Programming Foundations in Python
Adapted From: CMSC 201 Computer Science I for Majors

Lecture 06 – While Loops

Last Class We Covered

- Algorithms
- Program design
  - Input, process, output
  - Flowcharts and pseudocode
- Syntax and Logic Errors

Any Questions from Last Time?

Today's Objectives

- To learn about and use a **`while`** loop
  - To understand the syntax of a **`while`** loop
  - To use a **`while`** loop for interactive loops
  - To learn about infinite loops
    - (And how to avoid them)

- To practice conditionals

Pop Quiz: `if`, `elif`, and `else`

```
if
else
elif
```

```
if
elif
elif
else
else
```

```
if
elif
if
elif
```

```
if
if
elif
else
```

```
if
    if
    if
else
    if
```

```
if
else
    if
    elif
    elif
else
```

Answers: `if`, `elif`, and `else`

```
if
else
elif
```

```
if
elif
elif
else
else
```

```
if
elif
if
elif
```

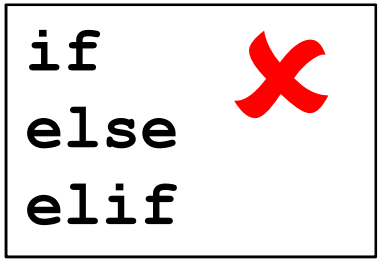
```
if
if
elif
else
```

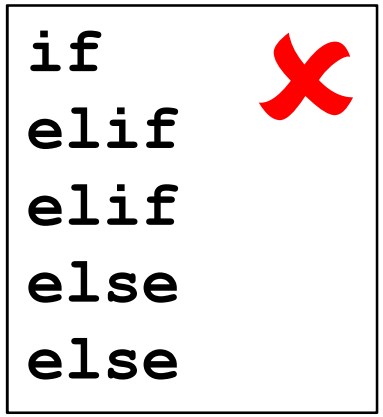
```
if
    if
    if
else
    if
```

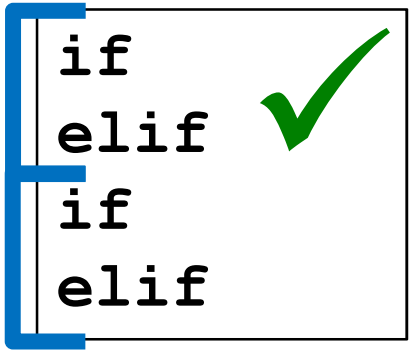
```
if
else
    if
    elif
    elif
else
```

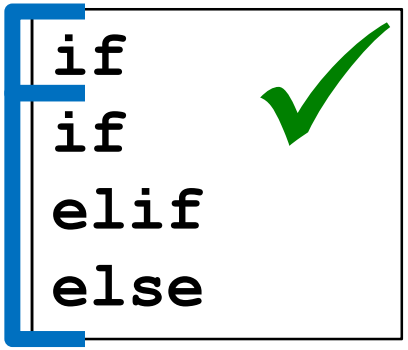
```
if
elif
```

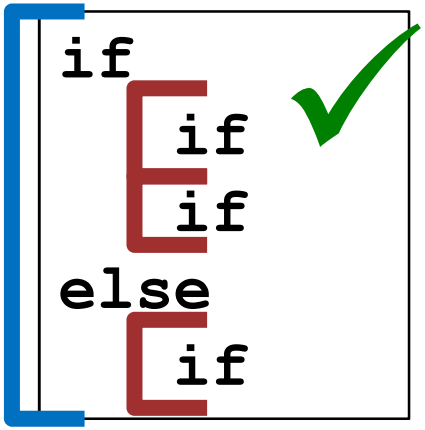
Answers: `if`, `elif`, and `else`

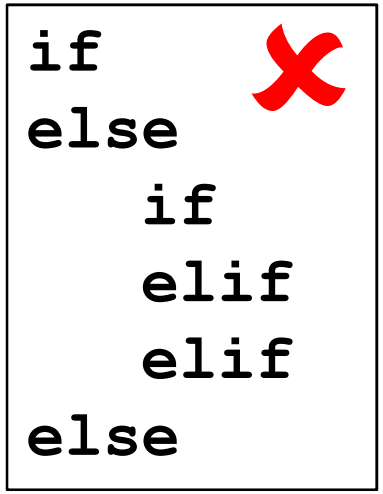


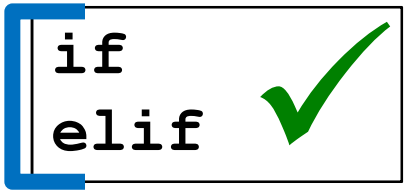












Looping

Control Structures

- Structures that control how the program "flows" or operates, and in what order

- Sequence ✓

we've already seen these

- Decision Making ✓

- Looping

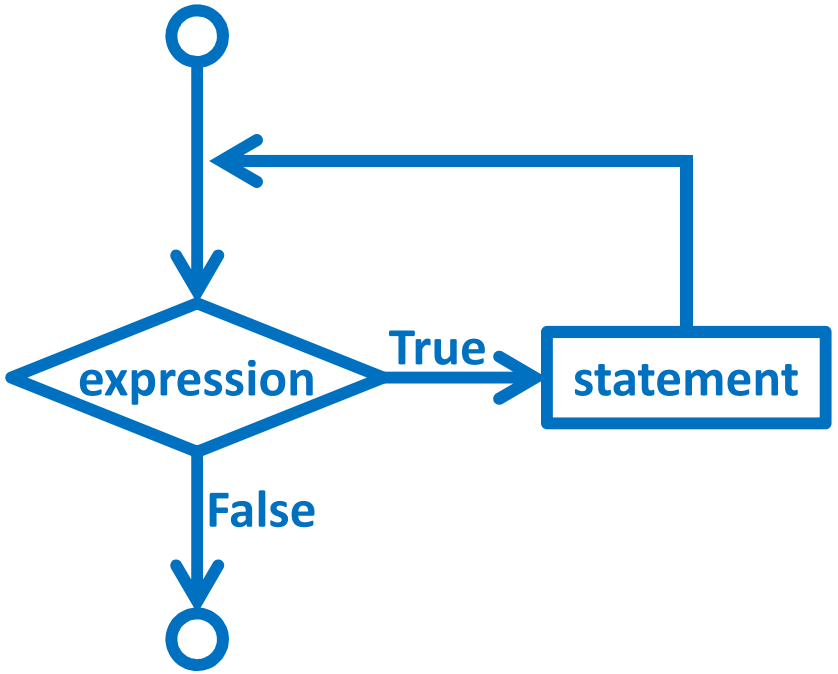
what we're covering today

Looping

- Doing something over and over (and over) again
- Used in combination with decision making
  - If not decision is being made, we just loop forever
    - This is called an "infinite loop"
- What are some real life examples?
  - Jumping rope
  - Walking up steps

Looping

- We'll cover this in detail today
- It looks something like this…



Looping

- Python has two kinds of loops, and they are used for two different purposes

what we're covering today

- The **`while`** loop
  - Works for basically everything
- The **`for`** loop:
  - Best at ***iterating*** over something
  - Best at counted iterations

The `while` Loop

"`while`" Loops

- **`while <condition>:`**
  `<body>`

- The **body** is a sequence of one or more statements indented under the heading
  - As long as the **condition** is **`True`**, the **body** will run (repeatedly if needed)

How a `while` Loop Works

- The **`while`** loop uses a Boolean condition
  - That evaluates to either **`True`** or **`False`**
- If the condition is **`True`**:
  - Body of **`while`** loop is executed
    - Once that's over, condition is checked again
- If the condition is **`False`**:
  - Body of **`while`** loop is skipped

Parts of a `while` Loop

- Here's some example code… let's break it down

```
date = 0

while date < 1 or date > 30:
    date = int(input("Enter the day: "))

print("Today is September", date)
```

Parts of a `while` Loop

- Here's some example code… let's break it down

initialize the variable the `while` loop will use for its decision

```
date = 0

while date < 1 or date > 30:
    date = int(input("Enter the day: "))

print("Today is September",
```

the loop's Boolean condition (loop runs until this is `False`)

the body of the loop (must change the value of the loop variable)

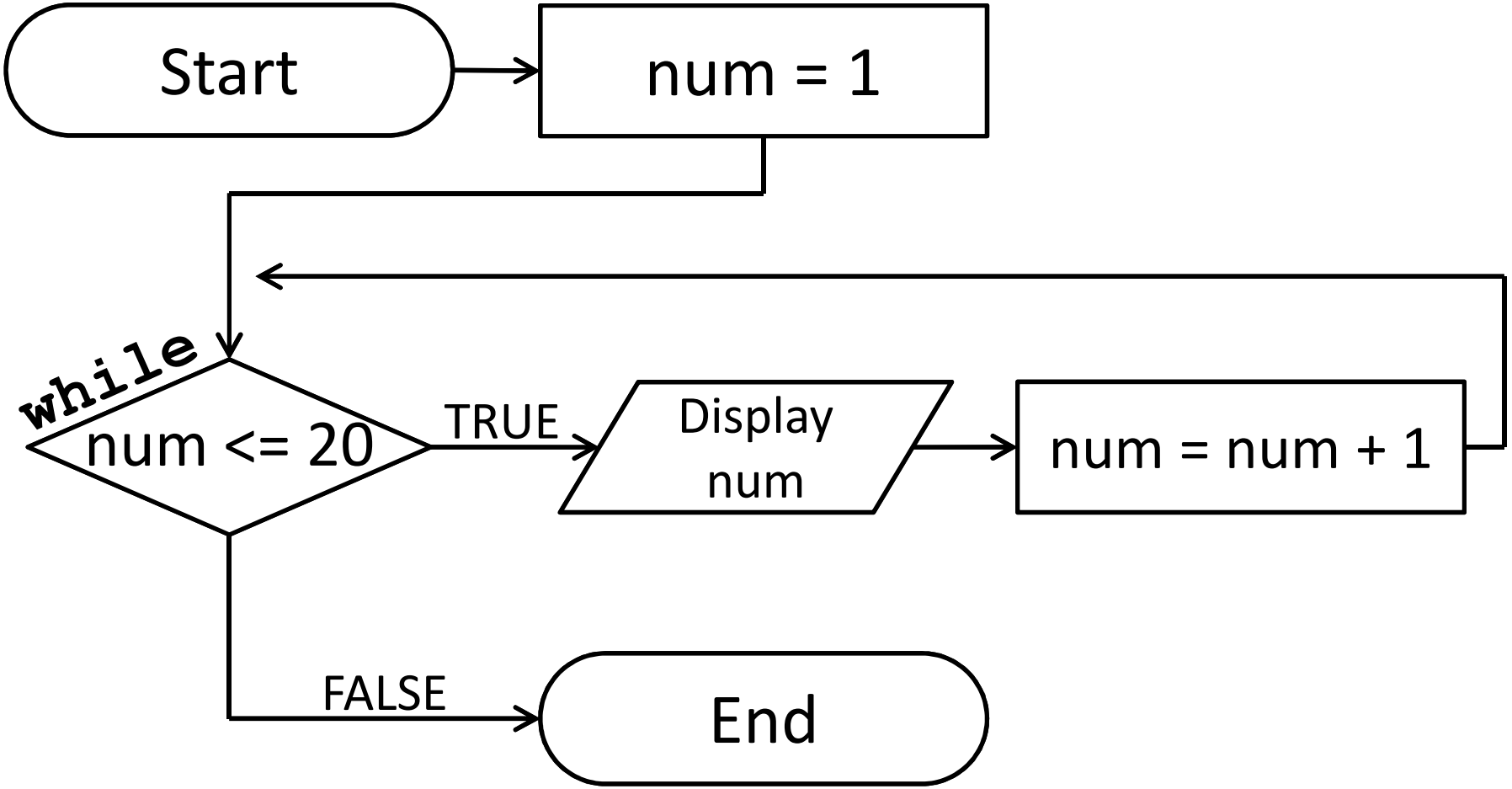
Example `while` Loop

- We can use a **`while`** loop to count
  - Count from 1 up to and including 20

```
num = 1                     # we have to initialize num

while num <= 20:            # so that we can use it here
    print(num)
    num = num + 1           # don't forget to update
                            # the loop variable
```

Example `while` Loop



Example Counting `while` Loops

- By changing a number of factors, you can change how a counting loop behaves

```
num = 1                    # this controls the start
while num <= 20:           # this controls the stop
    print(num)
    num = num + 1          # this controls the update
```

- How would you count 2, 4, 6, … 96, 98, 100?
- What about from 10 down to 0?

Calculating `while` Loops

- For a `while` loop that needs to calculate something, you must initialize the relevant variable outside of the loop, before it starts

- For example, if calculating the total of 10 user-provided numbers, initialize a "total" variable before the while loop

Totaling `while` Loop

- Here is a completed example:

```
count = 0
total = 0

while count < 10:
    num = int(input("Please enter a number: "))
    total += num
    count += 1
print("The total is", total)
```

Infinite Loops and Other Problems

Loop Body Not Being Reached

- A **`while`** loop's body may be skipped over entirely
  - If the Boolean condition is initially **`False`**

```
militaryTime = 1300

while (militaryTime < 1200):
    print("Good morning!")
    militaryTime = militaryTime + 100
```

Infinite Loops

- An ***infinite loop*** is a loop that will run forever
  - The conditional the loop is based on always evaluates to **`True`**, and never to **`False`**

- Why might this happen?
  - The loop variable is not updated
  - The loop variable is updated wrong
  - The loop conditional uses the wrong variable
  - The loop conditional checks the wrong thing

Infinite Loop Example #1

- Why doesn't this loop end? What will fix it?

```
age = int(input("How old are you? "))
while age < 18:   # can't vote until 18
    print("You can't vote at age", age)

print("Now you can vote! Yay!")
```

Infinite Loop Example #1

- Why doesn't this loop end? What will fix it?

the loop variable (**age**) never changes, so the condition will never evaluate to **False**

```
age = int(input("How
while age < 18:    # can't vote until 18
    print("You can't vote at age", age)

print("Now you can vote! Yay!")
```

Infinite Loop Example #2

- Why doesn't this loop end? What will fix it?

```
while True:
    # ask user for name
    name = input("What is your name? ")

print("Hello", name + "!")
```

Infinite Loop Example #2

- Why doesn't this loop end? What will fix it?

`True` will never evaluate to `False`, so the loop will never exit

```
while True:
    # ask user for name
    name = ir                    ame? ")

print("Hello", name + "!")
```

Don't ever do this! It's sloppy programming!.

Infinite Loop Example #3

- Why doesn't this loop end? What will fix it?

```
cookiesLeft = 50

while cookiesLeft > 0:
    # eat a cookie
    cookiesLeft = cookiesLeft + 1

print("No more cookies!")
```

Infinite Loop Example #3

- Why doesn't this loop end? What will fix it?

```
cookiesLeft = 50

while cookiesLeft > 0:
    # eat a cookie
    cookiesLeft = cookiesLeft + 1

print("No more cookies!")
```

the loop body is INCREASING the number of cookies, so we'll never reach zero!

Infinite Loop Example #4

- Why doesn't this loop end? What will fix it?

```
countdown = 10

print("Countdown begin...")
while countdown > 0:
    print(countdown, "...")

print("Blastoff!")
```

Infinite Loop Example #4

- Why doesn't this loop end? What will fix it?

```
countdown = 10

print("Countdown begin
while countdown > 0:
    print(countdown, "...")

print("Blastoff!")
```

the countdown variable is not being decremented, so it will never go below zero

Infinite Loop Example #5

- Why doesn't this loop end? What will fix it?

```
grade = ""
name  = ""
while name != "Hrabowski":
    # get the user's grade
    grade = input("What is your grade? ")

print("You passed!")
```

Infinite Loop Example #5

- Why doesn't this loop end? What will fix it?

the loop conditional is checking the wrong variable! we also never change the name, so this will never end

```
grade = ""
name  = ""
while name != "Hrabowski":
    # get the user's grade
    grade = input("What is your grade? ")

print("You passed!")
```

Ending an Infinite Loop

- If you run a Python program that contains an infinite loop, it may seem like you've lost control of the terminal!

- To regain control, simply type **`CTRL+C`** to interrupt the infinite loop
  - **`KeyboardInterrupt`** will be displayed, and you'll regain control

When to Use `while` Loops

- **`while`** loops are very helpful when you want to get input from the user that meets certain specific conditions
  - Positive number
  - A non-empty string
  - A number within a certain range

Example `while` Loop

- We can use a **`while`** loop to get correct input from the user by re-prompting them

```
num  = 0

# so that we can use it here
while num <= 0:
    num = int(input("Enter a positive number: "))

# while loop exits because num is positive
print("Thank you.  The number you chose is:", num)
```

Project 2: Guessing Game "updated"

- Update your "guessing game" to keep re-prompting the user until they guess the right number
- At each step, tell them whether their guess was high or low
- Exit the loop when the user guesses correctly.
- Send to my email address, the URL of your code implementation, and preferably a text file containing your code as well as an attachment.
- If you're running out of time, send me the URL first, then send attachment later.
- The subject of your email to me is as follows: Upward_Bound_Project_2
- Deadline is : Monday, July 9th at 11:59 pm.
- Ideally, should be done by Sunday night, but it is extended to Monday incase you run into any programming issues you can't resolve on your own, we can talk about it on Monday in class.
- Make sure to follow the above stated instructions, failure to do so will result in penalties on your overall score.

Image Sources

- Infinity symbol:
  - https://commons.wikimedia.org/wiki/File:Flat_UI_-_infinity.png