INPUT AND OUTPUT (I/0) OPERATIONS IN PYTHON

WRITING OUTPUT

STRING CONCATENATION

FIELDS FOR ALF'S INTRODUCTION

```
name = "Alf"
organization = "AWS Cloud Club"
university = "PUP Manila"
```

METHOD 1: Concatenating Strings Inside print() Function:

- Description: In Python 3, the print() function can concatenate strings separated by commas. The values are automatically separated by a space.
- Example:

```
print("Hi, I am", name, "ረ ~^ ˇ ^~ ɔɔ and I am the official mascot of", organization,
"of", university, "and ready to guide you from the clouds and beyond!")
```

METHOD 2: Using the + Operator:

- Description: The + operator is used to concatenate string. Note that you need to convert the non-string variable to a string using str() before concatenating it with other strings.
- Example:

```
alf_introduction = "Hi, I am " + name + " ረ ~^ ˇ ^~ ɔɔ and I am the official mascot of " +
organization + " of " + university + " and ready to guide you from the clouds and beyond!"
print(alf_introduction)

print("Hi, I am " + name + " ረ ~^ ˇ ^~ ɔɔ and I am the official mascot of " + organization
+ " of " + university + " and ready to guide you from the clouds and beyond!")
```

METHOD 3: Using the format() method:

- Description: The format() method allows you to insert values into a string by using placeholders { }. The values are provided in the format() function in the same order as the placeholders in the string.

INPUT AND OUTPUT (I/0) OPERATIONS IN PYTHON

- Example:

```
alf_introduction = "Hi, I am {} ʕ ·ᴥ· ʔ and I am the official mascot of {} of {} and ready to guide you from the clouds and beyond!".format(name, organization, university)
print(alf_introduction)

print("Hi, I am {} ʕ ·ᴥ· ʔ and I am the official mascot of {} of {} and ready to guide you from the clouds and beyond!".format(name, organization, university))
```

METHOD 4: Using F-strings (PYTHON 3.6 and above)

- Description: F-strings (formatted string literals) were introduced in Python 3.6. They allow you to embed expressions inside string literals, using curly braces { }. The expressions inside the curly braces are evaluated and formatted into the string.
- Example:

```
alf_introduction = f"Hi, I am {name} ʕ ·ᴥ· ʔ and I am the official mascot of {organization} of {university} and ready to guide you from the clouds and beyond!"
print(alf_introduction)

print(f"Hi, I am {name} ʕ ·ᴥ· ʔ and I am the official mascot of {organization} of {university} and ready to guide you from the clouds and beyond!")
```

METHOD 5: Using the join() Method

- Description: The join() method is used to concatenate a list of strings into a single string. An empty string(“”) is used as the separator between list elements.
- Example:

```
alf_introductionList = ["Hi, I am ", name, " ʕ ·ᴥ· ʔ and I am the official mascot of ", organization, " of ", university, " and ready to guide you from the clouds and beyond!"]

alf_introduction = "".join(alf_introductionList)
print(alf_introduction)

print("".join(["Hi, I am ", name, " ʕ ·ᴥ· ʔ and I am the official mascot of ", organization, " of ", university, " and ready to guide you from the clouds and beyond!"]))
```

METHOD 6: Using the % Formatting

- Description: The % formatting allows you to insert values into a string using placeholders. The %s is a placeholder for a string, and %d is a placeholder for an integer. The values to be inserted are provided in a tuple following the % operator.
- Example:

INPUT AND OUTPUT (I/0) OPERATIONS IN PYTHON

```python
alf_introduction = "Hi, I am %s ؏ ~^ ˘ ^~ ؏ and I am the official mascot of %s of %s and ready to guide you from the clouds and beyond!" % (name, organization, university)
print(alf_introduction)

print("Hi, I am %s ؏ ~^ ˘ ^~ ؏ and I am the official mascot of %s of %s and ready to guide you from the clouds and beyond!" % (name, organization, university))

```