

A decorative graphic on the left side of the slide. It consists of a blue parallelogram and a light green parallelogram, both tilted at an angle. The blue shape is in the foreground, and the green shape is partially behind it. They are set against a dark blue background with faint, lighter blue diagonal stripes.

Python String Formatting



What's String Formatting?

- Standard method of manipulating strings nicely
- Include variables in output
- Useful in any I/O program
- Nice visualization

```
PID  TTY          TIME CMD
  1  ?           00:00:00 init
  5  tty1         00:00:00 init
  6  tty1         00:00:00 bash
 32  tty1         00:00:00 ps
```



How to do String Formatting? (Python)

- String Concatenation
- Python String .format() method
- String slicing and indexing



String Concatenation

- Python lets you concatenate strings using '+'
- In order to use this operation, each side of the + sign must be a string
- Should be used for basic string formatting only

```
>>> str1 = "Hello "  
>>> str2 = "World"  
>>> str3 = str1 + str2  
>>> print(str3)  
Hello World  
>>>
```



String Concatenation - including variables

- What if we wanted to include variables?
- Use `str()` typecasting
- Can use string literals to include spaces

```
>>>
>>> str1 = "Hello "
>>> int1 = 100
>>> str2 = "World"
>>> str3 = str1 + str(int1) + " " + str2
>>> print(str3)
Hello 100 World
>>>
```



Python String format Method

- Easier to use, modify, and read
- Place '{}' in string where you want to include variables
- Then, at the end of the string call .format() and include the variables that you want to include in the order they appeared
- No type casting needed!

```
>>> int1 = 100
>>> int2 = 3
>>> str1 = "{} / {} = {}".format(int1, int2, int1/int2)
>>> print(str1)
100 / 3 = 33.333333333333336
```



Advanced Python String formatting

- What if I don't want all of those 3's???
- Try rounding!
- Change the {} with {:.yf}
 - x must be an integer representing the minimum number of characters used to print the string
 - If x > length of string, then spaces are filled on the left hand side of the string
 - y must be an integer representing the number of decimals you want
 - f indicates that the variable in the format is a float.

```
>>> int1 = 100
>>> int2 = 3
>>> str1 = "{:.2f} / {:.2f} = {:.2f}".format(int1, int2, int1/int2)
>>> print(str1)
100.00 / 3.00 = 33.33
```



Advanced String Formatting (cont.)

There's so many things you can do with `.format()`, where do I start?

- [Python docs](#)
- [Geeks for Geeks](#)
- [W3 Schools](#)

```
s – strings  
d – decimal integers (base-10)  
f – floating point display  
c – character  
b – binary  
o – octal  
x – hexadecimal with lowercase letters after 9  
X – hexadecimal with uppercase letters after 9  
e – exponent notation
```

From: Geeks for Geeks

A decorative graphic in the top-left corner consisting of two overlapping parallelograms. The front one is blue and the back one is a light green. They are both tilted at a 45-degree angle.

Large Group Project 1: Matching Output



String Indexing/ Slicing

- String Indexing works just like list indexing
 - To get the first index of string, simply use `s[0]`
- To get a chunk of the string use slicing
 - Syntax: `s[startIndex : stopIndex + 1]`
 - If you want to start at the beginning, remove `stopIndex`. Syntax: `s[: stopIndex + 1]`
 - If you want to end at the end of the string, remove `stopIndex + 1`. Syntax: `s[startIndex:]`

```
>>> string1 = "Hello World!"
>>> print(string1[:5])
Hello
>>> print(string1[6:])
World!
>>> print(string1[3:9])
lo Wor
```



String Indexing/ Slicing (cont.)

- What if you only want to include a chunk at the end or remove a chunk at the end, but you don't know how long the string is?
- Use negative numbers
- Removing chunks of string at the end:
 - Syntax: `s[:-n]` where `n` is the number of characters to remove
- Only including a chunk at the end:
 - Syntax: `s[-n:]` where `n` is the number of characters

```
>>> string1 = "Hello World!"
>>> print(string1[:-1])
Hello World
>>> print(string1[-6:])
World!
```

A decorative graphic on the left side of the slide consisting of two overlapping parallelograms. The front one is blue and the back one is a light green color. They are positioned diagonally, with the blue one in front of the green one.

Large Group Project 2: Making Tables

A decorative graphic on the left side of the slide. It consists of a blue parallelogram and a light green parallelogram, both tilted at an angle. The blue shape is in the foreground, and the green shape is partially behind it. They are set against a dark blue background with faint, lighter blue diagonal stripes.

Pair Project Pig Latin