CMSC201 Computer Science I for Majors

Lecture 07 – While Loops

All materials copyright UMBC unless otherwise noted

Last Class We Covered

- Decision Structures
  - Multi-way (using **`if-elif-else`** statements)
- How strings are represented
- How to use strings:
  - Indexing
  - Slicing
  - Concatenate and Repetition

Any Questions from Last Time?

UMBC
AN HONORS UNIVERSITY IN MARYLAND

Slicing Practice (Review)

0	1	2	3	4	5	6	7	8
T	r	u	e		G	r	i	t
-9	-8	-7	-6	-5	-4	-3	-2	-1

```
>>> grit[3:2]
''
>>> grit[4:-4]
' '
>>> grit[-8:-4]
'rue '
>>> grit[-4:]
'Grit'
```

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 apply our knowledge to create nested loops
- To practice conditionals

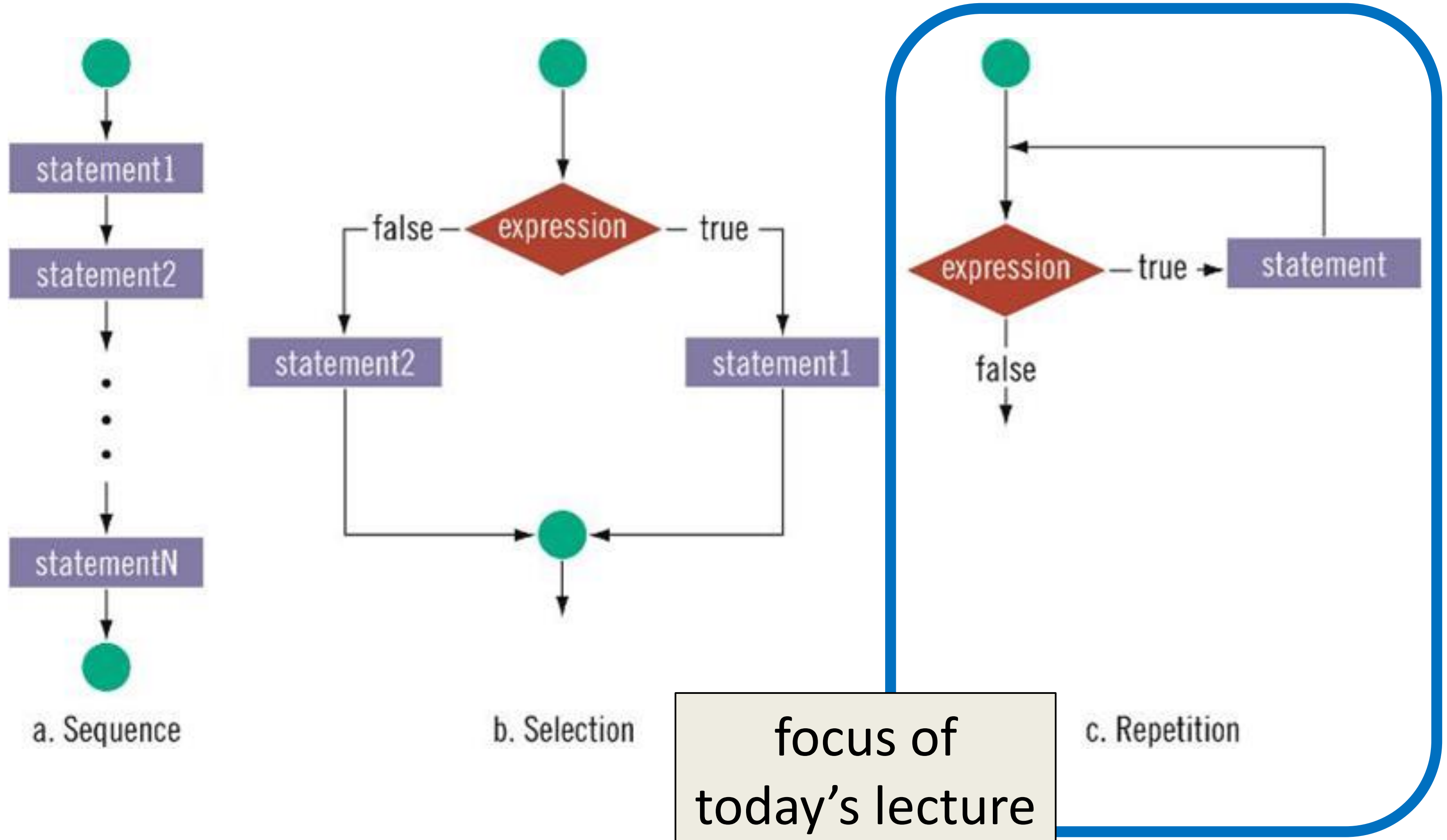
Looping

Control Structures (Review)

- A program can proceed:
  - In sequence
  - Selectively (branching): make a choice
  - Repetitively (iteratively): looping
  - By calling a function

focus of today's lecture

Control Structures: Flowcharts



focus of today's lecture

Looping

- Python has two kinds of loops, and they are used for two different purposes
- The `while` loop
  - Works for basically everything
- The `for` loop:
  - Best at ***iterating*** over a list
  - Best at counted iterations

what we're covering today

The **while** Loop

The **`while`** Loop

- The **`while`** loop is best used when we're not
  - Iterating over a list
  - Doing a "counted" loop

- Works the way its name implies:

While a conditional evaluates to True:

Do a thing (repeatedly, if necessary)

"**`while`**" Loops

- The Python **`while`** loop is used to control the flow of the program

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

- The **body** is a sequence of one or more statements <u>indented</u> under the heading
  - As long as the **condition** is **`True`**, the **body** will run

Parts of a `while` Loop

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

```
date = 0

while date < 1 or date > 31:
    date = int(input("Enter the day: "))

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

UMBC

AN HONORS UNIVERSITY IN MARYLAND

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

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

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

```
date = 0

while date < 1 or date > 31:
    date = int(input("Enter the day: "))

print("Today is September",
```

How a **`while`** Loop Works

- The **`while`** loop requires a Boolean condition
  - That evaluates to either **`True`** or **`False`**
- If the condition is **`True`**:
  - Body of **`while`** loop is executed
- If the condition is **`False`**:
  - Body of **`while`** loop is skipped

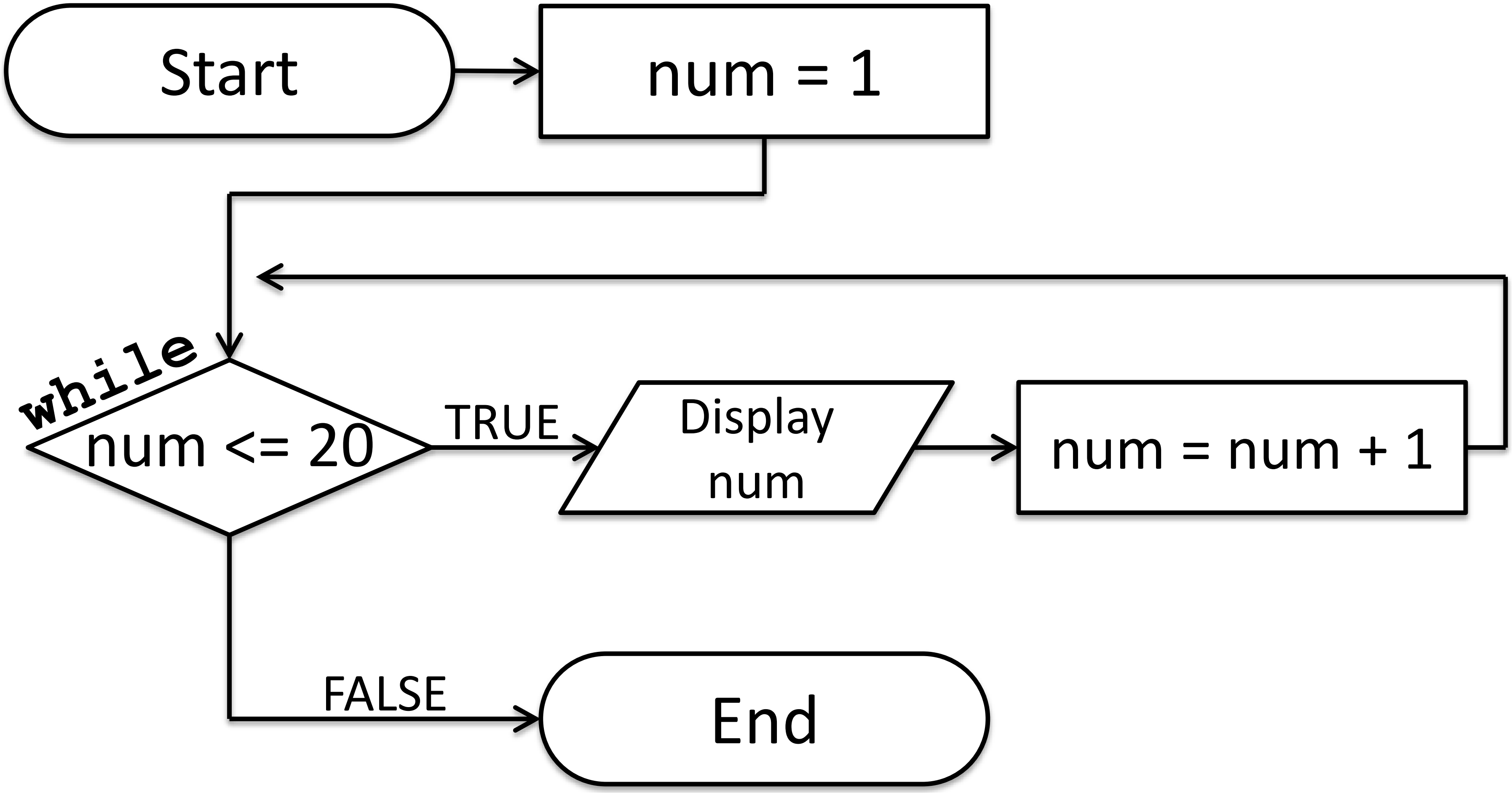
Example `while` Loop

- We can use a **`while`** loop to do a “counting” loop, just like we did earlier
  - 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



Infinite Loops and Other Problems

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 = 0
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 = 0
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 = input("What is your name? ")

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

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?

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

print("You passed!")
```

Infinite Loop Example #4

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

the loop conditional is checking the wrong thing! 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 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`**

Loop Body Isn't Being Run

- 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
```

Practice with Decisions

Loop Example #4 – Fixed

- Let’s update this to ask for the user’s grade
  – An “A” or a “B” means that they passed

```
grade = ""
while    ...what goes here?
    # get the user's grade
    grade = input("What is your grade? ")

print("You passed!")
```

Loop Example #4 – Truth Table

- Let’s evaluate this expression

```
grade != "A" or grade != "B"
```

`grade`	`grade != "A"`	`grade != "B"`	`or`
`"A"`			
`"B"`			
`"C"`			

Loop Example #4 – Truth Table

- Let’s evaluate this expression

```
grade != "A" or grade != "B"
```

`grade`	`grade != "A"`	`grade != "B"`	`or`
`"A"`	`False`	`True`	`True`
`"B"`	`True`	`False`	`True`
`"C"`	`True`	`True`	`True`

- This does not give us the answer we want
  - This just loops forever and ever (infinitely)

Loop Example #4 – Truth Table

- Let's try it with an **`and`** instead of an **`or`**

```
grade != "A" and grade != "B"
```

`grade`	`grade != "A"`	`grade != "B"`	`and`
`"A"`			
`"B"`			
`"C"`			

Loop Example #4 – Truth Table

- Let’s try it with an **`and`** instead of an **`or`**

  `grade != "A" and grade != "B"`

`grade`	`grade != "A"`	`grade != "B"`	`and`
`"A"`	**`False`**	**`True`**	**`False`**
`"B"`	**`True`**	**`False`**	**`False`**
`"C"`	**`True`**	**`True`**	**`True`**

- Now our program will behave how we want
  - You will sometimes have to stop and make a table!

Loop Example #4 – Fixed

- Let’s update this to ask for the user’s grade
  - An “A” or a “B” means that they passed

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

print("You passed!")
```

Interactive **`while`** Loops

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
  - Want to keep getting input until some "end"
    - User inputs a value that means they're finished
    - Reached the end of some input (a file, etc.)

Example **`while`** Loop

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

```
num  = 0                 # we have to initialize num

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

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

Nested Loops

Nesting

- You have already used nested statements
  - In HW 3, you used nested **`if/elif/else`** statements to help you guess a dog breed
- We can also nest loops!
  - First loop is called the ***outer loop***
  - Second loop is called the ***inner loop***

Nested Loop Example

- What does this code do?

```
course = 201
while course < 203:
    grade = input("What is your grade in", course, "? ")

    while grade != "A" and grade != "B":
        print("That is not a passing grade for", course)
        grade = input("New grade in", course, "? ")

    course = course + 1
```

Nested Loop Example

- What does this code do?

initializes **course**

continues until **course** is 203

will keep running while **grade** is unacceptable

```
course = 201
while course < 203:
    grade = input("What is your gra

    while grade != "A" and grade != "B":
        print("That is not a passing grade for", course)
        grade = input("New grade in", course, "? ")

    course = course + 1
```

updates **course** for the outer **while** loop

Time for…

LIVECODING!!!

Livecoding: Password Guessing

- Write a program that allows the user to try guessing a password. It should allow them to guess the password up to three times.

- You will need to use:
  - At least one while loop
  - String comparison
  - Conditionals
  - Decision Structures

Announcements

- Homework 3 is out
  - Due by Wednesday (September 28th) at 8:59:59 PM
- Homeworks are on Blackboard
  - Homework 1 grades will be released soon
- Pre Labs are available on the course website