IDX EXCHANGE

INTRO TO PYTHON PROGRAMMING

- WEEK 1 - S1 COURSE OVERVIEW AND PROGRAMMING BASICS
- WEEK 1 - S2 IDE AND GITHUB SETUP

July 2024

IDX EXCHANGE

WELCOME TO IDX EDUCATION

ABOUT US

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Amet tellus cras adipiscing enim eu turpis egestas. Fames ac turpis egestas maecenas pharetra convallis posuere. Adipiscing bibendum est ultricies integer quis auctor. Venenatis urna cursus eget nunc scelerisque viverra.

IDX EXCHANGE

Summer Course Overview

Course Structure

Hour-long Bi-weekly online class meetings

- Featuring Lab work and practical applications of pre-class material
- Requires self-exposure to topics via course provided online resources

Weekly Homework

- Guided steps of practical learning that will lead up to a capstone project

Intro to Python Course

July - August

Week 1:
Visual Studio and Github

Week 2:
Control Structures and Running a script

Week 3:
Functions and Modules

Week 4:
Data Structures

IDX EXCHANGE

Summer Course Overview

Course Structure

Hour-long Bi-weekly online class meetings

- Featuring Lab work and practical applications of pre-class material
- Requires self-exposure to topics via course provided online resources

Weekly Homework

- Guided steps of practical learning that will lead up to a capstone project

Intro to Python Course

September

Week 5:
File Handling and Command Line

Week 6:
Object Oriented Programming

Week 7:
Libraries and Frameworks

Week 8:
Final Project Social

IDX EXCHANGE

What is Programming?

Programming is the process of creating a set of instructions that a computer can understand and execute. These instructions, also known as code, tell the computer how to perform specific tasks, from simple calculations to complex operations. Programming allows us to solve problems, automate tasks, and create applications that can range from simple games to advanced scientific simulations.

Programming Languages

Programming languages are formal languages comprised of a set of instructions that produce various kinds of output. They are used to implement algorithms and control the behavior of computers. Each programming language has its syntax, rules, and unique features, making it suitable for specific tasks

IDX EXCHANGE

Programming Languages

Feature	Python	JavaScript	C	Java	PHP
Ease of Learning	Easy	Moderate	Moderate to Difficult	Moderate	Moderate
Primary Use	General-purpose	Web development	System programming	Enterprise, Android apps	Web development
Syntax	Readable, concise	C-like, flexible	Low-level, less intuitive	C-like, structured	C/Perl-like, flexible
Performance	Moderate	Moderate	High	High	Moderate
Community	Large, supportive	Large, supportive	Moderate	Large, supportive	Large, supportive

Pseudocode

Pseudocode

Pseudocode is a simplified, informal way of writing algorithms using plain language. It is not bound by the syntax rules of any specific programming language, making it easier to understand and communicate ideas. Pseudocode helps programmers plan and design their code before writing the actual code in a programming language.

Why Use Pseudocode?

Clarity: Pseudocode allows you to express complex ideas in a clear and concise manner without worrying about syntax errors.
Planning: It helps in planning and visualizing the structure of the program before implementation.
Communication: Pseudocode is an excellent tool for communicating algorithms and logic to others, including those who may not be familiar with specific programming languages.

Examples

```
START
  Declare variables num1, num2, sum
  Set num1 to 5
  Set num2 to 3
  Calculate sum as num1 + num2
  Print sum
END
```

IDX EXCHANGE

Foundations of Programming

At the lowest level, Machine code describes the interaction between software and the CPU. But most importantly, the fundamental operations include:

`ADD` `MOVE` `JUMP` `LOAD`

Translating this to python functionality, variables and arithmetic operators are essential to any algorithm or script.

IDX EXCHANGE

Variables

In programming, variables are used to store data that can be changed and manipulated throughout the execution of a program. Think of variables as containers or storage boxes that hold different types of information, such as numbers, text, or more complex data structures.

- Storage: Variables allow you to store data that your program can use later.
- Manipulation: You can perform operations on the data stored in variables, such as calculations or modifications.
- Reusability: Variables can be used multiple times within a program, making your code more efficient and readable.

Declare Variables

Declaring a variable means giving it a name and assigning it a value. Different programming languages have different rules for declaring variables, but the basic idea remains the same.

```
age = 16
name = "Alice"
is_student = True
```

IDX EXCHANGE

Python Variables

```
num1 = 10
num2 = 5
sum = num1 + num2
difference = num1 - num2
print("Sum is", sum)
print("Difference is", difference)
```

Data Types

Variables can hold different types of data.
The most common data types include:

- **Integers**: Whole numbers, such as 1, 2, 100
- **Floats**: Decimal numbers, such as 3.14, 2.71
- **Strings**: Sequences of characters, such as "Hello, World!"
- **Booleans**: Logical values, True or False

IDX EXCHANGE

Using Variables of Different Types

```
num1 = 10
num2 = 5
sum = num1 + num2
difference = num1 - num2
print("Sum is", sum)
print("Difference is", difference)
```

```
first_name = "John"
last_name = "Doe"
full_name = first_name + " " + last_name

print("Full Name:", full_name)
# Output: Full Name: John Doe
```

IDX EXCHANGE

Variable Naming Conventions

When naming variables, it’s important to follow certain rules to ensure your code is readable and error-free:

```
# Valid variable names
student_age = 16
StudentAge = 16
_student_age = 16
studentAge1 = 16

# Invalid variable names
1student_age = 16  # Starts with a number
student-age = 16  # Contains a special character
student age = 16  # Contains a space
```

IDX EXCHANGE

More on Arithmetic Operators

What are Arithmetic Operators?

Arithmetic operators are symbols used in programming to perform mathematical operations on variables and values. They allow you to carry out basic arithmetic tasks like addition, subtraction, multiplication, and division, as well as more complex operations like exponentiation and modulus.

Common Arithmetic Operators

Addition (+): Adds two numbers.

Subtraction (-): Subtracts one number from another.

Multiplication (*): Multiplies two numbers.

Division (/): Divides one number by another.

Modulus (%): Returns the remainder of a division.

Exponentiation (**): Raises one number to the power of another.

Floor Division (//): Divides one number by another and rounds down to the nearest whole number.

IDX EXCHANGE

```
# Complex expression
a = 10
b = 5
c = 2
result = a + b * c - b / c
print("Result:", result)
```

What is the result?

IDX EXCHANGE

Today’s Goals

Setup - VSCode

Step 1 :
Download Visual Studio Code

Step 2 :
Installing Packages

Step 3:
Logic and Programming

IDX EXCHANGE

Step 1: Download Visual Studio Code

VSCode is a IDE that easily integrates extensions for programming languages.

https://code.visualstudio.com/download

Step 2: Create a Github Account

- GitHub is a web hosting platform for managing git repositories. Github is often used to provide a web interface for file management, user management, as well as online web hosting services.
- Git is an open-source tool used for version control. Using a command line interface, git provides functionality to edit repositories for local or online use.

https://github.com

Step 3: Download Git via VSCode Terminal

https://git-scm.com/book/en/v2/Getting-Started-Installing-Git

In VSCode, open up a terminal to install git by entering the commands provided that correspond to your machine

IDX EXCHANGE

Step 4: Github connection

Because this is a crucial step, the lab will include office hours for debugging.

IDX EXCHANGE

Step 4a: Github Config

In order to remotely access and edit your repositories, it needs to know what GitHub account to use!
Skip this if you already have git configured

Steps (type these in carefully!):

1. Open up a new terminal in VSCode
2. Type up the following commands:
    - git config --global user.name “First Last”
    - git config --global user.email “youremail@address.edu”

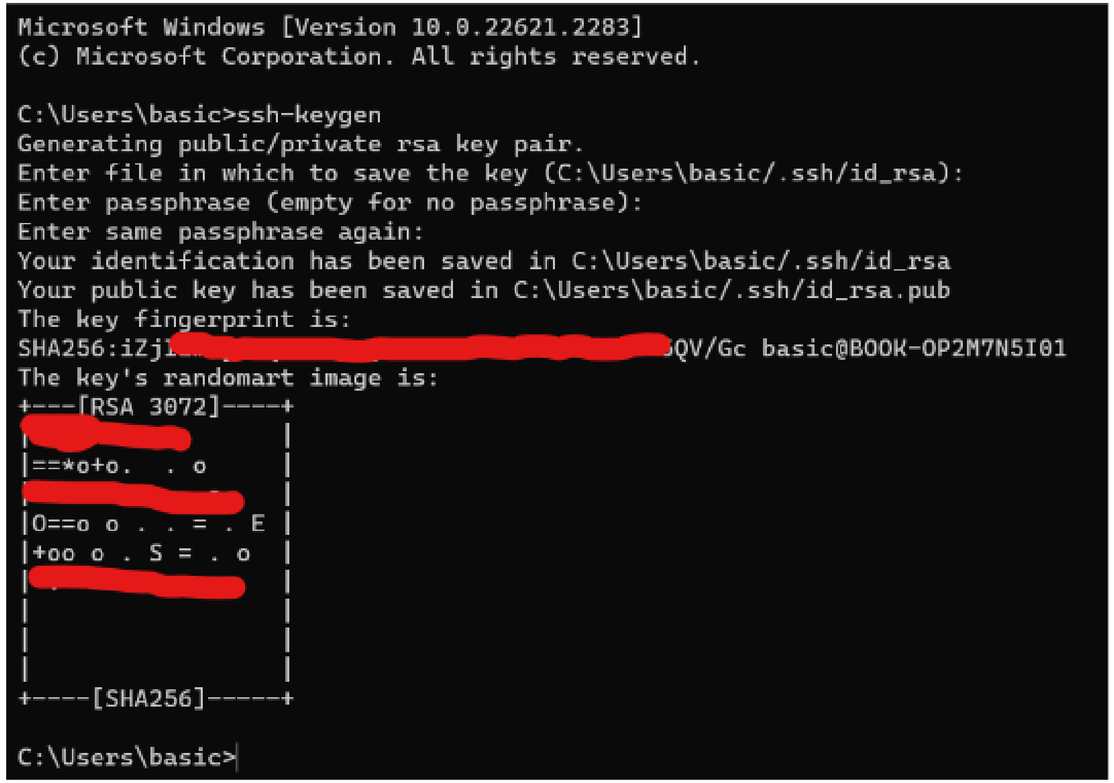
This must be the email you used to create your account!

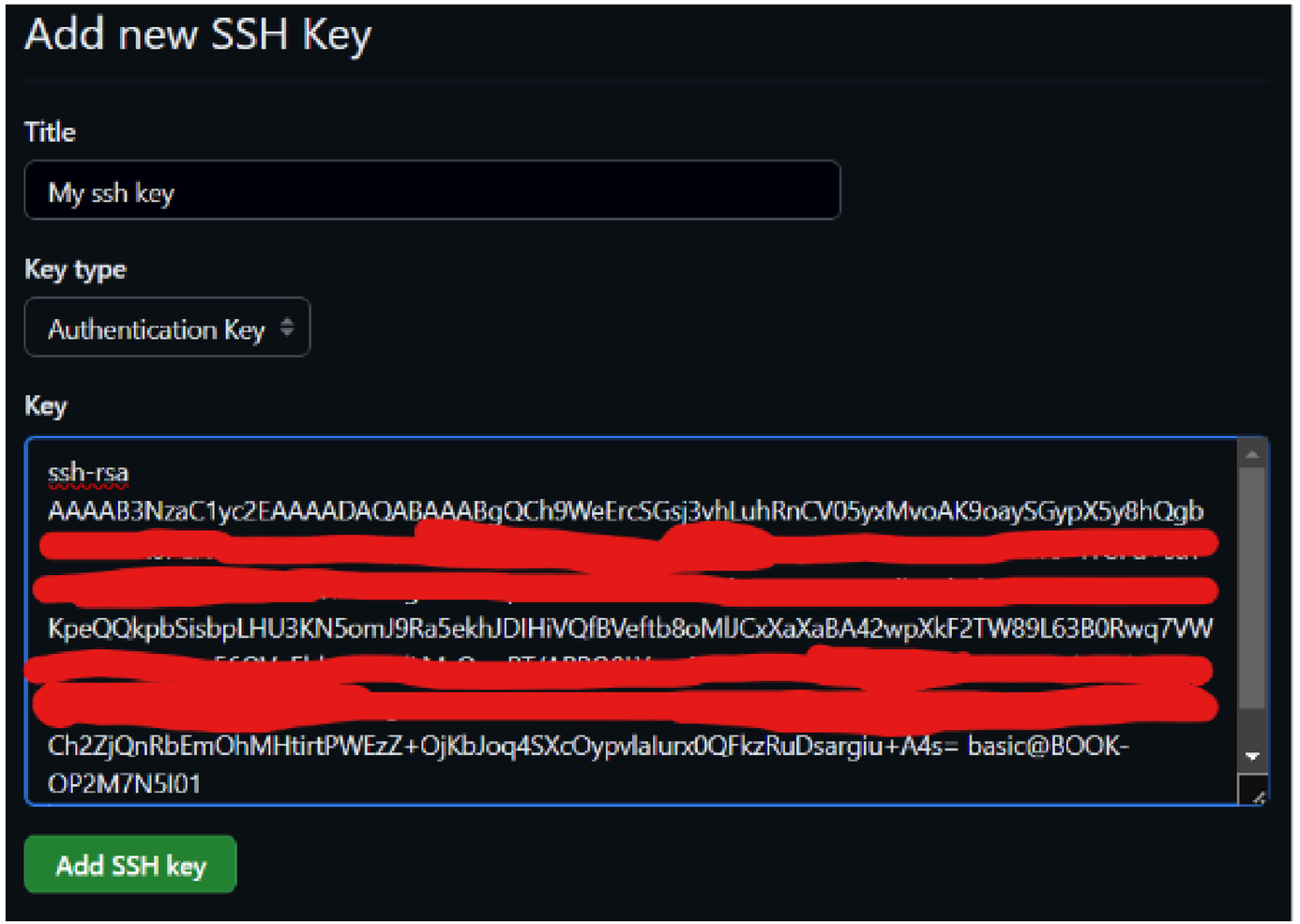
Try running git config --global user.email to verify!

IDX EXCHANGE

Step 4b: Github SSH Connection

1. Make an ssh key with ssh-keygen
   - Hit “enter” to skip destination/password
   - If it says you already have one, use that!
2. Go to https://github.com/settings/keys
   - Click “New ssh key” and set some title
   - Open “.ssh/id_rsa.pub” (use cat on linux, notepad or another text editor on windows/mac)
   - Copy ALL TEXT (CTRL-A) into “key”





- Go to https://github.com/settings/keys
- Click “New ssh key” and set some title
- Open “.ssh/id_rsa.pub” (use cat on linux, notepad or another text editor on windows/mac)
- Copy ALL TEXT (CTRL-A) into “key”

IDX EXCHANGE

Step 4: Complete!

Congrats you can know fork, clone, pull, and push to your repositories

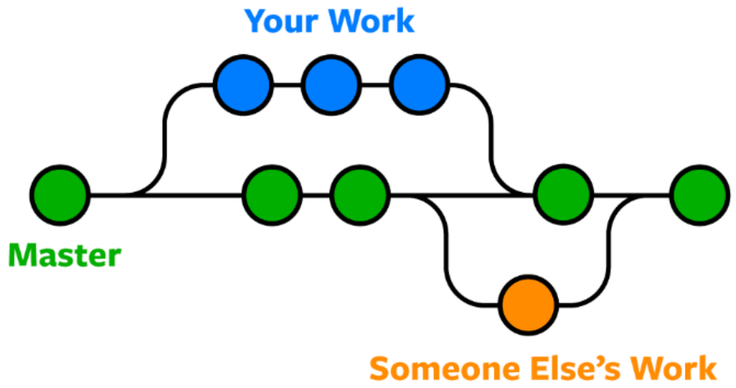
Basic git definitions:
docs.github.com/en/repositories/creating-and-managing-repositories/about-repositories
git reference sheet/documentation:
https://git-scm.com/docs

IDX EXCHANGE

Branching

What is a branch?

- “A parallel version of your code that is contained within the repository, but does not affect the primary or main branch.”
- Can see changes between branches
- “main” is the current (main) version of your repository –can merge branches to main



Making a branch in your fork (git bash):

git branch example_branch

git switch example_branch

Try making, adding, and committing a new file!

git push --set-upstream origin example_branch

Now you can PR the branch to your repo!