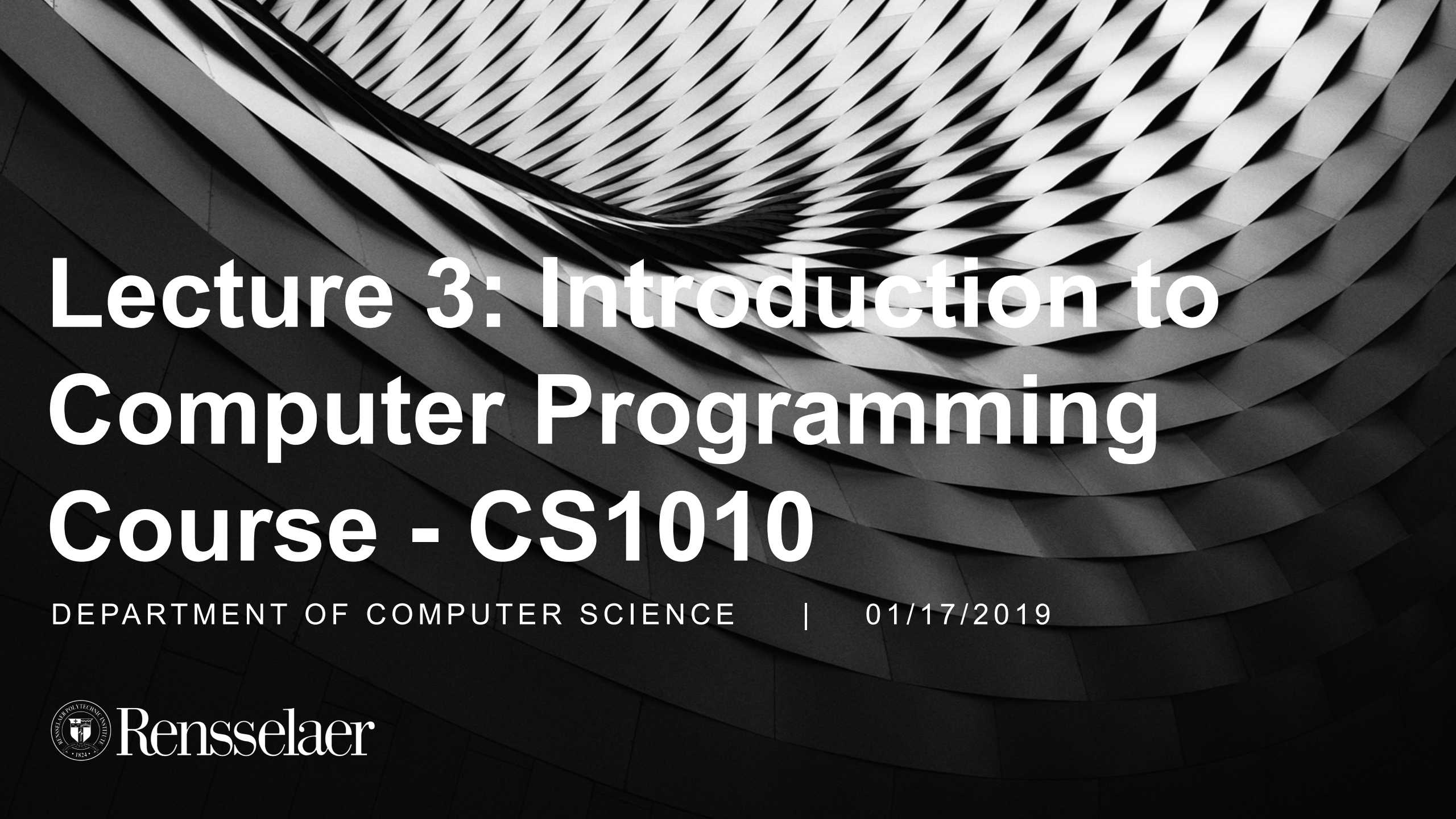

Lecture 3: Introduction to Computer Programming Course - CS1010

DEPARTMENT OF COMPUTER SCIENCE | 01/17/2019

Rensselaer

Announcements

- We will use Submitty (submitty.cs.rpi.edu) for course content and homework submission.
  - Log in with your RCS Account and Password
- Usually all home-works are posted on Mondays
- Homework 1 was posted 1 day late and the due date is extended by 1 day.
- From next week onwards my office hours:
  - Monday, Thursday: 12 pm to 1:30pm (updated on syllabus)

Goals for Today

- Talk about Booleans
- Elementary Boolean Algebra
- If, Elif and Else Statements in Python (Also called control statements)

Object Types (Table from Previous Lecture)

Name	Type (representation)	Example
Integers	int	Whole Numbers: 1, 5 , 7500
Floating Point	float	Decimal: 2.3, 4.6, 23.15
Strings	str	Ordered sequence of characters: “hello” “Sam” “2000”
Lists	list	Ordered sequence of objects: [10, “hello”, 500.5]
Dictionary	dict	Unordered Key Value pairs: {“mykey”:”Value”, “place”: “New York”}
Tuples	tup	Ordered immutable sequence of objects: (100,”Hello”, 20.5)
Sets	set	Unordered collection of unique objects: {“a”,”b”}
Booleans	bool	Logical Value: True, False

Booleans

- Boolean represents logical values (TRUE or FALSE)
- The **bool()** method is used to return or convert a value to a Boolean value
- The bool() method in general takes only one parameter, on which the standard truth testing procedure can be applied.
- **If no parameter is passed, then by default it returns False**.

Basic Boolean Algebra

- **Boolean Algebra** is a branch of algebra that involves bools, or true and false values.
- They're typically denoted as ***T or 1 for true*** and ***F or 0 for false***.
- Using this simple system we can boil down complex statements into easier/understandable logical statements.

Truth Table

- A **truth table** is a way of organizing information to list out all possible scenarios.
- p denotes proposition (condition) then ~p (read as not p) means everything opposite of the proposition.

p	~p
T	F
F	T

Binary Operators

- AND Operator
  - *Requires* both p and q to be True for the result to be True. All other cases result in False.

p	q	P AND q
T	T	T
T	F	F
F	T	F
F	F	F

- Keyword in Python: **and**

- OR Operator
  - *Requires* only one proposition to be True for the result to be True.

p	q	P OR q
T	T	T
T	F	T
F	T	T
F	F	F

- Keyword in Python: **or**

Operators and Expressions (In Python)

Operators	Expressions	Example
==	If the two operands are equal then the condition will be true	x=3, y=5; (x==y) is not true.
!=	If the two operands are not equal then the condition is true	(x!=y) is true
>	If the value on the left is greater than that on the right, then the condition is true	(x>y) is not true.
<	If the value on the right is greater than that on the left, then the condition is true	(x<y) is true
>=	If the value on the left is greater than or equal to the one on the right, then the condition is true	(x>=y) is false
<=	If the value on the right is greater than or equal to the one on the left, then the condition is true	(x<=y) is true

Important key words

- **True** : This keyword is used to represent a Boolean true. If a statement is true, “True” is printed.
- **False** : This keyword is used to represent a Boolean false. If a statement is False, “False” is printed.
- **None** : This is a special constant used to **denote a null value or a void**. **Its important to remember, 0, any empty container(e.g empty list) do not compute to None**.
- **not** : This logical operator **inverts the truth value** (For example not True will return False and vice versa).

In- Class Exercise (from previous class)

- Write a Python Script that takes user input as two numbers and performs a comparison of whether the numbers are equal or not.
  - Output ‘FALSE’ if they are equal and ‘TRUE’ if they are not.
  - Output ‘TRUE’ if they are equal and ‘FALSE’ if they are not.

If Statements in Python

- if Statements in Python allow us to tell the computer to perform alternative actions based on a certain set of results.
- Verbally, we can imagine we are telling the computer:
  - "Hey if this case happens, perform some action"
  - We can then expand the idea further with elif and else statements, which allow us to tell the computer:
  - "Hey if this case happens, perform some action. Else, if another case happens, perform some other action. Else, if *none* of the above cases happened, perform this action."
- NOTE: It is important to keep a good understanding of how indentation works in Python to maintain the structure and order of your code. We will talk about this topic again when we start building out functions!

If Statements in Python

- Most commonly used control flow statements.
- **Python Syntax**
- if *condition* :
- indented Statement Block
- Let’s Try this code:

```
weight = float(input("How many pounds does your suitcase weigh? "))
if weight > 50:
    print("There is a $25 charge for luggage that heavy.")
    print("Thank you for your business.")
```

If Else Statements

```
temperature = float(input('What is the temperature? '))
if temperature > 70:
    print('Wear shorts.')
else:
    print('Wear long pants.')
print('Get some exercise outside.')
```

- There is Elif statement if more than 1 condition needs to be tested.

```
loc = 'Bank'
if loc == 'Auto Shop':
    print('Welcome to the Auto Shop!')
elif loc == 'Bank':
    print('Welcome to the bank!')
else:
    print('Where are you?')
```

Nested Ifs

- if x:
- 	if y:
- 		code-statement
- else:
- 	another-code-statement

- Python is so heavily driven by code indentation and whitespace.
- This means that code readability is a core part of the design of the Python language.
-

Some Practice Problems

- To be finished in class.
- Today’s set is not graded.
- From next practice class onwards (which is in 2 weeks from now)- You can be asked to submit a code online in-class.
- You can work in teams for all in-class grade-able exercises. However, each student needs to submit their work when asked.

Reminding the Methodology (Lecture 1)

- U – Understand the Problem: Write down the inputs you have
- D – Devise a Good Plan to Solve: Write down the Algorithm you will use
- I – Implement the Plan: Translate Algorithm to code
- E – Evaluate the Solution: Run for a few test cases

Problem 1

- Write a program that asks the user for their name and greets them with their name.

Problem 2

- Modify the previous program such that only the users Alice and Bob are greeted with their names.

Problem 3

- Write a program that asks the user for a number n and gives them the possibility to choose between computing the sum or computing the product of n and n-1.

Problem 4

- Write a program that prints whether a user provided number is an even number or not.

Problem 5

- On my birthday I am planning to invite n friends and distribute some chocolates to all of them. At the chocolate shop I found each packet contains different number of m chocolates.
- Write a program that calculates whether a given packet will distribute all chocolates evenly to my friends or not. In addition the program must also tell me how many will be in surplus or short if I buy a particular packet.

Next Class

- Strings
- String Manipulation