CONCEPTS OF PROGRAMMING

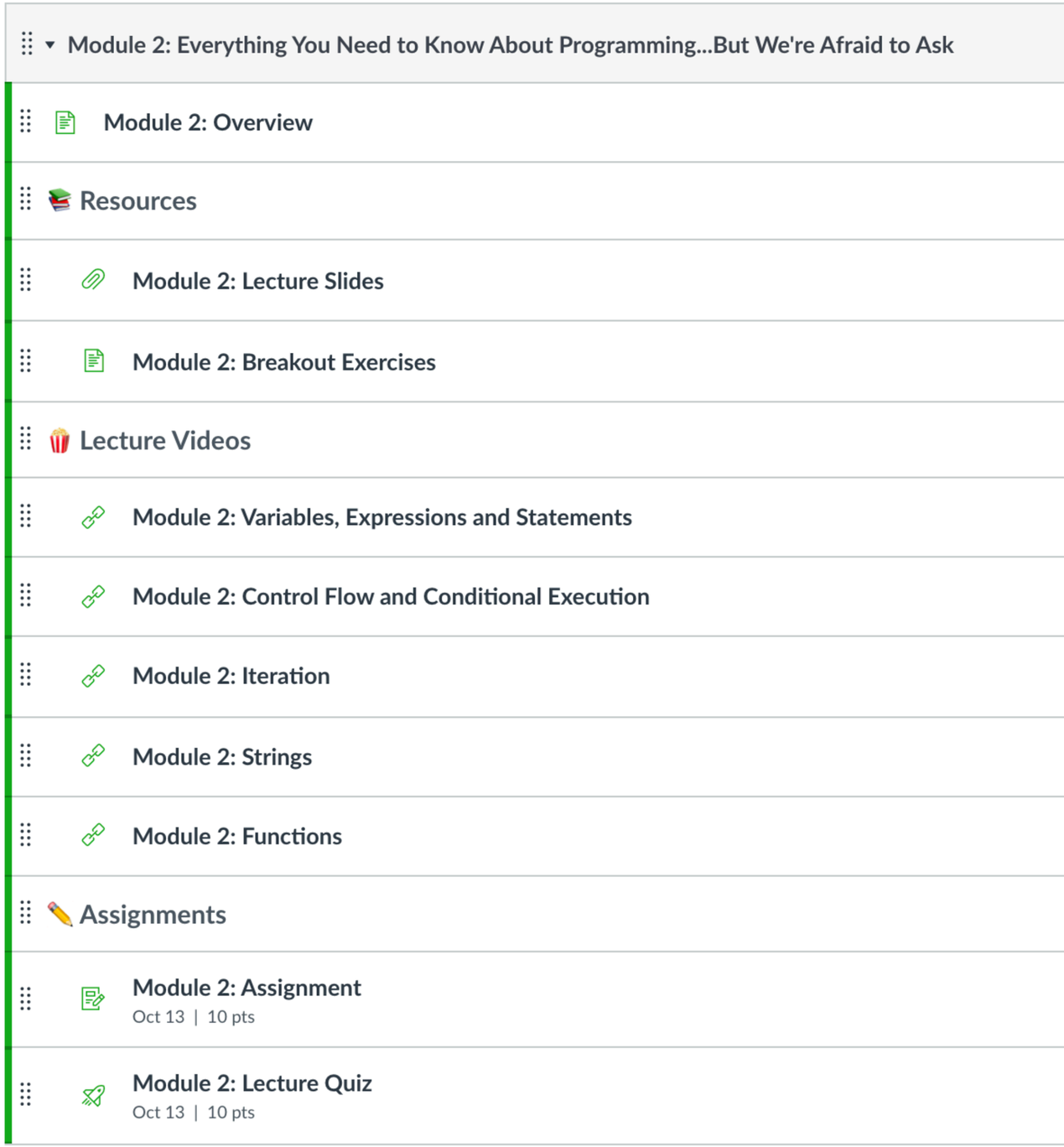
©TTAA.BBINKOWSKI,2019

MODULE 2
MPCS 50101

THE UNIVERSITY OF CHICAGO

OUTLINE

- Class News
- Review
  - Lecture Quiz
  - Unix Challenge
- Module Preview
- Assignment Preview



CLASS NEWS

© T.A. BINKOWSKI, 2020

MODULE 1
MPCS 50101

THE UNIVERSITY OF CHICAGO

CLASS NEWS

- Getting development environment setup is a big first step
- Timeline for grading and resubmission

```
while True:
    string_input = raw_input('>  ')
    guess = int(string_input)

    if guess == 4:
        continue
    if guess == 1:
        break

print 'You guessed it!'

```

```
505 % python guess.py
I am thinking of a number between 1 and 20.
>  3
>  2
>  1
You guessed it!
tabinkowski@Ts-MacBook-Pro ~/Google Drive/g-Teaching/uchicago.codes/uchicago.codes-courses/mpcs50101/mpcs50101-2018-winter/sessions/session2
```

COMPUTING NEWS

- Python Module of the Week

Python 3 Module of the Week

PyMOTW-3 is a series of articles written by Doug Hellmann to demonstrate how to use the modules of the Python 3 standard library. It is based on the original PyMOTW series, which covered Python 2.7. See About Python Module of the Week for details including the version of Python and tools used.

- Text
  - string — Text Constants and Templates
  - textwrap — Formatting Text Paragraphs
  - re — Regular Expressions
  - difflib — Compare Sequences
- Data Structures
  - enum – Enumeration Type
  - collections — Container Data Types
  - array — Sequence of Fixed-type Data
  - heapq – Heap Sort Algorithm
  - bisect — Maintain Lists in Sorted Order
  - queue — Thread-Safe FIFO Implementation
  - struct — Binary Data Structures
  - weakref — Impermanent References to Objects
  - copy — Duplicate Objects
  - pprint — Pretty-Print Data Structures
- Algorithms
  - functools — Tools for Manipulating Functions
  - itertools — Iterator Functions
  - operator — Functional Interface to Built-in Operators
  - contextlib — Context Manager Utilities
- Dates and Times
  - time — Clock Time
  - datetime — Date and Time Value Manipulation
  - calendar — Work with Dates
- Mathematics
  - decimal — Fixed and Floating Point Math
  - fractions — Rational Numbers
  - random — Pseudorandom Number Generators

COMPUTING NEWS

- Good resource for the most commonly used modules
  - This one useful for the homework

random — Pseudorandom Number Generators

Purpose: Implements several types of pseudorandom number generators.

The `random` module provides a fast pseudorandom number generator based on the *Mersenne Twister* algorithm. Originally developed to produce inputs for Monte Carlo simulations, Mersenne Twister generates numbers with nearly uniform distribution and a large period, making it suited for a wide range of applications.

Generating Random Numbers

The `random()` function returns the next random floating point value from the generated sequence. All of the return values fall within the range `0 <= n < 1.0`.

```
# random_random.py

import random

for i in range(5):
    print('%04.3f' % random.random(), end=' ')
print()
```

Running the program repeatedly produces different sequences of numbers.

```
$ python3 random_random.py

0.859 0.297 0.554 0.985 0.452

$ python3 random_random.py

0.797 0.658 0.170 0.297 0.593
```

To generate numbers in a specific numerical range, use `uniform()` instead.

```
# random_uniform.py

import random

for i in range(5):
```

COMPUTING NEWS

- 4th of July Observed

HTTPS://NEWS.YCOMBINATOR.COM/

REVIEW

© T.A. BINKOWSKI, 2020

MODULE 2
MPCS 50101

THE UNIVERSITY OF CHICAGO

Unix Challenge

UNIX CHALLENGE

- Unix challenge
  - Man tar

```
TAR(1)                    BSD General Commands Manual

NAME
     tar -- manipulate tape archives

SYNOPSIS
     tar [bundled-flags <args>] [<file> | <pattern> ...]
     tar {-c} [options] [files | directories]
     tar {-r | -u} -f archive-file [options] [files | directo
     tar {-t | -x} [options] [patterns]

DESCRIPTION
     tar creates and manipulates streaming archive files.  Th
     extract from tar, pax, cpio, zip, jar, ar, xar, rpm, 7-z
     cdrom images and can create tar, pax, cpio, ar, zip, 7-z
     chives.

     The first synopsis form shows a ``bundled'' option word.
     vided for compatibility with historical implementations.
     below for details.

     The other synopsis forms show the preferred usage.  The
     is a mode indicator from the following list:
     -c      Create a new archive containing the specified it
             form is --create.
     -r      Like -c, but new entries are appended to the arc
```

UNIX CHALLENGE

```
tar -cvf my_files.tar file1 file2
tar -cvf my_files.tar file* ex*.py

-c create
-v verbose (tell us what's going on)
-f archive file name
```

UNIX CHALLENGE

NOTE THE .TGZ

```
tar -cvzf my_files.tgz file1 file2
```

```
-c create
-v verbose (tell us what's going on)
-f archive file name
-z compress
```

APPLIES COMPRESSION ALGORITHM (OPTIONAL)

UNIX CHALLENGE

```
tar -xvf my_files.tar
tar -xvf my_files.tgz

-x extract
-v verbose (tell us what's going on)
-f archive file name
-z compress
```

Lecture Quiz Review

LECTURE QUIZ

Question 1 pts

Which of the following criteria that is used to evaluate "good" programming practices can be considered objective?

ect Answer
- Efficiency
- Readability
- Complexity
- Style

Question 1 1 pts

Select all of the following statements that are correct about Python:

ect Answer
- It is an interpreted language.

ect Answer
- It can be run in script mode.

ect Answer
- It can be run in interactive mode.
- It is a compiled language.

Question 1 pts

What is the latest release version of Python 3?

ect Answers
- 3.9.5
- 3.9

LECTURE QUIZ

Question 2 1 pts

Since the release of Python 3, Python 2 is obsolete.

- True

  Python 2 is still very widely used. When software depends on previous versions they are sometimes referred to as “legacy” and using them is “technical debt”. However, if something works companies are very hesitant to change it.

Correct Answer

- False

Question 1 pts

PEP 8, sometimes spelled PEP8 or PEP-8, is a document that provides guidelines and best practices on how to write Python code. It was written in 2001 by Guido van Rossum, Barry Warsaw, and Nick Coghlan. The primary focus of PEP 8 is to improve the readability and consistency of Python code.

What does PEP stand for?

Correct Answers

- Python Enhancement Proposal
- python enhancement proposal
- Python enhancement proposal

Question 1 pts

Low-level languages are machine-dependent.

Correct Answer

- True
- False

LECTURE QUIZ

Question 1 pts

High-level languages are typically easier to develop and debug.

Correct Answer

- (•) True
- ( ) False

Question 1 pts

Which of the following is not a types of programming error?

Correct Answer

- (•) Onboarding
- ( ) Syntax
- ( ) Runtime
- ( ) Semantic

Question 1 pts

What is the variable responsible for telling the shell where to look for programs?

Correct Answers

- $PATH

Wrong answer comments
Shell variables are case sensitive and prefixed with a "$".

LECTURE QUIZ

Question 1 pts

Which of the following is not a types of programming error?

Correct Answer

- Onboarding
- Syntax
- Runtime
- Semantic

Question 1 pts

What is the variable responsible for telling the shell where to look for programs?

Correct Answers

$PATH

Wrong answer comments
Shell variables are case sensitive and prefixed with a "$".

LECTURE QUIZ

Question 1 pts

Do the following print statement all print out the exact same string?

```
x = "world"

print("Hello %s" % x)
print("Hello {}".format(x))
print(f"Hello {x}")
```

Correct Answer

- True
- False

Question 1 pts

UNIX is an acronym.

- True

In the 1960s, Massachusetts Institute of Technology, AT&T Bell Labs, and General Electric developed an experimental operating system called Multics for the GE-645 mainframe.[2] Multics was highly innovative, but had many problems.. In the 1970s Brian Kernighan coined the project name Unics as a play on Multics, (Multiplexing Information and Computer Services). Unics could eventually support multiple simultaneous users, and it was renamed Unix.

Correct Answer

- False

Correct answer comments

In the 1960s, Massachusetts Institute of Technology, AT&T Bell Labs, and General Electric developed an experimental operating system called Multics for the GE-645 mainframe.[2] Multics was highly innovative, but had many problems.. In the 1970s Brian Kernighan coined the project name Unics as a play on Multics, (Multiplexing Information and Computer Services). Unics could eventually support multiple simultaneous users, and it was renamed Unix.

LECTURE QUIZ

Question 1 pts

True or False: An Archive file is a file that is composed of one or more files along with metadata. Archive files are used to collect multiple data files together into a single file for easier portability and storage, or simply to compress files to use less storage space.

Correct Answer

- (•) True
- ( ) False

Question 1 pts

What is the standard file extension for an archived `tar` file?

Correct Answers

- .tar
- tar

Question 0 pts

The movie Anaconda staring Jennifer Lopez and Ice Cube is underrated and deserves a Tomatometer score higher than 38% on Rotten Tomatoes.

Correct Answer

- [x] True
- [ ] False

BREAKOUT EXERCISES

© T.A. BINKOWSKI, 2020

MODULE 2
MPCS 50101

THE UNIVERSITY OF CHICAGO

BREAKOUT EXERCISES

Exercise 1

The following is a table of comparison and logical operators. Use to evaluate the following statements.

Python Comparison Operators

Comparison operators are used to compare two values:

Operator	Name	Example
==	Equal	x == y
!=	Not equal	x != y
>	Greater than	x > y
<	Less than	x < y
>=	Greater than or equal to	x >= y
<=	Less than or equal to	x <= y

Python Logical Operators

Logical operators are used to combine conditional statements:

Operator	Description	Example
and	Returns True if both statements are true	x < 5 and x < 10
or	Returns True if one of the statements is true	x < 5 or x < 4
not	Reverse the result, returns False if the result is true	not(x < 5 and x < 10)

MODULE PREVIEW

© T.A. BINKOWSKI, 2020

MODULE 2
MPCS 50101

THE UNIVERSITY OF CHICAGO

MODULE PREVIEW

- Variables, Expressions and Statements
- Control Flow and Conditional Execution
  - Boolean Expression
  - Conditional Statements
- Functions
  - Variable Arguments
- Iteration
  - For Loops
  - While Loops
- Strings

LET'S LEARN TO PROGRAM

MODULE PREVIEW

- Get the 🍿ready.

Module 2: Functions
5 hours ago

Module 2: Iteration
7 hours ago

Module 2: Strings
8 hours ago

Module 2: Variables, Expressions and Statements
16 hours ago

Module 2: Control Flow and Conditional Execution
16 hours ago

A PYTHON PROGRAM

A PYTHON PROGRAM

- Calculate the batting average for Johnny Ballgame. He has been to the play 100 times this season, has been walked 23 times, and has 41 hits. Print his stats in the following format:

  ```
  Johnny Ballgame's batting average is .###!
  ```

A PYTHON PROGRAM

```
# Exercise 1
#
# Andrew Binkowski


player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23
```

WHAT IS THIS FILE?

WHO IS THE AUTHOR

INFORMATIVE VARIABLE NAMES

A PYTHON PROGRAM

```
# Exercise 1
#
# Andrew Binkowski


player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearances - number_of_walks
```

CITE SOURCES FOR OUTSIDE INFORMATION

COMMENT FOR SOMEONE ELSE WHO MIGHT NOT KNOW THIS

RELAVANT VARIABLE NAME

A PYTHON PROGRAM

```
# Andrew Binkowski

player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearances - number_of_walks

# Calculate batting average
batting_average = number_of_hits/number_of_plate_appearances

print(batting_average)
```

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
0.41
```

A PYTHON PROGRAM

```
# Andrew Binkowski


player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appe

# Calculate batting average
batting_average = number_of_hits/number_of_plate_appearances

print(batting_average)
```

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
0.41
```

COMPATIBLITY ISSUE

PYTHON 2: 100/41 = 2
PYTHON 3: 100/41 = 0.41

A PYTHON PROGRAM

```
# Andrew Binkowski

player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossa
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearances - numb

# Calculate batting average
batting_average = number_of_hits/number_of_plate_appearances

print(batting_average)
```

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
0.41
```

SEMANTIC ERROR

READABILITY

A PYTHON PROGRAM

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
0.5324675324675324
```

```
# Andrew Dinkowski

player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearar

# Calculate batting average
batting_average = float(number_of_hits) / float(number_of_at_bats)

print(batting_average)
```

CONVERT TO FLOAT

READABILITY

A PYTHON PROGRAM

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
0.5324675324675324
```

```
# Andrew Dinkowski

player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearar

# Calculate batting average
batting_average = float(number_of_hits) / float(number_of_at_bats)

print(batting_average)
```

CONVERT TO FLOAT

READABILITY

A PYTHON PROGRAM

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
Johnny Ballgame's batting average is 0.532!
```

```
player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearances - number_of_walks

# Calculate batting average
batting_average = float(number_of_hits) / float(number_of_at_bats)

# Print output
print("%s's batting_average is %.3f" % (player_name, batting_average))
```

THIS IS CORRECT
(BUT WILL IT ALWAYS BE)

DIFFERENT WAYS TO FORMAT OUTPUT

A PYTHON PROGRAM

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
Johnny Ballgame's batting average is 0.532!
```

```
player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = number_of_plate_appearances - number_of_walks

# Calculate batting average
batting_average = float(number_of_hits) / float(number_of_at_bats)

# Print output
print("%s's batting_average is %.3f" % (player_name, batting_average))
```

THIS IS CORRECT

BUT WILL IT ALWAYS BE?

WHAT IS THE CORRECT BEHAVIOR THAT I WANT TO HAPPEN?

DIFFERENT WAYS TO FORMAT OUTPUT

A PYTHON PROGRAM

```
(base) andrew@Andrews-MacBook-Pro administrative % python example1.py
Johnny Ballgame's batting average is 0.532!
```

```
player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

# Calculate the batting average (http://mlb.com/glossary/)
# Walks do not count as an "at bat"
number_of_at_bats = numbe

# Calculate batting average
batting_average = float(number_of_hits) / float(number_of_at_bats)
batting_average_rounded = round(batting_average,3)

# Print output
print("%s's batting_average is %.3f!" % (player_name, batting_average_rounded))
```

BUILT-IN FUNCTION IN PYTHON

A PYTHON PROGRAM

```
def calculate_batting_average(hits, appearances, walks):
    # Calculate the batting average (http://mlb.com/glossary/)
    # Walks do not count as an "at bat"
    number_of_at_bats = number_of_plate_appearances - number_of_walks

    # Calculate batting average
    batting_average = float(number_of_hits) / float(number_of_at_bats)
    batting_average_rounded = round(batting_average,3)
    return batting_average_rounded


player_name = "Johnny Ballgame"
number_of_hits = 41
number_of_plate_appearances = 100
number_of_walks = 23

batting_average = calculate_batting_average(number_of_hits, \
                                            number_of_plate_appearances, \
                                            number_of_walks)

# Print output
print("%s's batting_average is %.3f!" % (player_name, batting_average))
```

RELATED CODE IN FUNCTION

FUNCTION DEFINED BEFORE CALLED

KEYBOARD INPUT

KEYBOARD INPUT

- Python has a built-in function called `input`
  - Prompts user for text entry
  - Returns what the user types as a string

```
name = input("What is your name?")
print("Hello " + name)


# What is your name? Andrew
# Hello  Andrew
```

KEYBOARD INPUT

```
name = input("What is your name?")
age = input("How old are you?")

print("Hello " + name)
print("Next year you will be ", age + 1)
```

```
Traceback (most recent call last):
  File "workspace.py", line 5, in <module>
    print("Next year you will be ", age + 1
TypeError: cannot concatenate 'str' and 'int' objects
```

KEYBOARD INPUT

CONVERT STRING TO A NUMBER

```
name = input("What is your name? ")
age = input("How old are you? ")

# Convert string age to a integer
age_number = int(age)

print("Hello " + name)
print("Next year you will be ", age_number + 1)

# What is your name? Oscar
# How old are you? 7
# Hello Oscar
# You are 8
```

KEYBOARD INPUT

```
# Calculate the area of the square
length = input("Enter length: ")
width = input("Enter width: ")

# Remember length and width are strings
length_int = int(length)
width_int = int(width)

# Calculate the area
area = length_int * width_int

print("Area equals: ", area)
```

CREATE A NEW INTEGER FROM A STRING

CODING EXAMPLES

CODING EXAMPLES

Write a program to determine if an inputed number is even or odd.

CODING EXAMPLES

```
num = input("Enter a number: ")
mod = int(num) % 2

if mod > 0:
    print("You picked an odd number.")
else:
    print("You picked an even number.")
```

THE % SYMBOL IS CALLED THE **MODULO OPERATOR**

IT RETURNS THE REMAINDER OF DIVIDING THE LEFT HAND OPERAND BY RIGHT HAND OPERAND

CODING EXAMPLES

```
num = input("Enter a number: ")
mod = int(num) % 2

if mod > 0:
    print("You picked an odd number.")
else:
    print("You picked an even number.")
```

EVEN NUMBERS WILL HAVE REMAINDER OF 0

ODD NUMBERS WILL HAVE REMAINDER OF 1

IF MOD == 0:

CODING EXAMPLES

Write a program to calculate the length of a string?

CODING EXAMPLES

```
def string_length(string):
    count = 0
    for char in string:
        count += 1
    return count


length = string_length('hello world')

print("The string is %d characters long" % length)
```

CODING EXAMPLES

Write a program to guess a random number between 1 and 20.

CODING EXAMPLES

```
import random
random_number = random.randint(1, 20)

print('I am thinking of a number between 1 and 20.')

while True:
    string_input = input('>  ')
    guess = int(string_input)
    if guess == 1:
        break

print('You guessed it!')
```

PROGRAMMING PATTERN:
WILL REPEAT FOREVER
UNTIL “BREAK”

Assignment

ASSIGNMENT

Problem 1

Create a temperature converter program that takes input from a user in Fahrenheit and converts it to Celsius. You should check that the input is valid (ie. the input is a number type).

The program should behave similar to the example below:

Example:

```
> Enter a temperature in Fahrenheit: 32
It is 0 in Celsius.
```

Problem 2

THE END

©TTAA.BB INKOWSKI, 2019

MPCS 50101
MODULE 2

THE UNIVERSITY OF
CHICAGO