# Summary of Key Concepts

## The Classical Stack and Programming

### Week of October 1, 2023

---



---

## Resources

- Public Copy of QXQ YLC Week 2 Lab Notebook [STUDENT]
- Public Copy of QXQ YLC Week 2 Homework [STUDENT]
- Public Copy of 1. QxQ YLC 23-24 Python Basics Cheat Sheet

## Key Terms

| Key Term | Definition |
|---|---|
| **Stack** | The **stack** organizes all layers of a computer's operation, starting with the most fundamental and hidden from sight on the bottom. |
| **Bit** | Bits are the fundamental unit of information in a computer. Every app, website, image, text, program, and more is built from bits. |
| **Gates** | Gates manipulate bits. They allow us to turn inputs into outputs. |
| **Circuit** | A circuit is a sequence of gates. By combining many different gates in certain orders, we can start to process input into outputs in more complex and interesting ways. |
| **Algorithms/Protocol** | Algorithms and protocols are the agreed-upon steps used to solve problems. |
| **Application** | Applications are what you see and use. |
| **Programming** | The process of creating a set of instructions in a specific programming language to direct a computer. |
| **Program** | A collection of these instructions written in a programming language, designed to complete a specific task or solve a particular problem. |
| **IDE (Integrated Development Environment)** | Software where programmers can create, edit, and run programs. |

# Lecture

## Learning Objectives

1. *Recognize* the layers of the classical stack: bits, gates, circuits, algorithms/protocols, and applications.
2. *Recognize* that programming is the act of writing or modifying instructions for a computer.
3. *Recognize* that python is a programming language we will be using, including specialized quantum computing libraries.
4. *Recognize* how to read basic python code.

## Key Ideas

1. The **Stack** organizes all layers of a computer's operation, starting with the most fundamental and hidden from sight on the bottom.
2. The layers of the classical stack: **bits, gates, circuits, algorithms/protocols,** and **applications**.
    a. **Bits** are at the bottom level of the stack and are the fundamental unit of information in a computer. Each bit of information can be either a **0** or **1.**
        i. Instead of using 26 letters like the English alphabet or 10 digits like our numbering system, bits just use 2 digits: **0 and 1–also known as binary**.
        ii. Physically, bits can be represented by a switch. There are many ways to create switches, but small electrical switches called **transistors** are the standard in computers today.
    b. **Gates** manipulate bits to perform certain operations. There are many different types of gates, but here we briefly introduce the NOT gate and the AND gate as examples for operation performed on **bits.**
    c. A **circuit** is a sequence of gates to perform a more complex operation than the ones that single gates perform.
    d. **Algorithms & Protocols** are the agreed-upon steps computers used to solve

problems. These are how applications perform their tasks through search algorithms, streaming protocols, and more.

e. **Applications** lie at the top layer of the stack and are the things we interact with on our computers. They are what you see and use.

3. Programming is the skill set that enables us to instruct computers to carry out the intricate tasks involved in quantum computing.

    a. **Programming** is writing a set of instructions for a computer. A computer **program** is this set of instructions, crafted in a specific **programming language** to perform a task.

    b. **Python** is the programming language we'll use for its ease of use, versatility, and rich ecosystem of tools and libraries for quantum computing.

4. **Python libraries** serve as specialized tools to enhance coding capabilities in quantum computing using pre-written code.

5. Libraries are advantageous because they save time and are community-optimized.

    a. Popular libraries used in quantum computing include **numpy, cirq, and qiskit**.

# Lab

## Learning Objectives

1. *Recognize* the framework of objects, methods, attributes.
2. *Recognize* the main data types and how to modify them with basic arithmetic operators (+, -, /, *).
3. *Recognize* that variables are a way to store objects.

## Key Ideas

1. Python is an Object-Oriented Programming (OOP) language that encapsulates data and behavior into objects.
    a. Attributes represent the data we work with.
    b. Methods are the actions we can perform on this data. Methods in Python are specialized functions that operate on an object's attributes, often encapsulating behavior relevant to those attributes.

2. Python classifies objects into various types, also known as classes or data types, to manage the diversity of objects.
    a. An "int" or integer represents whole numbers, like 1, 10, or -5.
    b. A "float" is used for numbers with decimal points, such as 3.14 or 2.71.
    c. A "string" stores text and is enclosed in quotes, like "Hello" or "Python."
    d. A "list" is an ordered collection of items that can contain mixed types, like [1, 'a', 3.14].
    e. A "boolean" takes the value of either True or False, and is often used for conditional logic.

3. Variables in Python serve as containers to store objects, facilitating their later use.
    a. Variables are initially assigned specific values, which can be updated as needed.
    b. Variables must first be declared before being referenced.
    c. The stored objects can vary in type, including integers, floats, and strings.