Loops

Dr. Baldassano

chrisb@princeton.edu

Yu's Elite Education

Last week recap

- if – elif – else
- Relational operators (<, <=, ==, …)
- Logical operators (not, and, or)
- Blackjack problem

Loops

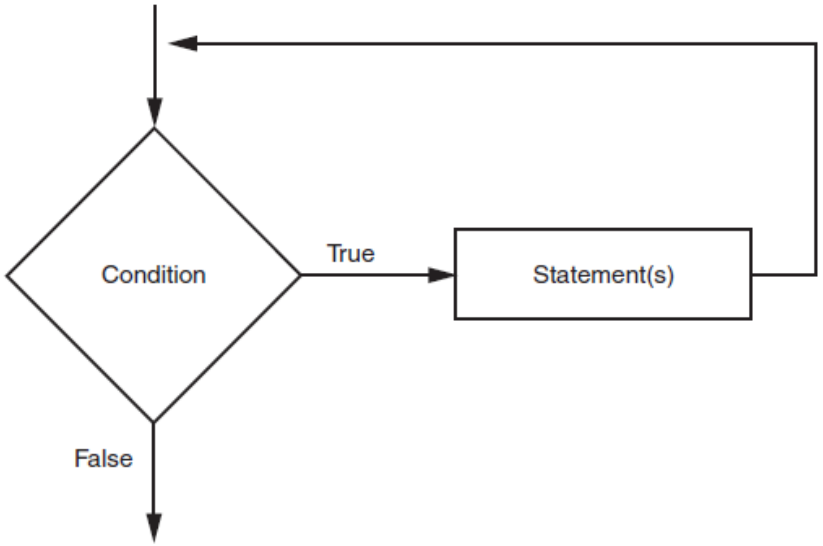
- Often want to repeat the same sequence of actions many times
- Could write a function and call it lots of times:

```
ComputeCardValue(card1)
ComputeCardValue(card2)
ComputeCardValue(card3)
```

- But what if there are many cards? Or a variable number of cards?

“While loops”

Figure 4-1 The logic of a `while` loop



The `while` Loop: a Condition-Controlled Loop

- `while` loop: while condition is true, do something
  - Condition tested for true or false value
  - Statements repeated as long as condition is true
  - General format:

```
while condition:
  statements
```

While loop example

```
while (x > 0):
    print(x)
    x = x - 1

print('Blastoff!')
```

While loop details

- The condition has the same format as the “if” statement (can use logical operators)
- The condition is tested at the beginning of each iteration of the loop
- So if the condition is false when we reach the while loop, the loop will never be run at all
- Something must happen within the loop that will cause the condition to change

Quiz Q1

- What will this code print out?

```
x = 10
while (x < 5):
    print(x)
    x = x + 1
```

Quiz Q2

- What will this code print out?

```
x = 3
y = 0
while (x < 5):
    print(x)
    y = y + 1
```

Quiz Q3

- What will this code print out?

```
x = 2
while (x < 5):
    print(x)
    x = 2*x
```

Reading repeated inputs

- Can use while loops to read multiple inputs

```
sum = 0
current_input = input('Number: ')
while current_input != 'done':
    sum = sum + int(current_input)
    current_input = input('Number: ')
print('Sum = ', sum)
```

Combining with if statements

```
x = -3
while x < 5:
    if (x < 0):
        print('Negative')
    elif (x == 0):
        print('Zero')
    else:
        print('Positive')
    x = x + 1
```

IDLE practice

- Compute n factorial (n!)
- Draw turtle rings
- Draw turtle spiral
- Print factors of number
- Compute blackjack for any number of aces

Common loop pattern

- Very often we want to loop through a range of numbers:

```
x = 1
while (x < 10):
    print(x)
    x = x + 1
```

The for loop

```
for x in range(1, 11):
    print(x)
```

- for [variable] in range([start], [stop]):
- Last number in loop is ONE LESS than stop
  - This will be useful for lists next week
- How to use this to compute factorial?

Range step argument

- Can also give a third “step” argument to range

```
for x in range(1,11,3):
    print(x)
```

- Negative step can count backwards

```
for x in range(10,0,-3):
    print(x)
```

Augmented Assignment

- We keep writing things like x = x + 2
- Can use “augmented assignment” as an abbreviation: x += 2

Table 4-2 Augmented assignment operators

Operator	Example Usage	Equivalent To
+=	x += 5	x = x + 5
-=	y -= 2	y = y - 2
*=	z *= 10	z = z * 10
/=	a /= b	a = a / b
%=	c %= 3	c = c % 3

Nested loops

- Can put loops inside loops

```
for n in range(1,10):
    factorial = 1
    for x in range(1,n+1):
        factorial *= x
    print(n,'factorial=',factorial)
```

IDLE practice

- Turtle grid of circles
- Prime factorization

Assignment: Collatz Conjecture

- Write a function that takes an integer as an argument
- Repeat:
  - If the number is even, divide by 2
  - If the number is odd, multiply by 3 and add 1
  - If you reach 1 then stop
- Print out the sequence of numbers
- Does it always stop? This is an unsolved problem in mathematics called the Collatz Conjecture