Python Skill Practice Exercises

🧵 Strings

1. Extract the 6th character from the string "Python programming is fun"

2. Find the character `'g'` from the string "Python programming is fun"

3. Use negative indexing to extract `'n'` from the string

4. From the string "Coding every day makes you better":
   a. Slice out the word `every`
   b. Retrieve the word `day` using a negative index range

5. Given the string "Learning to code is a valuable skill":
   a. Use slicing to print every second character
   b. Use slicing to print every third character

6. For `s = "Practice makes perfect"`, try these: a. `s[:]` b. `s[0:]` c. `s[:18]` d. Just `s`

7. Convert the string `"67890"` into an integer and verify the data type

---

📋 Lists

1. Create: a. A list of 5 numbers b. A list of 4 city names c. A mixed list with both strings and numbers

2. Perform these list operations: a. Get the number of elements in the list
   b. Display each element using a loop
   c. Add an element to the end
   d. Insert an element at the beginning
   e. Insert an element at position 3
   f. Combine two lists using both `+` and `.extend()` methods and observe differences
   g. Retrieve the 4th item from a list
   h. Retrieve items from index 1 to 3
   i. Replace an item with a new one

j. Add elements to an initially empty list

3. Simulate a stack using list operations

4. Simulate a queue using list operations

5. Given URLs:

urls = ["www.alpha.edu", "www.search.com", "www.newsnet.com", "www.site.org", "www.tech.gov.in"]

Sort based on their domain extension

6. Given:

color_names = ["Cyan", "Magenta", "Yellow", "Black"]
color_codes = [202, 303, 404, 505]

Perform: a. Print all colors, then print only the first color
b. Print the list twice
c. Combine color names and codes into a single list
d. Print from 2nd code to last
e. Show the data type of the final code item

7. Write `def count_same_ends(words)` that returns the count of words with:

   ○ length ≥ 2, and

   ○ same starting and ending character

📂 Tuples

1. Create an empty tuple

2. Create a tuple with a single value

3. Create one tuple with numbers and one with strings

4. Create a tuple of lists and a list of tuples

5. Access the 4th element of a tuple

6. Access elements from index 1 to 3

7. Write `def sort_by_last(tuples)` to sort a list of tuples by the last element in each

---

📆 Dictionaries

1. Use the following mapping:

```
paint_codes = {
  'p01': 'sky blue', 'p02': 'crimson red', 'p03': 'ice gray',
  'p04': 'sunshine yellow', 'p05': 'lime green'
}
```

Perform: a. Print only the color values
b. Print only the keys
c. Get value for key `'p02'`
d. Fetch a value using `.get()`
e. Combine keys and values into a list
f. Print from second item onward
g. Change the value of `'p04'` and verify

2. Use:

```
students = {
  "alice": {"math": 90, "science": 85},
  "bob": {"math": 78, "english": 88},
  "carol": {"math": 92, "english": 95, "science": 89}
}
```

Access: a. alice's science marks b. bob's english marks c. carol's math marks

---

📃 Sets

1. Given:

```
primary_colors = ["red", "blue", "yellow", "blue"]
secondary_colors = ["green", "purple", "orange", "red"]
```

Perform: a. Print unique primary colors
b. Print each color in a separate line
c. Merge both lists and show the combined set
d. Print colors that exist in both

📄 Files & Functions

1. Write to a file `sample1.txt` and copy contents to `sample2.txt`

2. Implement a calculator with functions:

    - `add`, `subtract`, `multiply`, `divide`, `sqrt`

3. Create a function that accepts any number of arguments (`*args`) and keyword arguments (`**kwargs`)

4. Use this data:

```
library = {
 "FICTION": ["1984", "George Orwell", "1949", "19.99"],
 "TECH": ["Clean Code", "Robert C. Martin", "2008", "34.95"]
}
```

Print formatted output for each book with title, author, year, and price

---

⚡ Exception Handling

1. Handle exceptions for:

```
nums = [1, 2, 3]
nums[5]

s = {1, 2, 3}
s.remove(4)
```

2. What happens when trying:

```
val = 5 / 0
```

3. Modify this to handle scope error:

```
def outer():
    x = 10
    def inner():
        x += 1
        print(x)
    inner()
outer()
```

4. Raise and handle:
    - `IOError`, `IndexError`, `KeyError`, `NameError`, `SyntaxError`, `TypeError`, `ValueError`, `ZeroDivisionError`, `StopIteration`

🔎 Regular Expressions

1. Write regex for: a. A string starting with a digit and ending with a digit
   b. A string with only alphabets and whitespace
   c. A string with no whitespace at all

2. Use regex to find words in a file with **two consecutive vowels**

3. Find words with **exactly two vowels** anywhere inside

4. Explain what these do:
    - `r"hello.*world"`
    - `r"^\d+\s*$"`
    - `r"\b[a-z]+\b"`
    - `r"(.)\1*"`

5. Create a regex to extract IP addresses from text files

6. Try these:

```
re.split('\W+', 'Test, test, test.')
re.split('(\W+)', 'Test, test, test.')
re.split('\W+', 'Test, test, test.', 1)
re.split('[a-c]+', 'abcABC123', flags=re.IGNORECASE)
```

7. Convert vowels to uppercase from a file

8. From:

```
log = 'Sent from dev.user123@demo.org on Tue Jan 03 14:55:22 2023'
```

Extract: a. email address b. domain name c. time string

9. From `"Python makes learning easy"`, extract each word as a group using regex