

The logo for COMP 110, featuring the text "COMP" on the top line and "110" on the bottom line. The text is rendered in a stylized, rounded font with a vibrant rainbow-to-cyan gradient. The logo is centered within a dark gray square background.

COMP  
110

# CL04 - Practice with while loops and functions

# Practice Writing Functions

Write a mimic function: you input a string and it **returns** the same string back to you

- Function name: `mimic`
- Parameters: `my_words: str`
- Return type: `str`
- Doc string: `"""Given the string my_words, outputs the same string"""`

Try calling it!

## Expected Code:

```
def mimic(my_words: str) -> str:
    """Given the string my_words, outputs the same string"""
    return my_words
```

## Calling it:

```
mimic("Hello!")
```

```
print(mimic("Hello!"))
```

```
my_words: str = "Hello!"
response: str = mimic(my_words)
print(response)
```

# Practice Writing Functions

Write a different mimic function: you input a string and an index and it **returns** the letter at that index. If the index is too high for the string length, **return** "Too high of an index".

E.g. `mimic_letter("hello",0)` returns "h", `mimic_letter("howdy",2)` returns "w", `mimic_letter("hi",3)` returns "Too high of an index"

Function name: `mimic_letter`

- Parameters: `my_words: str, letter_idx: int`
- Return type: `str`
- Doc string: `"""Outputs the character of my_words at index letter_idx"""`

## Expected Code:

```
def mimic_letter(my_words: str, letter_idx: int):  
    """Outputs the character of my_words at index letter_idx"""  
    if letter_idx >= len(my_words):  
        return("Index too high")  
    #If we made it here, that means the letter_idx is valid  
    return my_words[letter_idx]
```

# Memory Diagram

```
1  xs: str = "123"
2  ys: str = "45"
3
4  x_idx: int = 0
5  √ while x_idx < len(xs):
6  |   y_idx: int = 0
7  |   √ while y_idx < len(ys):
8  |   |   print(f"({xs[x_idx]},{ys[y_idx]}")
9  |   |   y_idx = y_idx + 1
10 |   x_idx = x_idx + 1
```

# Memory Diagram

```
1  def main():
2  |     """Main code of program"""
3  |     y: float = double(2.0)
4  |     print(halve(y))
5
6  def halve(x: float) -> float:
7  |     """Returns half the value of x"""
8  |     print(f"halve({x})")
9  |     return x / 2.0
10
11 def double(x: float) -> float:
12 |     """Double a value"""
13 |     print(f"double({x})")
14 |     return x * 2.0
15
16 main()
```