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

Keywords in Python are reserved words that have predefined meanings and cannot be used as variable names or identifiers.

In [1]:
```python
# List of keywords is given as follows:
import keyword

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

```
['False', 'None', 'True', 'and', 'as', 'assert', 'async', 'await', 'break',
'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'finally', 'fo
r', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'nonlocal', 'no
t', 'or', 'pass', 'raise', 'return', 'try', 'while', 'with', 'yield']
```

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

The rules for creating variables in Python:

1. Variable names must start with a letter (a-z, A-Z) or an underscore (_). They cannot start with a number.
2. They are case-sensitive, so "myVariable" and "myvariable" are considered different variables.
3. Python keywords (reserved words) cannot be used as variable names.
4. Variable names should not contain spaces. Use underscores to separate multiple words in a variable name.
5. Avoid using special characters like !, @, #, $, %, etc. in variable names.

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

Use grammatically correct variable names, the class name should start with an uppercase and must follow camelCase convention If more than two words are to be used. In the same way, a function name should be joined with an underscore, and it must be lowercase.

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

In [5]:
```
# If we use keyword as variable name it will throw a syntax error.
try = 5
try
```

```
  Cell In[5], line 2
    try = 5
        ^
SyntaxError: expected ':'
```

Q.5. For what purpose def keyword is used?

The "def" keyword is used for defining functions in Python, allowing you to modularize your code, promote reusability, and improve code organization.

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

The special character \ is known as the backslash. It is used as an escape the character. When you want to include special characters within a string, such as quotes or newlines, you can use the backslash to escape them. It is also used as indication of continuation of lines.

In [12]:
```
#Eg:
print("Hi my name is \"Hritik\"")
print("Line 1\nLine 2")

```

```
Hi my name is "Hritik"
Line 1
Line 2
```

Q.7. Give an example of the following conditions: (i) Homogeneous list (ii) Heterogeneous set (iii) Homogeneous tuple

Homogeneous List: A homogeneous list is a list where all elements are of the same data type. For example, a list containing only integers or only strings. Eg: Hom_list=[1,2,3,4]

Hetergeneous Set: A sets in python are considered as heterogeneous because they can contain elements of different data types. Eg: Het_set = {1, "two", 3.14, True}

Homogeneous Tuple: A homogeneous Tuple is a tuple where all elements are of the same data type. For example, a tuple containing only integers or only strings. Eg: Hom_list=('hi','bye','hello')

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

Mutable date types: Mutable data types are those whose values can be modified after creation. This means you can change individual elements of a mutable object without creating a new object. Examples of mutable data types in Python include lists, dictionaries, and sets. Eg: a=[1,2,3,4] a[2]=6 #output will be a=[1,2,6,4]

Immutable date types: Immutable data types are those whose values cannot be changed after creation. If you want to modify an immutable object, you need to create a new object with the desired changes. Examples of immutable data types in Python include integers, floats, strings, and tuples. Eg: a=(1,2,3,4) a[2]=6 # it will throw TypeError.

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

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

In [24]:
```python
n=int(input("Enter number of rows: "))

for i in range(1, n+1):
    spaces = " " *(n-i)
    stars = "*" *(2*i-1)
    print(spaces + stars)

```

```
Enter number of rows: 5
    *
   ***
  *****
 *******
*********
```

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

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

In [3]:
```
n=0
while n<=5:
    print(" "*(n)+"|"*(2*(5-n)-1))
    n+=1
```

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