/month to my workers.

How many of the 100 receive 5 units/month?

Could you help me with a Python program?

Last example with while()

Write a program that computes the sum of all digits of an integer read from the user. Your program should continue asking user input as long as the integer specified is positive.

Sample run:

```
Enter a positive integer: 1234
Sum of all digits: 10
Enter a positive integer: 80009
Sum of all digits: 17
Enter a positive integer: 101010101
Sum of all digits: 5
Enter a positive integer: -5
BYE
```