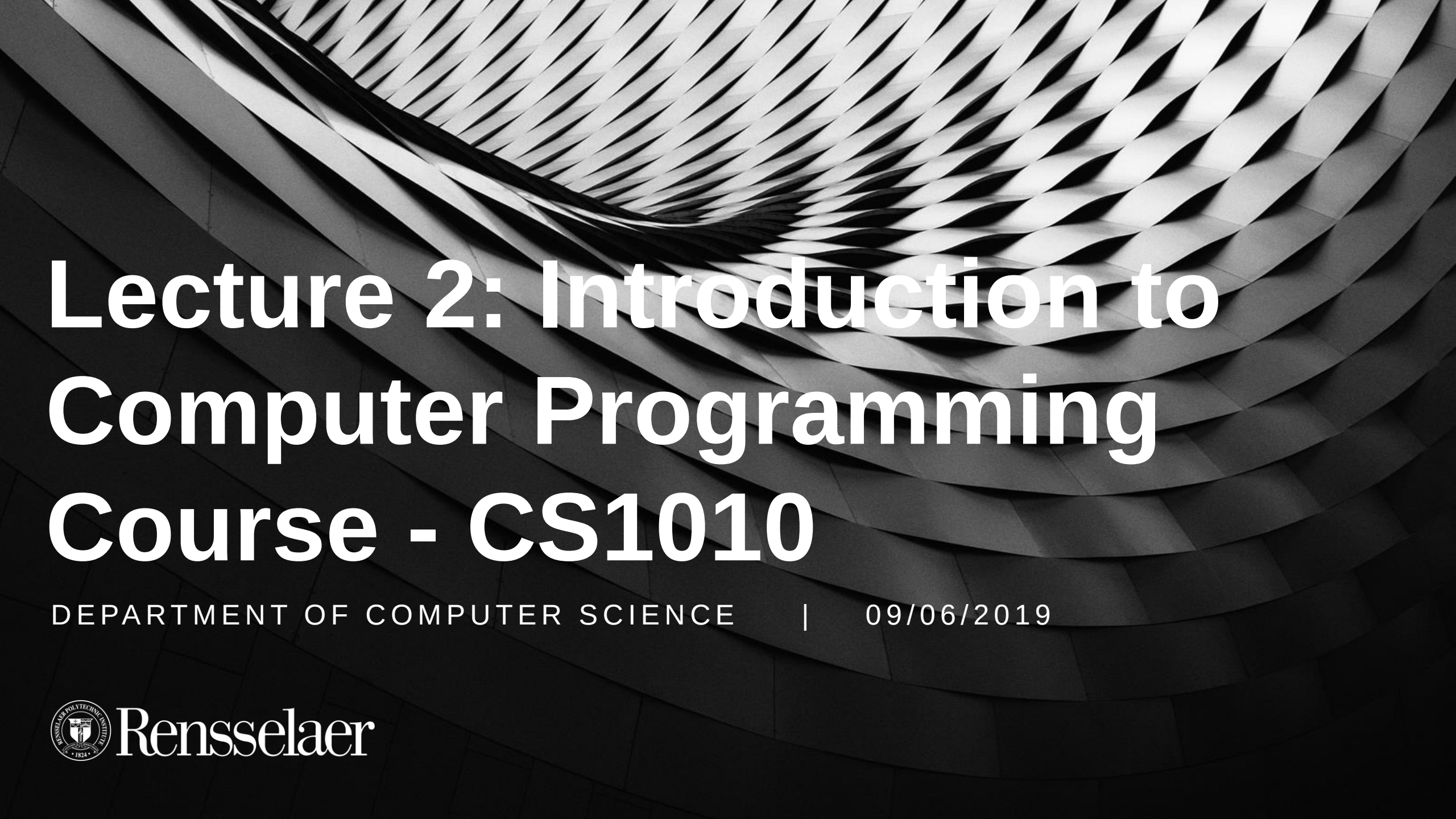


Lecture 2: Introduction to Computer Programming Course - CS1010

DEPARTMENT OF COMPUTER SCIENCE | 09/06/2019

Rensselaer

Announcement

- 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 Tuesdays
- Homework 1 will be posted tonight (Friday)
- Due date will be Tuesday the 17th .

Goals

- Install Miniconda/Anaconda on our Machines
- Get Started with Spyder
- Python as a Calculator
- Variables and Expressions
- Comparison Operators

Tooling Up

To Install Python we will use either

The free Anaconda Distribution or Miniconda Distribution.

Anaconda/Miniconda Can Be Installed on any major OS : Windows, Linux or MacOS.

Anaconda includes Python as well as many other useful Libraries including Spyder that we will use in this course.

Can use other IDEs (Integrated Development Environment) like Jupyter or IDLE.

Which One to Select?

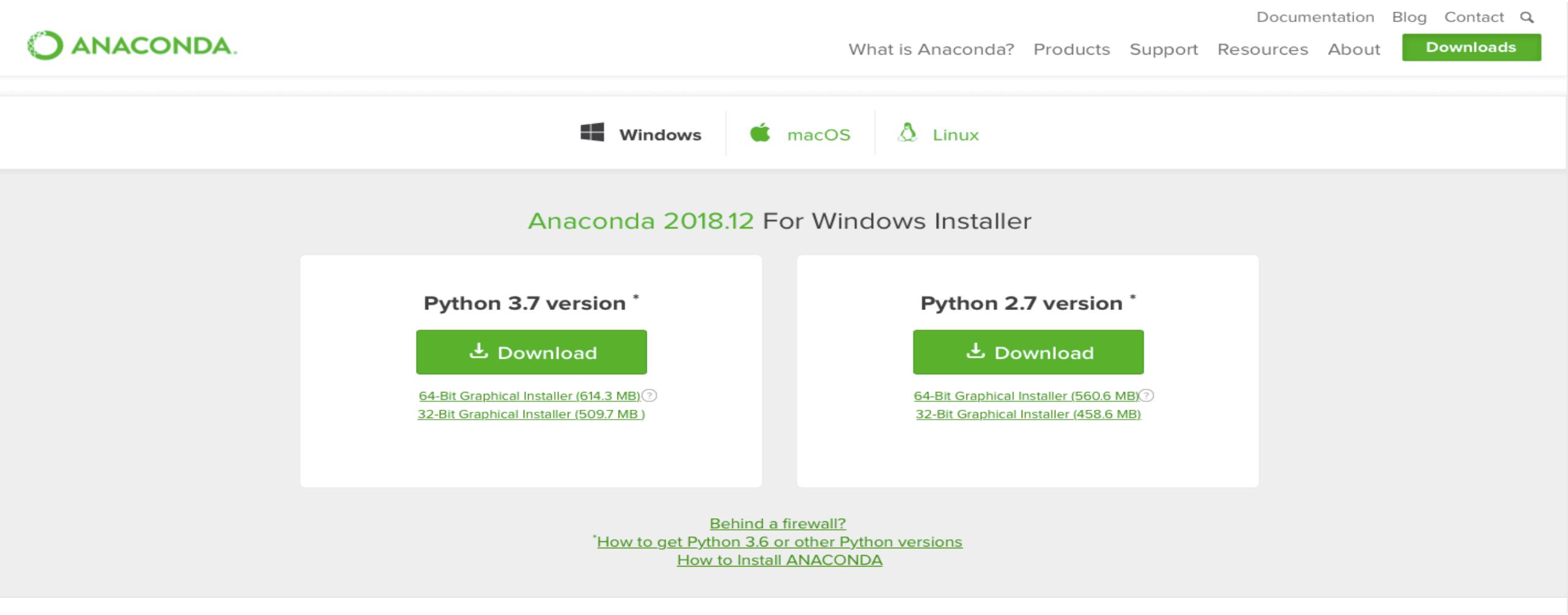
- Choose **Anaconda** if you:
  - Are new to conda or Python.
  - Like the convenience of having Python and over 150 scientific packages automatically installed at once.
  - Have the time and disk space—a few minutes and 300 MB.
  - Do not want to individually install each of the packages you want to use.
- Choose **Miniconda** if you:
  - Do not mind installing each of the packages you want to use individually.
  - Do not have time or disk space to install over 150 packages at once.
  - Want fast access to Python and the conda commands and you wish to sort out the other programs later

Reference: https://conda.io/docs/user-guide/install/download.html#anaconda-or-miniconda

Course Software Setup

1. Install Anaconda

Download Anaconda:
www.anaconda.com/downloads



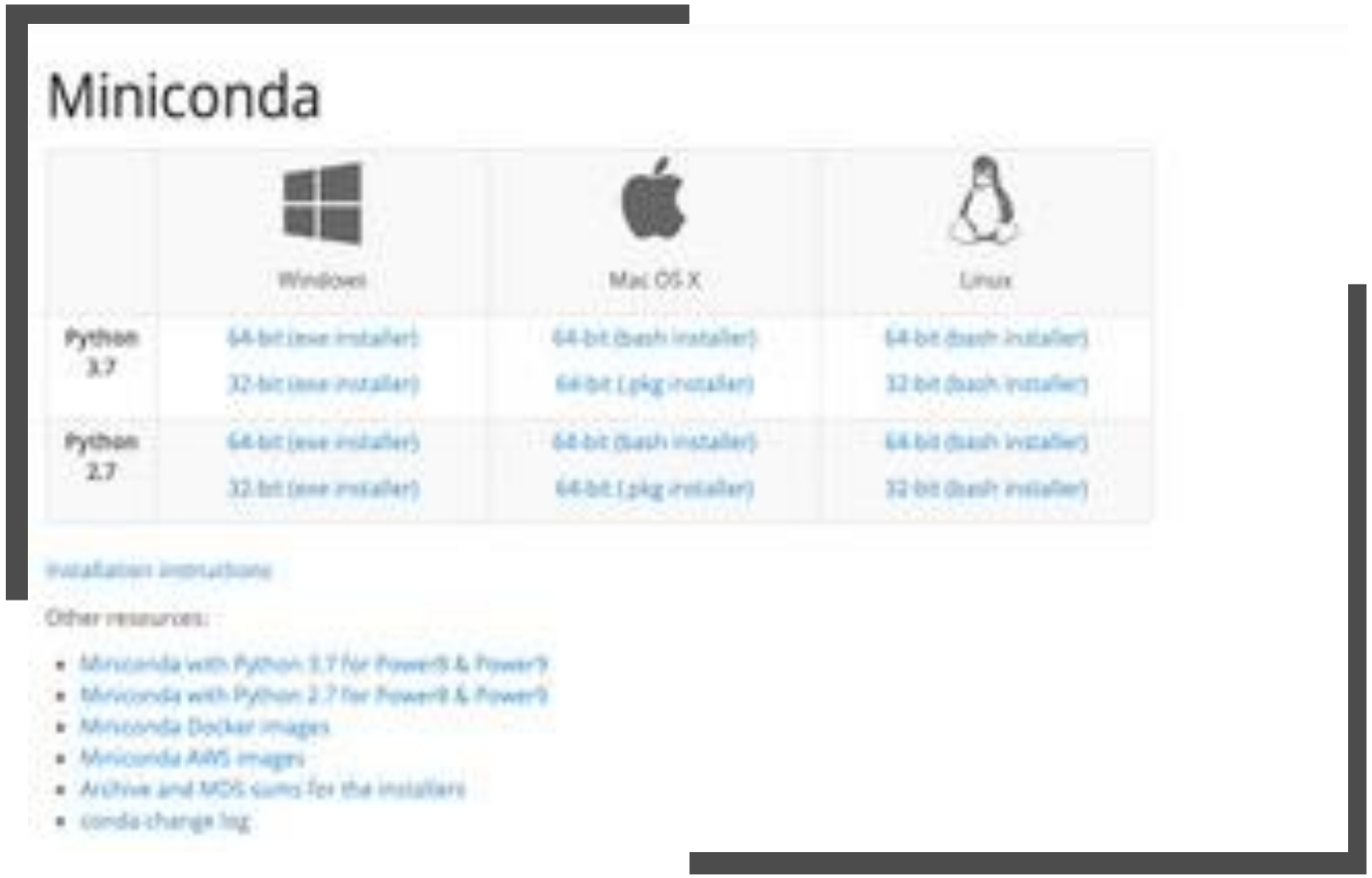
Select **Windows, Mac or Linux.**
Follow the instructions (Wizard will guide you through the process)
Guide : http://docs.anaconda.com/anaconda/install/ windows/

To Get Spyder

- On the **Windows** Start Menu:
  - Go to Anaconda Navigator
  - Click Spyder
- On **Mac:**
  - Go to Anaconda App through Launchpad
  - Click Spyder

Install Miniconda

Install Miniconda



- https://conda.io/miniconda.html

Install (Windows)

- Click on the downloaded file to start the installation. When the installer window appears,
  - Click *Next* to get started,
  - Click *I Agree* to accept the license terms, and
  - Then click *Next* to accept the defaults for the next several screens.
  - When you reach the screen with the *Install* button, verify the the two Advanced Options checkboxes to Add Anaconda to my PATH environment variable and to Register Anaconda as my default Python 3.7 are both checked. Then click *Install*.
  - When the install finishes, click *Next* then *Finish*. You can ignore the Anaconda Cloud window that pops up in your browser.

Install (Mac)

- Click on OS X Miniconda install
- Click on the Miniconda installer for OS X
  - On this page select the *64-bit (.pkg installer)* for *Python 3.7* and Mac OS X
  - Wait for it to download. It will place a file called **Miniconda3-latest-MacOSX-x86_64.pkg** in your downloads folder. On my computer using Safari this is ~/Downloads
  - Double click on the file to begin the installation
  - Follow the steps on the installer
  - Select the location you want Minicinda3 to be saved.

For Miniconda Get Spyder

- For Windows
  - In the start menu open Anaconda Prompt
  - Type conda install spyder
    - Will take some time to download
  - Proceed ([y]/n)? y
  - Go to the start menu again and click on Spyder

- For Mac: Go to the Terminal window (In Applications)
  - Uzmas-MacBook-Air:~ uzma$ cd ~/Applications
  - Uzmas-MacBook-Air:Applications uzma$ conda install spyder
    - Will take some time to download
  - Proceed ([y]/n)? y
  - Uzmas-MacBook-Air:Applications uzma$ which spyder
    - Usually in the bin folder within miniconda3

Spyder

- For detailed information go to:

- https://docs.spyder-ide.org

Before we begin

- Please make sure all your installations are correctly done.
- If not then please bring your computers to the instructor, TA or mentor for help.

Basic Elements of Python

- A Python **program** is called a **script**: It is a sequence of definitions and commands.
- These definitions are evaluated and the commands are executed by the Python **interpreter** in something called the **shell**.
- A new shell is created whenever execution of a program begins.
- A **command**, often called a **statement**, instructs the interpreter to do something.

Example: print (‘Hello World!’)

Is a command that makes the interpreter print:

Hello World!

Jump in to our first Program!

- We create a file called hello.py containing just the two lines of Python code:
  - print('Hello, World!’)
  - print('This is Python')

Objects and Operators

- **Objects** are the core things that Python programs manipulate.

- Every object has a **type** that defines the kinds of things that programs can do with objects of that type.

Object types for numeric data

Name	Type (representation)	Example
Integers	int	Whole Numbers: 1, 5 , 7500
Floating Point	float	Decimal: 2.3, 4.6, 23.15

Types of numbers

- Python has various "types" of numbers. We'll mainly focus on integers and floating point numbers.
- Integers are just whole numbers, positive or negative. For example: 2 and -2 are examples of integers.
- Floating point numbers in Python are the ones that have a decimal point in them, or use an exponential (e) to define the number.
- Throughout this course we will be mainly working with integers or simple float number types.
- Lets do some practice in Spyder.

Numbers

We'll learn about the following topics:

1.) Types of Numbers in Python
2.) Basic Arithmetic
3.) Differences between classic division and floor division
4.) Assignment in Python

Variable assignment

- In this section:

  - We've seen how to use numbers in Python as a calculator let's see how we can assign names and create variables.
  - Variables are placeholders in Python that hold a value
  - These can be referred to again and again in a program

Assigning and Re-Assigning Variables

- Variable assignment follows `name = object`, where a single equals sign `=` is an ‘assignment operator’
- Python lets you re-assign variables with a reference to the same object using a short cut.
- Python lets you add, subtract, multiply and divide numbers with reassignment using `+=`, `-=`, `*=`, and `/=`.
- Let us try in Spyder.

Dynamic Typing

- Python uses ‘dynamic typing’, meaning you can reassign variables to different data types.
- This makes Python very flexible in assigning data types; it differs from other languages that are ‘statically typed’.
- Lets check in Spyder

Variable Names

- The names you use when creating these labels need to follow a few rules:
  1. Names can not start with a number.
  2. There can be no spaces in the name, use _ instead.
  3. Can't use any of these symbols :'",<>/?|\()!@#$%^&*~-+
  4. It's considered best practice that names are lowercase.
  5. Avoid using the characters 'l' (lowercase letter el), 'O' (uppercase letter oh), or 'I' (uppercase letter eye) as single character variable names.
  6. Avoid using words that have special meaning in Python like "list" and "str"
- Using variable names can be a very useful way to keep track of different variables in Python. For example: (Go to Spyder example)

Determine Variable Type

- You can check what type of object is assigned to a variable using Python's built-in `type()` function. Common data types include:
- int (for integer)
- Float (for decimals)

User Input

- User enters the values in the Console and that value is then used in the program as it was required.
- To take input from the user we make use of a built-in function *input()*.
- Let’s try in Spyder.

For Next Lecture

- Now you have the necessary tools required to write programs that involve calculations