1. What are keywords in python? Using the keyword library, print all the python keywords.

In [1]:

```
import keyword

all_keywords = keyword.kwlist
print(all_keywords)
```

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'brea
k', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finall
y', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonloc
al', 'not', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yiel
d']
```

2. What are the rules to create variables in python?

Ans:-
    (1) Variable Names:
Variable names must start with a letter (a-z, A-Z) or an underscore (_).
The first character cannot be a number.
Variable names can contain letters, digits (0-9), and underscores.
Python is case-sensitive, so myVariable and myvariable are considered different variables.

    (2)Valid Examples:
Valid variable names: my_variable, count, *_data, name1, total_sum_2, etc.*
Invalid variable names: 2nd_variable (starts with a digit), my-variable (contains a hyphen), @symbol (contains a special character).

    (3)Reserved Keywords:
Python has some reserved keywords that cannot be used as variable names since they have predefined meanings in the language. For example, if, else, while, for, def, class, etc.
For Example:-

In [3]:

```
name = "John"
age = 25
is_student = True
total_score = 98.5
```

3. What are the standards and conventions followed for the nomenclature of variables in python to improve code readability and maintainability?

Ans:-

(1) Variable Names:
Use descriptive names that indicate the purpose or content of the variable. Variable names should be lowercase, and words should be separated by underscores (snake_case). For example: total_count, user_name, data_list.

(2) Constants:
Constants are typically written in uppercase with underscores. For example: MAX_ITERATIONS, PI, DEFAULT_VALUE.

(3) Private Variables:
To indicate that a variable is intended for internal use only, prefix its name with a single underscore. For example: *_internal_data*.

(4) Protected Variables:
For variables that are intended to be used only within the class and its subclasses, prefix the name with a double underscore. For example: **__protected_data.**

4. What will happen if a keyword is used as a variable name?

Ans:-
If a keyword is used as a variable name in a programming language, it will typically result in a syntax error. Keywords are reserved words in programming languages, and they have specific meanings and purposes defined by the language itself. These words cannot be used as identifiers (such as variable names, function names, or class names) because doing so would lead to ambiguity and potential conflicts with the language's syntax and functionality.
For Example:-

In [4]:

```
if = 10
print(if)
```

```
  Cell In[4], line 1
    if = 10
       ^
SyntaxError: invalid syntax
```

5. For what purpose def keyword is used?

Ans:-
The def keyword is used in Python to define a function. Functions are blocks of code that perform a specific task or set of tasks and can be reused throughout the program. By using the def keyword, you can create user-defined functions that can be called multiple times from different parts of the code.
For Example:-

In [5]:

```
def greet(concept):
    return f"Hello, {concept}!"

result = greet("World")
print(result)
```

```
Hello, World!
```

6. What is the operation of this special character ‘\’?

```
Ans:-
    The special character '\' is known as the backslash. In programming and computer
systems, the backslash serves several purposes depending on the context in which it is
used.The backslash is commonly used as an escape character in many programming
languages and regular expressions. It allows you to use characters that would
otherwise be interpreted differently or have special meanings. For example:

'\n' represents a newline character.
'\t' represents a tab character.
'\' represents a literal backslash character itself.
```

7. Give an example of the following conditions:

(i) Homogeneous list

(ii) Heterogeneous set

(iii) Homogeneous tuple

```
Ans:-
    (1) Homogeneous list:
A homogeneous list is a list where all elements have the same data type. In Python, it
could be a list of integers, a list of strings, or a list of booleans.
For Example:-
```

In [6]:

```
homogeneous_list = [1, 2, 3, 4, 5]
print(homogeneous_list)
```

```
[1, 2, 3, 4, 5]
```

```
   (2) Heterogeneous set:
A heterogeneous set is a set that contains elements of different data types. In
Python, a set can have elements of various types, such as integers, strings, and
floats.
```

For Example:-

In [7]:

```
heterogeneous_set = {1, 'hello', 3.14, True}
print(heterogeneous_set)
```

```
{1, 3.14, 'hello'}
```

(3)Homogeneous tuple:
A homogeneous tuple is a tuple where all elements have the same data type. In Python, like lists, tuples can also be homogeneous, containing elements of the same data type.
For Example:-

In [8]:

```
homogeneous_tuple = (10, 20, 30, 40, 50)
print(homogeneous_tuple)
```

```
(10, 20, 30, 40, 50)
```

8. Explain the mutable and immutable data types with proper explanation & examples.

Ans:-
(1) Mutable Data Types:
Mutable data types are those whose values can be modified or changed after they are created. This means that you can alter the content of a mutable object without creating a new object in memory. This can be advantageous in certain situations where you need to perform frequent updates or modifications.
Examples of mutable data types in Python include:

Lists
Dictionaries
Sets

In [9]:

```
# Create a list
mutable_list = [1, 2, 3, 4]

# Modify the list by changing one of its elements
mutable_list[2] = 10

print(mutable_list)
```

```
[1, 2, 10, 4]
```

(2) Immutable Data Types:
Immutable data types are those whose values cannot be changed after they are created. Once an immutable object is created, any attempt to modify it results in the creation of a new object with the updated value. This means that the original object's value remains unchanged, and a new object is stored in a different location in memory.
Examples of immutable data types in Python include:

```
Integers
Floats
Strings
Tuples
```

In [10]:

```
# Create a tuple
immutable_tuple = (1, 2, 3)

# Attempt to modify the tuple will raise an error
# immutable_tuple[1] = 10  # This will result in a TypeError

# To modify the tuple, you need to create a new one
modified_tuple = immutable_tuple + (10,)
print(modified_tuple)
```

```
(1, 2, 3, 10)
```

Q.9. Write a code to create the given structure using only for loop.

*****

****

****

****

****

In [11]:

```
# Number of rows in the structure
num_rows = 5

# Outer loop for rows
for i in range(1, num_rows + 1):

    # Inner loop for printing asterisks in each row
    for j in range(1, 2 * i):
        print("*", end="")

    # Move to the next line after printing each row
    print()
```

```
*
***
*****
*******
*********
```

10. Write a code to create the given structure using while loop.

|||||||||

|||||||

|||||

|||

|

In [12]:

```
def create_structure(rows):
    while rows > 0:
        print('|' * rows)
        rows -= 1

# Set the number of rows for the structure
num_rows = 5

create_structure(num_rows)
```

```
|||||
||||
|||
||
|
```

In [ ]:

```

```