Assignment_3

March 20, 2024

[1]:
```
#Python keywords are reserved words that have special meaning and significance␣
 ↪in the Python language.
#They are words that cannot be used as identifiers,
#such as variable names, function names, or argument names.
#Instead, they have a fixed meaning and syntax,
#they are used to perform specific actions or functions in Python.

#Example:

#print: The print keyword is used to output information to the file.
#It is used to display the value of expression or any input.


x = 5
y = 10
print(x + y)

#if, elif, and else: The if, elif, and else keywords are used to perform␣
 ↪conditional actions in Python.
#They are used to test whether a condition is true or false.

x = 5
if x > 0:
    y = x + 1
else:
    y = x - 1
print(y)


#for: The for keyword is used to perform iterative actions in Python.
#It is used to repeat a sequence of actions for each element in a collection,
#such as a list.

my_list = [1, 2, 3, 4, 5]
for x in my_list:
    print(x * 2)
```

```
#while: The while keyword is used to perform looping actions in Python.
#It is similar to for, but it is used when the number of iterations is not␣
 ↪known in advance.


x = 0
while x < 5:
    print(x)
    x = x + 1


#try and except: The try and except keywords are used to perform exception␣
 ↪handling in Python.
#They are used to handle exceptions, or unexpected errors that occur during the␣
 ↪execution of a program.

try:
    x = 1 / 0
except ZeroDivisionError:
    print("Cannot divide by zero!")
```

```
15
6
2
4
6
8
10
0
1
2
3
4
Cannot divide by zero!
```

[1]:
```
#DescriOe the rules for defining identifiers in Python and provide an example


#An identifier must start with a letter or an underscore character _.
#An identifier cannot start with a digit,
#even though digits are allowed (in theory) allowed anywhere else in the␣
 ↪identifier.
#An identifier can consist of letters, digits, and underscores.
#it cannot contain spaces or other special characters.
```

```
#An identifier is not allowed to contain the keywords False, None, or True as a␣
 ↪substring.
#An identifier cannot contain the characters ~, !, @,  #, $, %, ^, &, *, -, +,␣
 ↪=, <, >, ?, :,', or .

#example:

lo = 42
This_is_my_variable = True
_my_identifier = 123
number1 = 12
```

[ ]:
```
#What are comments in Python, and why are they useful? Provide an example.

#in Python, an indentation is used importOant for defining blocks of code that␣
 ↪belong to a statement or a compound statement. Python uses indentation to␣
 ↪determine the scope and structure of statements,
#and it is a critical element of Python's syntax.

# This makes the code easier to read and understand,
#as it visually separates different blocks of code and helps to clarify the␣
 ↪structure of a program.

#example:


if x > 0:
y = x + 1
else:
y = x - 1

if y > 0:
z = y + 1
else:
z = y - 1

print(z)
```

[ ]:
```
#Proper indentation is important in Python because it is used to define blocks␣
 ↪of code that belong to a statement or a compound statement. Python uses␣
 ↪indentation to determine the scope and structure of statements,
and it is a critical element of Python's syntax.


#Proper indentation also makes your code more readable and easier to understand.
 ↪
```

```
#By using consistent indentation, you can clearly show the structure of your␣
 ↪code and make it easy for other developers to understand.
#This is important when working on a team or when maintaining a large codebase.
```

[ ]:
```
#What happens if indentation is incorrect in Python?

#if indentation is incorrect in Python, it may encounter a SyntaxError,
which will prevent the program from running.
This is because Python uses indentation to determine the scope and structure of␣
 ↪statements,
and it is a critical element of Python's syntax.If you do not use consistent␣
 ↪indentation,
Python may not be able to correctly determine the scope of a variable or a␣
 ↪function.
This can lead to unexpected behavior and errors in the code.
```

[ ]:
```
#Differentiate between expression and statement in Python with examples.

An expression is a methods that return a value.

while a statement is used to obtain a desired output.

Example:

  #if , while and for statements implement traditional control flow constructs.
#while the with statement allows the execution of initialization and␣
 ↪finalization code around a block of code.
```