**Q1.** In Python, Keywords are considered as reserved keywords, which have special meanings. It cannot be used as a variables/ identifiers/ function name. The list of Keyword can be found through the ‘keyword’ module.

Code to print all the keyword, though keyword module in python is as follows:

**import keyword**

**keywords = keyword.kwlist**

**for keyword in keywords:**

**print(keyword)**

**Q2.** Rules to create variables in Python:

1. In Python, the variable names must be valid identifiers. Starting with a letter (a-z, A-Z) or an underscore (\_) and can be followed by letters, digits (0-9), or underscores.
2. It is Case-sensitive, so ‘playGround’, ‘PlayGround’, ‘playground’ are all different variables.
3. Cannot use reserve keywords as variable names. Keywords are used for special purposes like ‘if, ‘else’, ‘for’, ‘while’, ‘def’, ‘class’ etc.,
4. No special characters have been used in the variable names. Only letters, digits and underscores are allowed.
5. Cannot start with a digit (0-9).

**Q3.**

1. In Python, the most widely followed conventions are outlined with PEP 8 style guide, considered as de facto standard for Python code style.
2. Variables names should be descriptive, purposeful, meaningful and easier to understand.
3. The variable names should be in lowercase, with words separated by underscores (example, ‘baby\_sitter’).
4. Single letter variable names (like ‘x’, ‘y’, ‘i’, etc.,) should generally be avoided unless they’re used as loop counters or in other contexts where their meaning is clear from the surrounding code.
5. Choose variable names that clearly convey their purpose or meaning within the context of the code. Avoid overly generic names like ‘temp’, ‘data’, or ‘value’.
6. For constansts, Use upper letters with underscore to seperate words
7. Variables intended for internal use within a class should start with a single underscore(‘\_’).
8. Variables intended for limited use within a class or subclass should start with a double underscore (‘\_ \_’).

**Q4.** If a keyword is used as a variable name in Python, it will result in a syntax error. Because keywords are reserved by the Python language for specific purposes and cannot be used as identifiers (Such as, variable names, function names, or class names).

**Q5.** The ‘def’ keyword in Python is used to define a function. Function are blocks of reusable code that perform a specific task. When you define a function using the ‘def’ keyword, you specify the name of the function, the parameter it takes and the block of code that constitutes the function’s body.

**Q6.** In Python, the backslash (‘\’) character is used as an escape character. It represents special characters, control formatting, and deals with various text processing tasks.

**Q7.** Examples as follows:

1. Homogeneous\_list\_int = [1, 2, 3, 4, 5]

Homogeneous\_list\_str = [“car”, “bike”, “truck”]

1. Heterogeneous\_set = {1, “apple”, 3.14}
2. Homogeneous\_tuple\_int = (1, 2, 3, 4, 5)

Homogeneous\_tuple\_str = (“apple”, “banana”, “orange”, “grape”)

**Q8.**

**Mutable Data Type:**

Values can be changed after they are created, leading to modifying the contents of a mutable object without creating a new object.

**Example: lists, dictionaries and sets**

**Example with a mutable data type:**

**Input:**

My\_list = [1, 2, 3]

print(“Original list:”, my\_list)

My\_list.append(3)

print(“Modified list:”, My\_list)

**Output:** Modified\_list: [1, 2, 3, 3]

**Immutable Data Type:**

Values cannot be changed after they are created. While performing operation that appear to modify an immutable object, leads to creating a new object with modified value, rather than modifying the original object.

**Example: integers, floats, strings, tuples and frozensets**

**Example of an immutable data type:**

**Input:**

my\_tuple = (1, 2, 3)

print(“Original tuple:”, my\_tuple)

new\_tuple = my\_tuple + (4,)

print(“New tuple:”, new\_tuple)

**Output:** New tuple: (1, 2, 3, 4)

**Q9.**

rows = 5

for i in range(1, rows + 1):

for j in range(rows - i):

print(" ", end="")

for k in range(2 \* i -1):

print("\*", end="")

print()

**Q10.**

rows = 5

for i in range(row, 0, -1):

for j in range(rows - i):

print(" ", end="")

for k in range(2 \* i -1):

print("|", end="")

print()