Q1. What is the purpose of the try statement?

Ans: **try** statement is used for exception handling in python. it consists of a block of risky statements which might cause an exception during runtime. if code within try block raises exception, then the exception will be reverted to the corresponding except block if multiple except blocks are present, else it willbe reverted to the default except block. In short try and except in union avoid programs from crashing during runtime due to exceptions.

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

Ans: The Popular try statement variations are:

1. **try,except**
2. **try,except,finally**
3. **try,except,finally,else**

The **try** block is used to check code for exceptions during runtime. ie code inside try block will execute completely when there is no error in the program. Whereas the code inside except block will execute whenever the program encounters some error in the preceeding **try** block.

Whereas the code enters the **else** block if and only if the try clause does not raise and exception. The code in the **finally** block will execute irrespective of exception.

Q3. What is the purpose of the raise statement?

Ans: **raise** statement is used to trigger an exception explicitly, if a certain condition is not as per requirement of programmer. **raise** statement helps in triggering exception as per programming logic.

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

Ans: There are few assertions that programmer always want to be true to avoid code failure. This type of requirement is fulfilled by **assert** statement. This statement takes a boolean condition output of which is True, Further Program Executes. if output of assert statement is False, it raises an Assertion Error.

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

Ans: **with/as** statement simplifies use of file handling in python. When we use a **with** statement for file reading, there is no need for programmer to explicitly takecare of activities like resource deallocation and file closing by using file.close() method. **with** statement itself ensures proper acquisition and release of resources. this avoids triggering of exceptions if file closing is unknowingly forgotten in the code execution.

```
with open('sample_file.txt','w') as file: file.write('Hello World')
```