CSC 280 Introduction to Computer Science: Programming with Python

Lecture 7 : Conditionals, Nested Conditionals, For Loop
September 21 , 2015

Prof. Bei Xiao

Fall, 2015

American University

Outline

- Review of if elif else
- Nested Conditionals
- "While" loop
- Iterations

Conditionals: If-Elif-Else-Statement

Format

if <*boolean-expression*>:
    <*statement*>
    ...
elif <*boolean-expression*>:
    <*statement*>
    ...
...
else:
    <*statement*>
    ...

Example

- No limit on number of elif
  - Can have as many as want
  - Must be between if, else
- The else is always optional
  - if-elif by itself is fine
- Booleans checked in order
  - Once it finds a true one, it skips over all the others
  - else means **all** are false

Conditionals: Nested If

Format

```
If  <boolean-expression>:
        <statement>
        if <boolean-expression>:
                <statement>
        elif <boolean-expression>:
                <statement>
elif <boolean-expression>:
        <statement>
elif:
        <statement>
```

Example

```
If age>= 18:
        if foo> 21:
                print “you can drink”
        else:
                print “no drink, can vote”
else:
        print “needs parental control”
```

Demo: find if a number is odd or even

```
def oddnumber(x):
    if (x%2) == 0:
        print "Even"
    else:
        print  "Odd"
        if x%3 != 0:
            print 'and not divisable by 3'

oddnumber(15)
```

min(x,y,z)

```
def min(x,y,z)

   if x<y:
        if x<z:
            print "x is the least"
        else:
            print "z is the least"
   else:
        if y<z:
            print "y is the least"
        else:
            print "z is the least”

min(34,2,34)
```

Exercise: nested if else

Exercise 3 Write a program to print whether a number can be divided by 2 and 3 or only by 2 or only by 3.

For example:

x = 4

Divisible by 2 but not by 3

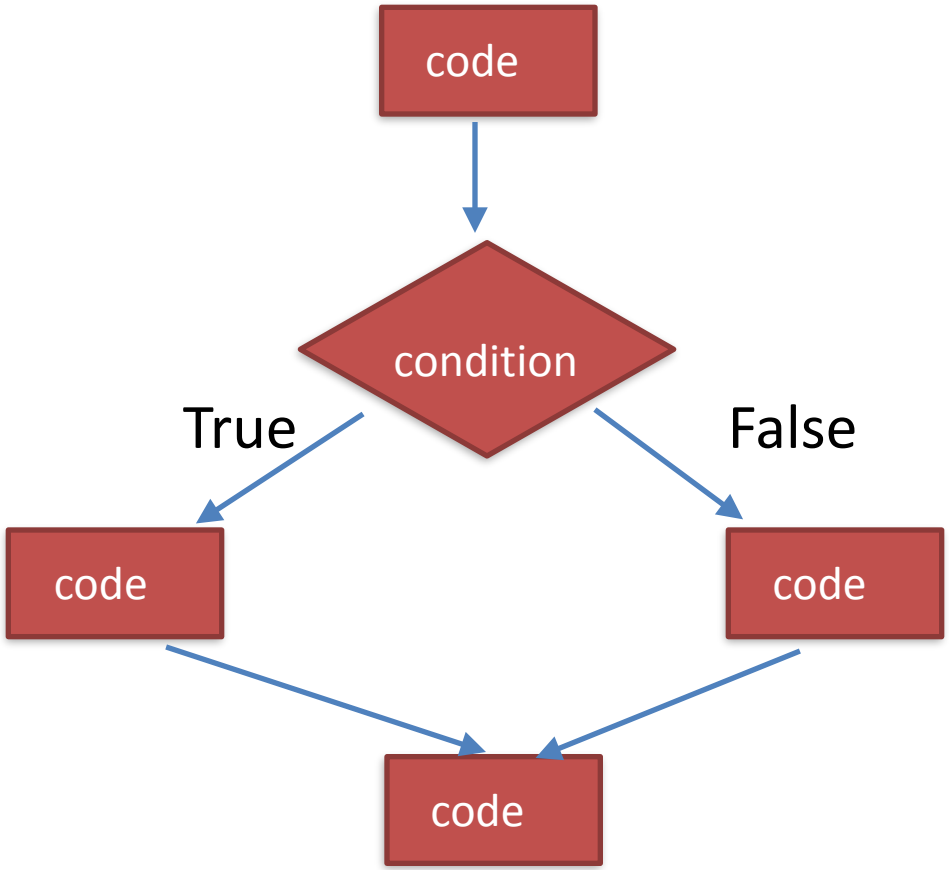
x = 6

Divisible by 2 and 3

x = 9

Divisible by 3 but not 2

Recall conditionals



not firm?

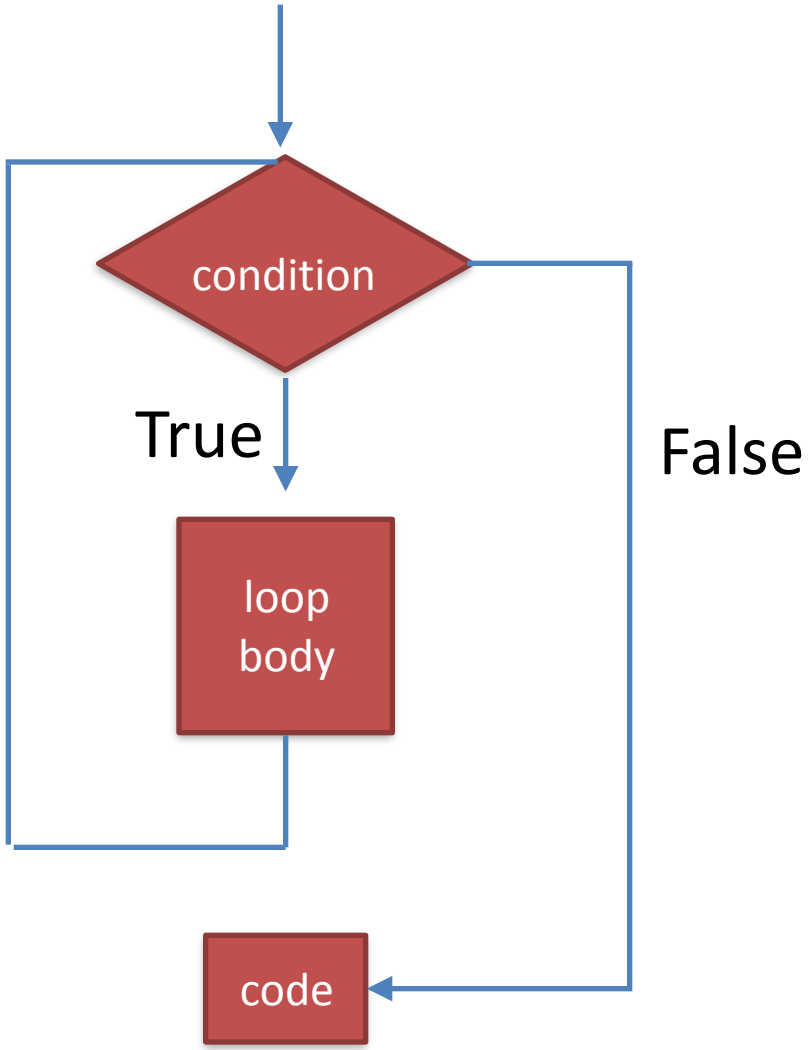
Yes

No

keep Whip

Serve

Iteration



Example: are you bored?

- Ask the user to say whether they are bored yet. The can answer “yes” or “no”.
- Repeat the question until the user typed “yes”.

While Loop

```
while <Boolean expression>:
    stmt1
    stmt2
stmt3
```

Demo: compute square, the hard way

```
x = 4
ans = 0
itersLeft = x
while (itersLeft !=0):
    ans = ans+x
    itersLeft = itersLeft-1

print str(x) + '*'+ str(x) +'='+ str(ans)
```

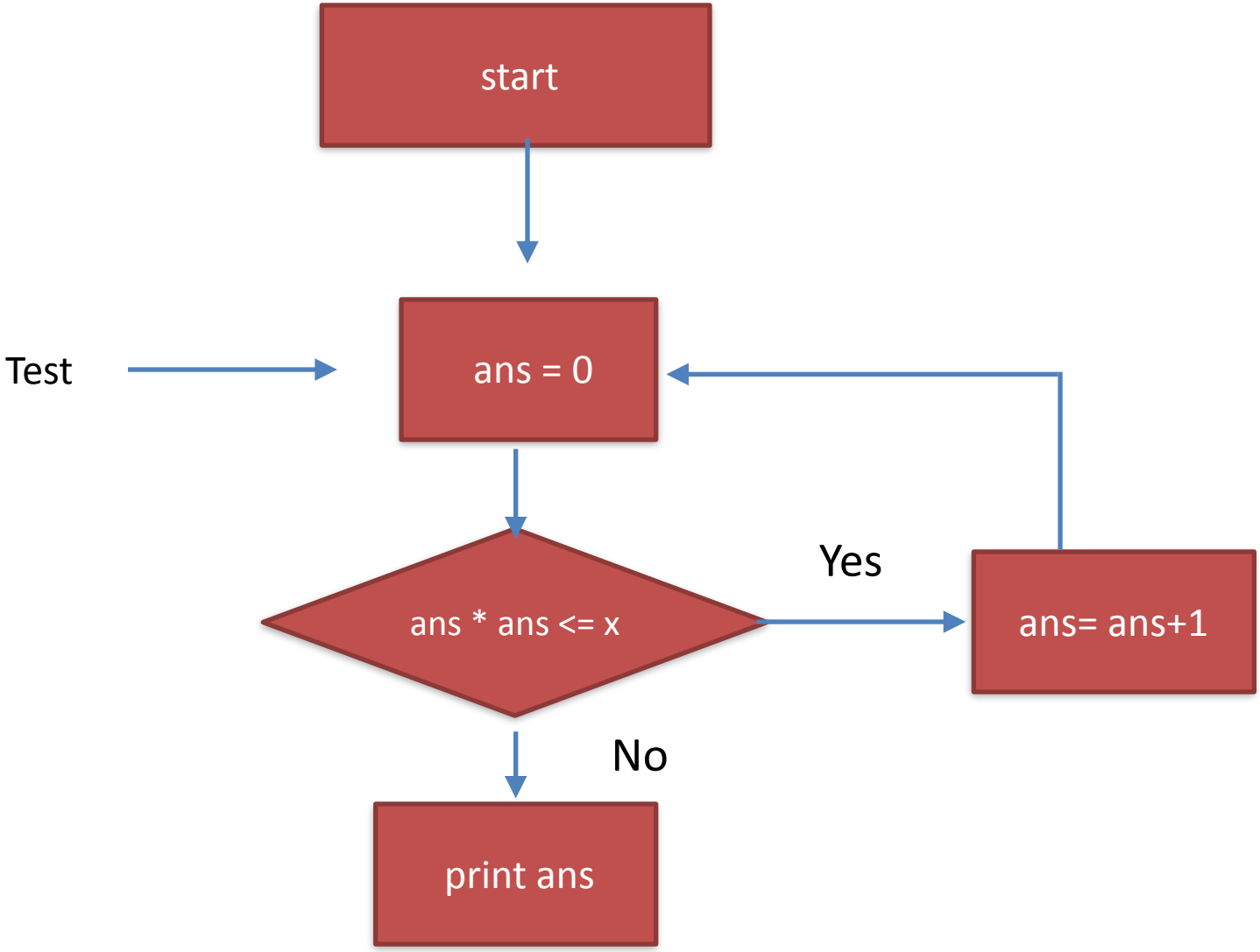
- Hand-simulating: What happened at each iteration?
- Write it down
- What happens if I remove the test after "while"?
- How about if I make "x = 3.5", what will happen to the code?
- **How about x = -1? Will the loop run?**

Demo: compute square root

1. x = 25, compute the square root
2. start with y = 1, is and y* y > x?, no
3. y = y+1, is y*y <=x, no,
4. .....

6. until y*y = 25, end, print y

Flowchart



Questions to ask about while loop:

- To what value x does the code terminate?
- To what value of x does the code give me the right answer?

Exercise: cube root of a perfect cube

Use Exhaustive Enumeration (while loop)

to compute the **cube root** of a perfect cube (e.g. 3 is cube root of 27). If the number is not a perfect cube, tell the user that it is not the perfect cube.

Hint: cube of a number can be negative. using abs()

```
while ans**3 <abs(x):
        ans = ans +1
……
```

For loop

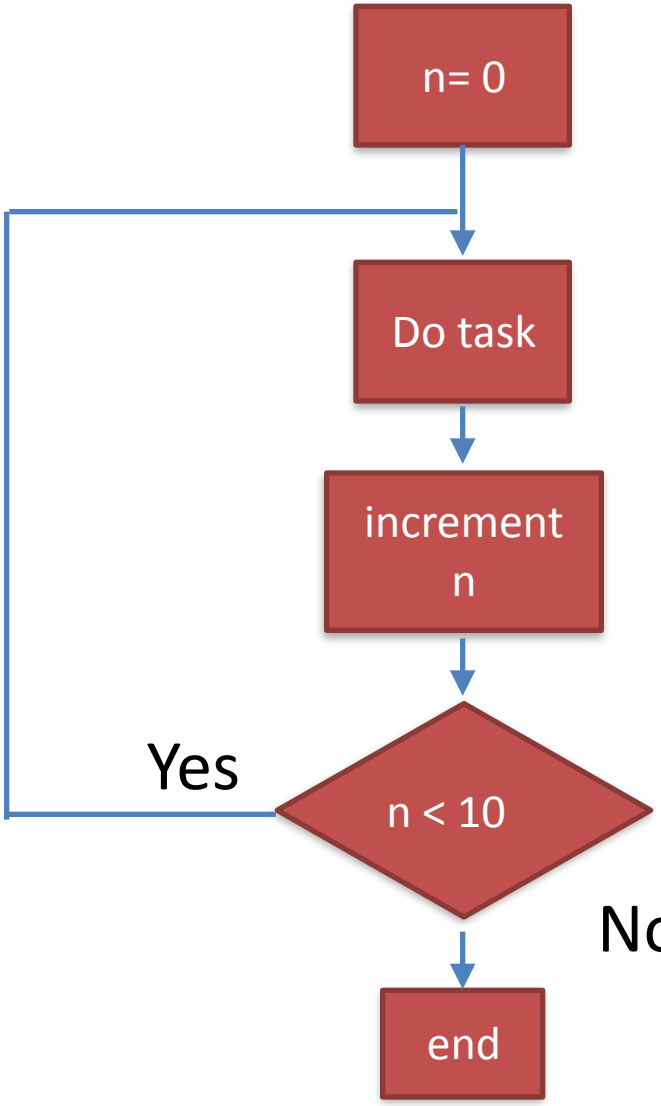
for <var> in <some collection> :

    code block

Going through a predefined sequences automatically without worrying bout explicit updates.

As long as collection is finite, the loop will end.

For Loop



e.g. print “hello world!”

Loop: range() function

range(start, stop[, step])

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(1, 11)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

All parameters can be positive or negative

range() is 0 index based, meaning list indexes start at 0, not 1

https://docs.python.org/2/library/functions.html#range

Loop: in

in means “is a member”

```
>>> for i in range(5):
...     print(i)
...
0
1
2
3
4
```

Quiz

Write down the output of the following commands:

1. range(5,10)
2. range(0, 10, 3)
3. range (-10,-100, -30)
4. range(1,7)
5. range(5, -1, -1)
6. for i in range(3,6):

   print i

range(start, stop[, step])

```
>>> range(10)
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>> range(1, 11)
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
```

Take-home

- Read Chapter 5
- http://www.greenteapress.com/thinkpython/html/thinkpython006.html#toc58
- Practice with conditional flows:
- http://learnpythonthehardway.org/book/ex30.html
- Must Drill Hard on While and For Loop!