Q1. What is the purpose of the try statement?

Ans the block in which we write block of statements generating exceptions. In otherwords what are all the statements generating exceptions, those statements must be written within try block and hence try block is called Exception monitering block.

=>When an exception occurs in try block then PVM comes out of try block and executes

appropriate except block.

=>After executing appropriate except block, PVM never goes to try block for executing rest of

the statements in try block.

=>Every try block must be immediately followed by except block ( Otherwise we get

SyntaxError)

=>Every try block must contain atleast one except block . It is recommended to write multiple

except blocks for generating User-Friendly error messages.

Q2. What are the two most popular try statement variations?

Ans =>When an exception occurs in try block then PVM comes out of try block and executes

appropriate except block.

=>After executing appropriate except block, PVM never goes to try block for executing rest of

the statements in try block.

=>Every try block must be immediately followed by except block ( Otherwise we get

SyntaxError)

Q3. What is the purpose of the raise statement?

Ans =>raise keyword is used for hitting / raising / generating the exception provided some

condition must be satisfied.

=>raise keyword always used inside of Function Definition only.

=>PVM uses raise keyword implicitly for hitting pre-defined Exceptions where as Programmer makes the PVM to use raise keyword explicitly for Hitting or Generating Programmer-defined Exceptions.

Q4. What does the assert statement do, and what other statement is it like?

Ans assert statement is used to check types, values of argument and the output of the function

Q5. What is the purpose of the with/as argument, and what other statement is it like?

Ans In Python, the with statement replaces a try-catch block with a concise shorthand. More importantly, it ensures closing resources right after processing them.