Module 1l: Introduction to Problem Solving and Python Fundamentals

Premanand S

Assistant Professor,
School of Electronics and Engineering,
Vellore Institute of Technology, Chennai

premanand.s@vit.ac.in

August 2, 2025

What is a String in Python?

- A **string** is a sequence of Unicode characters.
- It is used to store text — words, sentences, symbols, etc.
- Strings are enclosed in single (’), double (") or triple quotes (’’’, """)
- Strings are **immutable** — cannot be changed after creation.

Example Strings

```
s1 = ’Hello’
s2 = "Python"
s3 = """Multiline
String"""
```

String Indexing and Slicing

- Strings are indexed from 0 (left to right).
- Negative indexing accesses from the end.
- Slicing extracts a part of the string: `string[start:end]`

Examples

```
text = "Python"
print(text[0])      # 'P'
print(text[-1])     # 'n'
print(text[1:4])    # 'yth'
```

String Operations and Concatenation

- Use + for concatenation
- Use * to repeat a string
- Use `in` to check membership

Examples

```
a = "Hi"
b = "There"
print(a + b)        # 'HiThere'
print(a * 3)        # 'HiHiHi'
print("H" in a)     # True
```

How to Explore String Methods

- Use `dir()` to list all available string methods.
- Use `help()` to get detailed documentation of any method.
- Useful in Jupyter, IDLE, or Python interactive shell.

Try These in Python

```
s = "Python"

print(dir(s))          # Lists all string methods

help(str.upper)        # Shows docstring for .upper() method

print(s.isalpha())     # True (if all characters are letters)
print(s.startswith("P"))  # True
```

1. upper()

Converts all characters in a string to uppercase.

Example: upper()

```
s = "hello" print(s.upper())
```

2. lower()

Converts all characters in a string to lowercase.

Example: lower()

```
s = "HELLO" print(s.lower())
```

3. title()

Capitalizes the first character of each word in the string.

Example: title()

```
s = "python programming" print(s.title())
```

4. capitalize()

Capitalizes only the first character of the string.

Example: capitalize()

```
s = "hello world" print(s.capitalize())
```

5. strip(), lstrip(), rstrip()

Removes whitespace or specified characters from the beginning and/or end.

Example: strip()

```
s = " hello " print(s.strip()) print(s.lstrip())
print(s.rstrip())
```

6. replace()

Replaces all occurrences of a specified substring with another substring.

Example: replace()

```
s = "apple banana apple" print(s.replace("apple",
"orange"))
```

7. find()

Returns the index of the first occurrence of a substring, or -1 if not found.

Example: find()

```
s = "hello world" print(s.find("world"))
print(s.find("Python"))
```

8. index()

Returns the index of the first occurrence of a substring. Raises ValueError if not found.

Example: index()

```
s = "hello world" print(s.index("world"))
print(s.index("Python"))
```

9. count()

Counts how many times a substring appears in the string.

Example: count()

```
s = "apple apple orange" print(s.count("apple"))
```

10. split()

Splits the string into a list using the specified separator.

Example: split()

```
s = "apple,banana,orange" print(s.split(","))
```

11. join()

Joins the elements of an iterable with a string separator.

Example: join()

```
fruits = ["apple", "banana", "orange"]
print(",".join(fruits))
```

12. startswith() and endswith()

Check if a string starts or ends with a specified substring.

Example: startswith()

```
s = "python programming" print(s.startswith("python"))
print(s.endswith("ing"))
```

13. Character-type checks

Check if a string contains only alphabetic, numeric, or whitespace characters.

Example: isalpha()

```
print("abc".isalpha()) print("123".isdigit())
print("abc123".isalnum()) print(" ".isspace())
```

14. Case-checking methods

Check the case format of a string.

Example: isupper()

```
print("HELLO".isupper()) print("hello".islower())
print("Hello World".istitle())
```

15. zfill()

Pads a string from the left with zeros to make it a given width.

Example: zfill()

```
print("42".zfill(5))
```

16. format() and f-strings

Used for string formatting with variables.

Example: format() and f-strings

```
name = "Prem" age = 38 print("Name: , Age:
".format(name, age)) print(f"Name: name, Age: age")
```

- `f-strings` allow embedding expressions inside string literals.
- Syntax: prefix string with `f` and place variables inside {}.
- Faster and more readable than `.format()` or concatenation.
- Supports expressions and formatting options like precision, padding, etc.

Examples

```
name = "Prem"
score = 92.568
print(f"Hi {name}, your score is {score:.2f}")

x = 10
y = 5
print(f"Sum: {x + y}")
```

Escape Characters in Strings

- Escape characters begin with \
- Used to insert special characters like newline, tab, quotes

Examples

```
print("Line1\nLine2")       # New line
print("She said \"Hi\"")    # Double quote
print("Path\\to\\file")     # Backslash
```

Email Slicer

Extract username and domain from an email address.

Example: Email Slicer

```
email = "premanand.s@vit.ac.in" username =
email.split("@")[0] domain = email.split("@")[1]
print(f"Username: username, Domain: domain")
```

University Email Generator

Create an email ID using student's name: first letter + last name.

Example: University Email Generator

```
name = "Premanand" parts = name.lower().split()
email_id = parts[0][0] + parts[1] + "@university.edu" print(email_id)
```

Text Sanitizer

Replace bad words with censored versions.

Example: Text Sanitizer

```
comment = "This game is stupid and dumb" comment =
comment.replace("stupid", "s****").replace("dumb",
"d***") print(comment)
```

Sentence Capitalizer

Capitalize the first letter of every sentence in a paragraph.

Example: Sentence Capitalizer

```
text = "hello world. python is fun. welcome!"
sentences = text.split('. ') result = '.
'.join([s.capitalize() for s in sentences])
print(result)
```

Initials Extractor

Extract and print initials from a full name.

Example: Initials Extractor

```
name = "Santhalakshmi Premanand Nikhilesh Krithiksha"
initials = ".".join([word[0].upper() for word in
name.split()]) print(initials)
```

Chat Message Shortener

Task: Replace common words with short forms.

SMS Shortener Code

```
msg = "see you later at the meeting"
short = msg.replace("see you", "cu").replace("later", "l8r"
short = short.replace("at", "@").replace("meeting", "mtg")
print(short)
```

Domain Name Extractor

Task: Extract domain from a URL.

Domain Extractor Code

```
url = "https://www.google.com/search?q=python"
domain = url.split("//")[1].split("/")[0]
print(domain)  # www.google.com
```

Assignment 1: Create and Print Strings

- Create strings to store:
  - Your full name
  - Your college and city
  - A motivational quote
- Print each string using `print()` and `type()`.

Assignment 2: Indexing and Slicing Strings

- Use a string variable: `name = "PythonProgramming"`
- Print:
  - First character
  - Last character
  - Substring from index 2 to 7
  - Every second character
  - Reversed string

Assignment 3: String Methods Practice

- Given a string: `text = " Welcome to Python Programming "`
- Perform the following:
  - Convert to uppercase and lowercase
  - Remove leading/trailing spaces
  - Replace "Python" with "Coding"
  - Count occurrences of 'o'
  - Check if the string starts with "Welcome"

Assignment 4: f-String Formatting

- Input name, age, and CGPA from the user.
- Use an f-string to print: `"Hello [Name], age: [Age], CGPA: [CGPA]"`
- Display CGPA with 2 decimal places.
- Add a line showing current year and next year using expressions.

Assignment 5: Email Parser Challenge

- Given an email: `"student123@college.edu"`
- Extract and print:
  - Username
  - Domain name
  - Email provider (part after `@` but before `.`)
- Use slicing, `find()`, and `split()`.

mail me: er.anandprem@gmail.com / premanand.s@vit.ac.in
ring me: +91 73586 79961
Follow me: Linkedin
Medium Blogs
Analytics Vidhya: Blogs

Don't just code — think, plan, and solve