1. Double Quote Issue (SyntaxError)

```python
chai = "He said, "Masala chai is awesome" "
```

🔴 **Error:**

This gives a `SyntaxError` because Python gets confused — it thinks the string ends at the second double quote (`"He said, "`), then sees `"Masala...` as invalid code.

✅ **Fix it using escape character** `\"` :
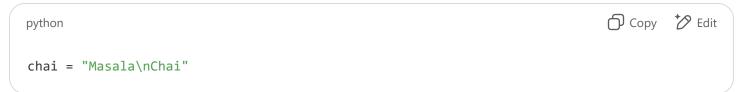
```python
chai = "He said, \"Masala chai is awesome\" "
```

This tells Python to include the quotes **inside** the string instead of ending it.

✅ 2. New Line with `\n`

```python
chai = "Masala\nChai"
```

- `\n` means **new line**.
- When you `print(chai)`, it looks like:

```nginx
Masala
Chai
```
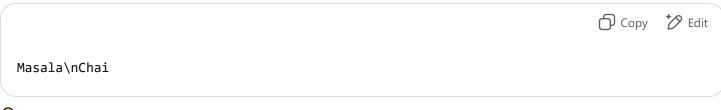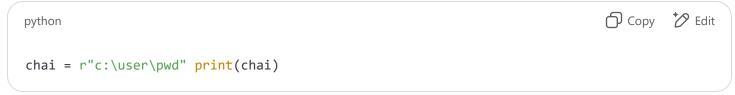

✅ 3. Raw String (`r""`)

```python
```

```python
chai = r"Masala\nChai"
```

- `r""` **means raw string** → Python treats `\n` as **normal characters**, not as newline.
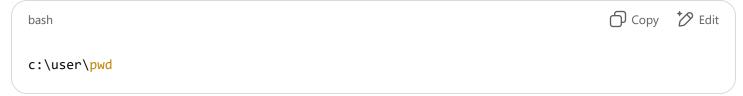- So `print(chai)` gives:

Copy   Edit

```
Masala\nChai
```

💡 Use raw strings when you **don't want Python to process backslashes**, like in file paths.

✅ 4. File Path with Raw String

python   Copy   Edit

```python
chai = r"c:\user\pwd" print(chai)
```

Output:

bash   Copy   Edit

```
c:\user\pwd
```

- Without `r`, `\u` and `\p` might be read as Unicode escape sequences — which causes an error.

🔴 5. Unicode Error Without `r`

python   Copy   Edit

```python
chai = "c:\user\pwd"
```

❌ This gives:

vbnet   Copy   Edit

```
SyntaxError: (unicode error) 'unicodeescape'
```

- Because `\u` expects a Unicode character like `\u1234`, but here it's invalid (`\user`).
- ✅ Always use **raw string** for Windows paths: `r"c:\user\pwd"`

✅ Summary

Purpose	Example	Output / Effect
Escape quotes inside string	`"He said, \"Hi!\""`	He said, "Hi!"
New line	`"Hello\nWorld"`	Hello World
Raw string	`r"Hello\nWorld"`	Hello\nWorld (no new line)
File path	`r"c:\user\pwd"`	✅ No error
File path (❌ wrong)	`"c:\user\pwd"`	❌ Unicode error