1. What are the benefits of using Python?

- Python is known for its simplicity, readability, and ease of use.
- It has a large standard library and a vast ecosystem of third-party libraries, making it versatile and suitable for a wide range of applications.

Source: Python.org

2. What are the supported standard data types in Python?

- Python supports several standard data types, including:
  - Numeric types: int, float
  - Sequence types: list
  - Text type: str
  - Boolean type: bool

Source: Python Documentation - Built-in Types

3. What is pep 8?

- PEP 8 is the official style guide for Python code. It provides guidelines and recommendations on how to write Python code to enhance its readability and maintainability.

Source: PEP 8 - Style Guide for Python Code

4. What are the limitations of Python?

- Python can be slower compared to low-level languages like C or C++.
- Memory consumption can be relatively high in Python.

Source: Real Python - Limitations of Python

5. How do you distinguish between NumPy and SciPy?

- NumPy is a fundamental library for scientific computing in Python. It provides support for large, multi-dimensional arrays and matrices, along with a collection of mathematical functions to operate on them.
- SciPy, on the other hand, builds upon NumPy and provides additional functionality for scientific and technical computing, including optimization, interpolation, signal processing, linear algebra, and other options.

Source: NumPy - What is NumPy? and SciPy - Overview

6. What is Flask and what are the benefits of using it?

- Flask is a lightweight web application framework for Python. It is designed to be simple and flexible, allowing developers to create web applications quickly and easily.

Source: Flask - Home

7. How are modules imported in Python?

- Modules in Python can be imported using the import statement, followed by the name of the module. For example, to import the math module, you would use: import math

Source: Python Documentation - Modules

8. What rules govern local and global variables in Python

- In Python, variables declared inside a function are considered local to that function and can only be accessed within the function's scope.
- Global variables, on the other hand, are declared outside of functions and can be accessed from anywhere in the code.

Source: Real Python - Python Scope & the LEGB Rule: Resolving Names in Your Code

9. What is a Package in Python?

- Packages are simply directories that contain one or more Python modules, along with a special file called __init__.py that is used to initialize the package when it is imported.

Source: Python Documentation - Packages

10. What is lambda in Python?

- Lambda functions are often used as a shortcut for creating simple functions that are only needed once in a program.

Source: Python Documentation - Lambda Expressions

11. What is a PIP?

- PIP ("Pip Installs Packages") is the default package installer for Python. It is used to install, upgrade, and manage Python packages from the Python Package Index (PyPI).

Source: Python Packaging User Guide - Installing Packages

12. What are Pandas in Python?

- Pandas is a popular data manipulation library for Python. It provides high-performance, easy-to-use data structures and data analysis tools for handling and analyzing large datasets.

Source: Pandas Documentation - About Pandas

13. What library is required in order to run games using Python?

- One library that we used in classfor creating games in Python is Pygame.
- Pygame is a cross-platform set of Python modules designed for game development. It provides functionality for handling graphics, sound, and user input, making it suitable for building games.

Source: Pygame - Introduction

14. What does the ‘#’ symbol do in Python?

- In Python, the '#' symbol is used to indicate a single-line comment.

Source: Python Documentation - Comments

15. What is a pass in Python?

- pass is a placeholder statement in Python that does nothing.

Source: Python Documentation - The pass Statement

16. What is the purpose of the __init__ method in Python classes?

- The __init__ method is used to initialize the attributes of an object by assigning values to them.

Source: Python Documentation - 9.3.3. Special method names - __init__

17. Is Indentation Required in Python?

- Yes, indentation is required in Python and serves as a part of the language's syntax.
- Indentation is used to define the structure and hierarchy of blocks of code, such as function and class definitions, loops, and conditional statements.

Source: Python Documentation - 2.1.3. Indentation

18. Is Python good for building AI environments?

- Yes, Python is widely used for building AI environments and is one of the most popular programming languages for AI and machine learning.
- Python provides a rich ecosystem of libraries and frameworks, such as TensorFlow, PyTorch, and scikit-learn, which offer powerful tools for AI development.

Source: Towards Data Science - Why Python is the Best Programming Language for AI

19. How do you convert the data from a .csv file using Python?

- To convert data from a CSV file in Python, you can use the csv module, which provides functionality for reading and writing CSV files.

```
import csv
        with open('data.csv', 'r') as file:
                reader = csv.reader(file)
                data = list(reader)
```

- This code opens the data.csv file, reads its contents using the csv.reader function, and converts it into a list of rows.

Source: Python Documentation - csv - CSV File Reading and Writing

20. Which library allows the user to utilize the Actor class?

- The multiprocessing module in Python allows users to utilize the Actor class.

Source: Python Documentation - multiprocessing - Process-based Parallelism

21. Which library must be imported for your python IDE to run a game?

- The specific library required to run a game in a Python IDE depends on the game's development framework or engine.
- In class, we are using Pygame to develop a game, we would need to import the Pygame library in our Python IDE to run the game.

22. Describe the process of creating and using virtual environments in Python development.

- Virtual environments in Python are used to create isolated environments with their own Python installations and package dependencies.
- To create a virtual environment, you can use the venv module, which is included in Python's standard library.
- Here are the basic steps to create and use a virtual environment:
    1. Open a terminal or command prompt.
    2. Navigate to the desired directory for your project.
    3. Create a virtual environment using the command python3 -m venv myenv (replace myenv with the desired name).
    4. Activate the virtual environment:
        - On Windows: myenv\Scripts\activate.bat
        - On macOS and Linux: source myenv/bin/activate
    5. Install packages and dependencies within the virtual environment using pip.
    6. Run your Python code within the activated virtual environment.
    7. To deactivate the virtual environment, use the command deactivate.

Source: Python Documentation - Virtual Environments and Packages

23. Can you provide an example of a challenging problem you solved using Python, and explain your thought process and approach?

- One challenging problem that

24. What applications have you built using Python

25. What are the advantages of Python over C++?

- Python and C++ are different programming languages with their own strengths and weaknesses. However, Python offers several advantages over C++:
    - Python is known for its simplicity, readability, and ease of use, making it more beginner-friendly compared to C++.
    - Python has a rich set of libraries and frameworks for various tasks, such as data analysis, web development, and machine learning, which can speed up development.

Source: Towards Data Science - Python vs C++: 8 Key Differences You Should Know

26. What is *args and **kwargs?
    - *args is used to pass a variable number of non-keyword arguments to a function. It collects the arguments into a tuple.
    - **kwargs is used to pass a variable number of keyword arguments to a function. It collects the arguments into a dictionary.

Source: Real Python - *args and **kwargs in Python: What They Are and How to Use Them

27. What is slicing in Python?
    - Slicing in Python is a way to extract a portion (subsequence) of a sequence, such as a string, list, or tuple.

Source: Python Documentation - Common Sequence Operations

28. What are generators in Python
    - Generators in Python are a type of iterable that can be iterated over only once.
    - They are defined using a special kind of function called a generator function, which uses the yield statement to return values one at a time, preserving the function's state between each yield.

Source: Real Python - Python Generators: A Beginner's Guide

29. What are decorators in Python?
    - Decorators in Python are a way to modify the behavior of functions or classes without directly modifying their source code.
    - They allow additional functionality to be added to existing functions or classes by wrapping them inside another function.
    - Source: Real Python - Primer on Python Decorators

30. What is the difference between deep and shallow copy in Python?
    - In Python, a shallow copy and a deep copy are two methods for creating copies of objects.
    - A shallow copy creates a new object that references the same memory as the original object. Changes made to the original or the shallow copy will affect both.
    - A deep copy creates a new object and recursively copies the values of nested objects as well. Changes made to the original or the deep copy will not affect each other.

Source: Python Documentation - copy - Shallow and Deep Copy Operations