Q1. Is it permissible to use several import statements to import the same module? What would the goal be? Can you think of a situation where it would be beneficial?

Answer: Yes we can use several import statements to import different sub modules from one particular library.

Q2. What are some of a module's characteristics? (Name at least one.)

Answer: A module always comes first before . operator when we are trying to import a function from that module.

Q3. Circular importing, such as when two modules import each other, can lead to dependencies and bugs that aren't visible. How can you go about creating a program that avoids mutual importing?

Answer: To avoid circular importing, we can organize the code to reduce errors and move all references belonging to a particular module inside one code block.

Q4. Why is \_ \_all\_ \_ in Python?

Answer: \_\_all\_\_ is a list of all public objects belonging to a particular module that we want to import in another module with help of import statement.

Q5. In what situation is it useful to refer to the \_ \_name\_ \_ attribute or the string '\_ \_main\_ \_'?

Answer: \_\_name\_\_ is used to set the name of the current script that we are working on . \_\_main\_\_ is used when we want to import a particular module in an another script.

Q6. What are some of the benefits of attaching a program counter to the RPN interpreter application, which interprets an RPN script line by line?

Answer: Program counter to RPN interpreter will help in knowing the number of instructions passed if we set the program counter for reading every line one by one.