**ASSIGNMENT 24**

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?

**Ans:** Yes, it is possible to use several import statements for importing the same module. This can be used to import different parts of the module in the file. For example, if we want to import different elements or submodules from the same module. For instance, it can be beneficial in importing any big module like pandas or NumPy which contain many elements and we want to organize our code accordingly.

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

**Ans:** Modules are very useful for implementing same code in various parts of the python project. It can be imported as an object and can be reused decreasing the complexity of the code.

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?

**Ans:** To avoid mutual import bugs we can try to create a separate module, if two modules depend on each other. Another way is to limit the import of the module to just the required element.

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

**Ans:** \_\_all\_\_ is a list type object used to store names of the functions which will be publicly available after using the ‘from module import \*’ statement. This is an optional variable and used to protect the private functions in the module, so that they cannot be imported externally.

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

**Ans:** \_\_name\_\_ attribute and ‘\_\_main\_\_’ string is almost used together in the code. For instance, in the line of code ‘if \_\_name\_\_=”\_\_main\_\_”:’ whatever is written after this will be executed if the module is running as a main program. That is it is not imported by any other program.

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?

**Ans:** There are many benefits of attaching a program counter to the RPN interpreter application, one of them is it is a useful tool for debugging, error reporting and performance optimization. Second is program counter enables the step-by-step execution of the RPN script making it easier to understand.

Q7. What are the minimum expressions or statements (or both) that you’d need to render a basic programming language like RPN primitive but complete— that is, capable of carrying out any computerized task theoretically possible?

**Ans:** We need at least Arithmetic operations expressions and statements, stack manipulation, Comparison operations, Conditional statements, Loops, Variables and Input/Output statements.