Python Programming - Beginner Level Notes

Prepared for: Bikkad IT Institute Students
Topic: Strings & Break-Continue-Pass
Date: February 2026

---

1. Break, Continue, Pass

What are Break, Continue, Pass?

Control flow statements that alter loop execution.

Statement	Description
break	Exit the loop immediately
continue	Skip current iteration, continue with next
pass	Do nothing, placeholder statement

Table 1: Control Flow Statements

Code Examples

Example 1: Break Statement

```
for i in range(1, 11):
if i == 6:
break # Exit loop when i is 6
print(i)
```

Output: 1 2 3 4 5

Example 2: Continue Statement

```
for i in range(1, 11):
if i % 2 == 0:
continue # Skip even numbers
print(i)
```

Output: 1 3 5 7 9

Example 3: Pass Statement

```
for i in range(5):
if i == 2:
pass # Do nothing, just placeholder
print(i)
```

Output: 0 1 2 3 4

Example 4: Practical Use

Search in list

```
numbers = [10, 20, 30, 40, 50]
search = 30
```

```
for num in numbers:
if num == search:
print("Found:", num)
break
else:
print("Not found")
```

---

2. Python Strings

What is a String?

A string is a sequence of characters enclosed in quotes. Strings are immutable (cannot be changed after creation).

Create a Python String

Different ways to create strings:

Single quotes

```
name = 'Python'
```

Double quotes

```
language = "Programming"
```

Triple quotes (multi-line)

```
message = """This is a
multi-line
string"""
```

Empty string

empty = ""

---

3. Indexing and Negative Indexing

What is Indexing?

Accessing individual characters using their position. Index starts at 0.

Index Type	Description
Positive (0, 1, 2...)	Count from left to right
Negative (-1, -2, -3...)	Count from right to left

Table 2: String Indexing

Code Examples

Example 1: Positive Indexing
text = "Python"

012345

print(text[0]) # P
print(text[1]) # y
print(text[5]) # n

Example 2: Negative Indexing
text = "Python"

-6-5-4-3-2-1

print(text[-1]) # n
print(text[-2]) # o
print(text[-6]) # P

---

4. Slicing

What is Slicing?

Extract a portion of string using `[start:stop:step]`.

Syntax: `string[start:stop:step]`

- start: Starting index (inclusive)
- stop: Ending index (exclusive)
- step: Jump between characters (default 1)

Code Examples

Example 1: Basic Slicing
```
text = "Python Programming"
```

```
print(text[0:6]) # Python
print(text[7:18]) # Programming
print(text[:6]) # Python (from start)
print(text[7:]) # Programming (to end)
```

Example 2: Using Step
```
text = "Python"
```

```
print(text[::2]) # Pto (every 2nd character)
print(text[1::2]) # yhn (start at 1, every 2nd)
print(text[::-1]) # nohtyP (reverse string)
```

Example 3: Negative Indices
```
text = "Programming"
```

```
print(text[-4:]) # ming (last 4 characters)
print(text[:-4]) # Program (except last 4)
```

---

5. Edit and Delete a String

Important Point

Strings are immutable - cannot be changed directly. Must create new string.

Code Examples

Example 1: Cannot Edit Directly
```
text = "Python"
```

text[0] = 'J' # ERROR: strings are immutable

Example 2: Create New String
```
text = "Python"
```

Replace by creating new string

```
new_text = 'J' + text[1:]
print(new_text) # Jython
```

Example 3: Delete String
```
text = "Python"
del text # Delete entire string
```

print(text) # ERROR: text no longer exists

6. Operations on a String

Common String Operations

Operation	Operator	Example
Concatenation	+	"Hello" + "World"
Repetition	*	"Ha" * 3 = "HaHaHa"
Membership	in	"a" in "Python"
Length	len()	len("Python") = 6

Table 3: String Operations

Code Examples

Example 1: Concatenation
```
first = "Hello"
last = "World"
result = first + " " + last
print(result) # Hello World
```

Example 2: Repetition
```
text = "Ha"
print(text * 3) # HaHaHa

line = "-" * 20
print(line) # --------------------
```

Example 3: Membership
```
text = "Python Programming"

print("Python" in text) # True
print("Java" in text) # False
print("Java" not in text) # True
```

Example 4: Length
```
text = "Python"
print(len(text)) # 6

name = "Bikkad IT Institute"
print(len(name)) # 19
```

7. Common String Functions

Capitalize, Title, Upper, Lower, Swapcase

Function	Description
capitalize()	First character uppercase
title()	First character of each word uppercase
upper()	All characters uppercase
lower()	All characters lowercase
swapcase()	Swap case of all characters

Table 4: Case Conversion Functions

Code Examples

```
text = "python programming"

print(text.capitalize()) # Python programming
print(text.title()) # Python Programming
print(text.upper()) # PYTHON PROGRAMMING
print(text.lower()) # python programming

text2 = "PyThOn"
print(text2.swapcase()) # pYtHoN
```

---

8. Count, Find, Index Functions

Function	Description
count(sub)	Count occurrences of substring
find(sub)	Find first occurrence (returns -1 if not found)
index(sub)	Find first occurrence (raises error if not found)

Table 5: Search Functions

Code Examples

Example 1: count()

```
text = "Python Programming"

print(text.count('P')) # 2
print(text.count('o')) # 2
print(text.count('gram')) # 1
```

Example 2: find()

```
text = "Python Programming"
```

```
print(text.find('P')) # 0
print(text.find('gram')) # 10
print(text.find('Java')) # -1 (not found)
```

Example 3: index()

```
text = "Python Programming"

print(text.index('P')) # 0
print(text.index('gram')) # 10
```

print(text.index('Java')) # ERROR: substring not found

---

9. endswith and startswith

Function	Description
startswith(prefix)	Check if string starts with prefix
endswith(suffix)	Check if string ends with suffix

Table 6: Prefix/Suffix Check Functions

Code Examples

Example 1: startswith()

```
filename = "report.pdf"

print(filename.startswith("report")) # True
print(filename.startswith("doc")) # False

text = "Python Programming"
print(text.startswith("Python")) # True
```

Example 2: endswith()

```
filename = "document.pdf"

print(filename.endswith(".pdf")) # True
print(filename.endswith(".txt")) # False

email = "user@gmail.com"
print(email.endswith("@gmail.com")) # True
```

---

10. format() Function

What is format()?

Insert values into string using placeholders {}.

Code Examples

Example 1: Basic Format

```
name = "Rahul"
age = 25

message = "My name is {} and I am {} years old".format(name, age)
print(message)
```

Output: My name is Rahul and I am 25 years old

Example 2: Positional Arguments

```
text = "{0} is {1} years old. {0} lives in Nashik.".format("Raj", 22)
print(text)
```

Output: Raj is 22 years old. Raj lives in Nashik.

Example 3: Named Arguments

```
message = "Name: {name}, Age: {age}, City: {city}".format(
name="Priya",
age=20,
city="Nashik"
)
print(message)
```

Example 4: f-strings (Python 3.6+)

```
name = "Amit"
marks = 85

message = f"Student {name} scored {marks} marks"
print(message)
```

---

11. isalnum, isalpha, isdigit, isidentifier

Function	Returns True if
isalnum()	All characters are alphanumeric (a-z, A-Z, 0-9)
isalpha()	All characters are alphabets (a-z, A-Z)
isdigit()	All characters are digits (0-9)
isidentifier()	String is valid Python identifier

Table 7: String Validation Functions

Code Examples

Example 1: isalnum()

```
print("Python3".isalnum()) # True
print("Python 3".isalnum()) # False (space)
print("12345".isalnum()) # True
```

Example 2: isalpha()

```
print("Python".isalpha()) # True
print("Python3".isalpha()) # False (digit)
print("Hello World".isalpha()) # False (space)
```

Example 3: isdigit()

```
print("12345".isdigit()) # True
print("123.45".isdigit()) # False (dot)
print("12a34".isdigit()) # False (letter)
```

Example 4: isidentifier()

```
print("variable_name".isidentifier()) # True
print("_age".isidentifier()) # True
print("2variable".isidentifier()) # False (starts with digit)
print("my-var".isidentifier()) # False (hyphen)
```

---

12. split() and join()

Function	Description
split(separator)	Split string into list
join(iterable)	Join list into string

Table 8: Split and Join Functions

Code Examples

Example 1: split()

```
text = "Python Programming Language"

words = text.split() # Split by space (default)
print(words) # ['Python', 'Programming', 'Language']
```

```
csv = "Apple,Banana,Mango,Orange"
fruits = csv.split(",") # Split by comma
print(fruits) # ['Apple', 'Banana', 'Mango', 'Orange']
```

Example 2: join()

```
words = ['Python', 'Programming', 'Language']

text = " ".join(words) # Join with space
print(text) # Python Programming Language

csv = ",".join(words) # Join with comma
print(csv) # Python,Programming,Language
```

Example 3: Practical Use

Convert sentence to list and back

```
sentence = "Learn Python Programming"
word_list = sentence.split()
print(word_list)

new_sentence = "-".join(word_list)
print(new_sentence) # Learn-Python-Programming
```

---

13. replace() Function

What is replace()?

Replace occurrences of substring with another substring.

Syntax: `string.replace(old, new, count)`

Code Examples

Example 1: Basic Replace

```
text = "Python Programming"

new_text = text.replace("Python", "Java")
print(new_text) # Java Programming
```

Example 2: Replace All Occurrences

```
text = "I love Python. Python is easy."

new_text = text.replace("Python", "Java")
print(new_text) # I love Java. Java is easy.
```

Example 3: Replace with Count

```
text = "one one one one"

new_text = text.replace("one", "two", 2) # Replace first 2
print(new_text) # two two one one
```

Example 4: Remove Character

```
phone = "123-456-7890"

clean = phone.replace("-", "")
print(clean) # 1234567890
```

---

14. strip() Function

What is strip()?

Remove leading and trailing characters (default: whitespace).

Function	Description
strip()	Remove from both sides
lstrip()	Remove from left side only
rstrip()	Remove from right side only

Table 9: Strip Functions

Code Examples

Example 1: strip() - Remove Whitespace

```
text = " Python "

print(text.strip()) # "Python"
print(text.lstrip()) # "Python "
print(text.rstrip()) # " Python"
```

Example 2: strip() - Remove Specific Characters

text = "***Python***"

```
print(text.strip("*")) # Python

url = "https://example.com/"
print(url.strip("https://")) # example.com/
```

Example 3: Practical Use - Clean User Input

Remove extra spaces from user input

```
name = input("Enter name: ") # User enters " Rahul "
clean_name = name.strip()
print(f"Welcome, {clean_name}!")
```

Practice Exercises

Exercise 1: String Slicing

Create a string "Python Programming" and extract:

- First word
- Last word
- Reverse the string

Exercise 2: String Methods

Take a sentence as input and:

- Convert to uppercase
- Count number of words
- Replace a word

Exercise 3: String Validation

Write a program to check if a string is:

- All alphabets
- All digits
- Valid identifier

Exercise 4: Split and Join

Take a comma-separated string and convert it to space-separated.

Exercise 5: Clean Text

Remove all spaces from a string and convert to lowercase.

Exercise 6: Search in String

Search for a substring in a string using find() and count occurrences.

Exercise 7: Format String

Create a formatted string with name, age, and city using format() or f-strings.

Exercise 8: Break and Continue

Print numbers 1-20, skip multiples of 3, stop at 15.

Quick Reference

Concept	Key Function/Syntax
Indexing	text[0], text[-1]
Slicing	text[start:stop:step]
Concatenation	"Hello" + "World"
Length	len(text)
Case	upper(), lower(), title()
Search	find(), index(), count()
Check	startswith(), endswith()
Validation	isalpha(), isdigit(), isalnum()
Split	split(separator)
Join	separator.join(list)
Replace	replace(old, new)
Strip	strip(), lstrip(), rstrip()
Format	format(), f-strings
Break	Exit loop
Continue	Skip iteration
Pass	Do nothing

Table 10: Python Strings Quick Reference

End of Notes

Prepared by: Bikkad IT Institute
For: Python Beginner Level Students
Date: February 2026