```python
# 1. Significance of Python Keywords and Examples

# Python keywords are reserved words that have predefined meanings in Python's syntax.
# These keywords cannot be used as identifiers (variable names, function names, etc.).

# Example of Python keywords:
# - if
# - for
# - def
# - return
# - else

# Example usage of keywords:
x = 15
y = 3

# if keyword: Conditional statement
if x > y:
    print(f"{x} is greater than {y}")

# for keyword: Looping through a range
for i in range(5):
    print(i)

# def keyword: Defining a function
def add(a, b):
    return a + b

# return keyword: Returning a value from a function
result = add(10, 5)
print(f"Sum of 10 and 5 is {result}")

# else keyword: Alternative block in a conditional
if x < y:
    print(f"{x} is less than {y}")
else:
    print(f"{x} is not less than {y}")

# 2. Rules for Defining Identifiers in Python

# Identifiers in Python are names used to identify variables, functions, classes, etc.
# Rules for identifiers:
# - An identifier must start with a letter (a-z, A-Z) or an underscore (_).
# - The rest of the identifier can include letters, digits (0-9), and underscores.
# - Identifiers are case-sensitive.
# - Cannot use Python keywords as identifiers.
# - Identifiers cannot start with a number.

# Valid identifiers:
variable_name = 10
_variable2 = 20
x_value = 30

# Invalid identifier:
# 2variable = 10  # This would result in a syntax error because it starts with a number

# 3. Comments in Python and Why Are They Useful?

# Comments are used to explain code, making it more understandable to humans.
# They are ignored by the Python interpreter.

# Single-line comment
# This is a single-line comment in Python.

# Multi-line comment (using triple quotes)
"""
This is a multi-line comment.
It spans multiple lines and is often used for docstrings.
"""

# Example of comments:
# This function adds two numbers
def add_numbers(a, b):
    return a + b  # Returning the sum of a and b

# 4. Why is Proper Indentation Important in Python?

# In Python, indentation is used to define the blocks of code.
# Code blocks are defined by the level of indentation, not by curly braces as in other languages.

# Correct indentation:
x = 10
if x > 5:
    print("x is greater than 5")  # This line is indented and part of the if block

# 5. What Happens if Indentation is Incorrect in Python?

# Incorrect indentation will lead to an IndentationError or logic errors.
# Example of incorrect indentation:

# This will cause an IndentationError:
# if x > 5:
# print("x is greater than 5")  # This is not indented correctly

# Corrected version:
if x > 5:
    print("x is greater than 5")

# 6. Expression vs Statement in Python

# An expression is a combination of values, variables, operators, and function calls that can be evaluated.
# An expression always returns a value.

# Example of an expression:
result = 5 + 3  # 5 + 3 is an expression, and the result is assigned to 'result'

# A statement is a complete instruction that performs an action, but does not return a value.

# Example of a statement:
if result > 7:  # 'if result > 7' is a statement that evaluates the condition
    print("The result is greater than 7")  # The print function is a statement

# Another example of an expression:
expression_result = 5 * 4  # This is an expression, which evaluates to 20

# Another example of a statement:
print("This is a statement")  # The print() function call is a statement

# Summary of Expression vs Statement:
# An expression produces a value, while a statement performs an action.

# Expression Example:
x = 5 * 2  # Expression that produces the value 10

# Statement Example:
if x > 5:  # This is a statement that performs the action of checking the condition
    print("x is greater than 5")
```