Homework

Welcome to the world of Python programming! This set of exercises will help you build a strong foundation in:

1. **Variables**: Storing and manipulating data.
2. **Data Types**: Understanding different types of information like numbers, strings, and Boolean.
3. **Data Structures**: Managing collections like lists, dictionaries, tuples, and sets.
4. **Basic Operations**: Performing calculations, comparisons, and logical operations.
5. **String Manipulations**: Working with text effectively.

How to Approach These Exercises

1. Start with **Variables** to understand how to store and update information.
2. Move to **Data Types** to explore various kinds of data Python can handle.
3. Practice **Data Structures** to organize your data efficiently.
4. Use **Basic Operations** to manipulate data and build logic.
5. Finally, enjoy solving challenges with **String Manipulations** to master text handling.

Each exercise is designed to enhance your problem-solving skills step by step.

Remember to:

- Test your code frequently.
- Experiment with modifications to deepen your understanding.
- Ask questions if you're unsure about any concept.

Ready? Let’s Get Started! 🚀

1. Variables

1. Declare a variable and assign it your name.
2. Create two variables and assign them integer values, then print their sum.
3. Swap the values of two variables without using a third variable.
4. Assign a floating-point number to a variable and display it.
5. Assign a boolean value to a variable and print it.
6. Update the value of a variable and print the updated value.
7. Declare a constant-like variable using uppercase and demonstrate why Python allows updates.
8. Create a variable with a null value and then assign it a string.
9. Create three variables in a single line with the same value.
10. Create three variables in a single line with different values.
11. Assign multiple types (int, float, string) to variables and print their types.
12. Create a variable with a string containing your name and age using placeholders.
13. Use the del keyword to delete a variable and try accessing it after deletion.
14. Use type hinting to declare a variable and assign it a value.
15. Assign the same value to multiple variables and verify their memory addresses.
16. Assign a value to a variable using an expression.
17. Print a variable without assigning it a value.
18. Assign a list of numbers to a variable and print the sum of its elements.
19. Create a variable inside a function and try accessing it outside.
20. Use global and local variables in a small function example.

Solution:

variables_exercise.py

Python Programming > Week 01 > variables_exercise.py > ...

```python
# 1
name = "Your Name"

# 2
a, b = 10, 20
print(a + b)

# 3
x, y = 5, 10
x, y = y, x

# 4
float_num = 3.14
print(float_num)

# 5
is_active = True
print(is_active)

# 6
num = 10
num += 5
print(num)

# 7
PI = 3.14159
PI = 3  # Python allows this but avoid modifying constants

# 8
value = None
value = "Now assigned"
print(value)

# 9
x = y = z = 5

```

```python
# 10
x, y, z = 1, 2, 3

# 11
integer = 42
float_num = 3.14
string = "text"
print(type(integer), type(float_num), type(string))

# 12
info = f"My name is {name}, and I am {25} years old."
print(info)

# 13
var = 10
del var
# print(var)  # Will raise a NameError

# 14
name: str = "John"
age: int = 30

# 15
a = b = 100
print(id(a), id(b))

# 16
expr_var = 10 * 5

# 17
# print(not_declared)  # Will raise NameError

# 18
numbers = [1, 2, 3, 4, 5]
print(sum(numbers))

```

```python
# 19
def my_function():
    local_var = "Hello"
# print(local_var)  # Will raise NameError

# 20
global_var = 10

def modify_global():
    global global_var
    global_var = 20
modify_global()
print(global_var)

```

---

2. Data Types

1. Identify the type of 42.
2. Identify the type of 3.14.
3. Identify the type of "Hello".
4. Identify the type of True.
5. Identify the type of None.
6. Convert an integer to a float.
7. Convert a float to an integer.
8. Convert a string to an integer.
9. Convert a string to a list.
10. Identify the type of {'key': 'value'}.
11. Identify the type of [1, 2, 3].

12. Identify the type of (1, 2, 3).

13. Identify the type of {1, 2, 3}.

14. Demonstrate how Python dynamically changes the type of a variable.

15. Check if a variable is of type integer.

16. Create a complex number and print its real and imaginary parts.

17. Demonstrate type casting by converting a boolean to an integer.

18. Create a variable with mixed data types using list.

19. Use isinstance() to verify a variable's data type.

20. Write a program to check if an input is a number.

Solution:

datatype_exercise.py

Python Programming > Week 01 > datatype_exercise.py > ...

```python
# 1 - 5
print(type(42), type(3.14), type("Hello"), type(True), type(None))

# 6
print(float(42))

# 7
print(int(3.99))

# 8
print(int("42"))

# 9
print(list("text"))

# 10
print(type({"key": "value"}))

# 11 - 13
print(type([1, 2, 3]), type((1, 2, 3)), type({1, 2, 3}))

# 14
a = 5
a = "text"

# 15
print(isinstance(42, int))

# 16
complex_num = 3 + 4j
print(complex_num.real, complex_num.imag)

# 17
print(int(True))

```

```python
# 18
mixed_list = [1, "text", 3.14]

# 19
print(isinstance(42, int))

# 20
try:
    user_input = int(input("Enter a number: "))
    print("It is a number!")
except ValueError:
    print("Not a number.")

```

---

3. Data Structures

1. Create a list of five numbers and print it.
2. Append an element to a list.
3. Insert an element at the beginning of a list.
4. Remove an element from a list by value.
5. Remove an element from a list by index.
6. Sort a list in ascending order.
7. Reverse a list.
8. Create a dictionary with three key-value pairs.
9. Access a value in a dictionary using its key.
10. Add a new key-value pair to a dictionary.
11. Remove a key-value pair from a dictionary.

12. Create a tuple with five elements.

13. Access the second element of a tuple.

14. Unpack a tuple into separate variables.

15. Create a set and add an element to it.

16. Remove an element from a set.

17. Check if an element exists in a set.

18. Perform a union operation on two sets.

19. Perform an intersection operation on two sets.

20. Create a nested dictionary and access an inner value.

Solution:

data_structure.py

Python Programming > Week 01 > data_structure.py > ...

```python
# 1
numbers = [1, 2, 3, 4, 5]
print(numbers)

# 2
numbers.append(6)

# 3
numbers.insert(0, 0)

# 4
numbers.remove(3)

# 5
numbers.pop(2)

# 6
numbers.sort()

# 7
numbers.reverse()

# 8
my_dict = {"name": "John", "age": 30, "job": "Engineer"}

# 9
print(my_dict["name"])

# 10
my_dict["location"] = "USA"

# 11
del my_dict["age"]
```

```python

# 12
my_tuple = (1, 2, 3, 4, 5)

# 13
print(my_tuple[1])

# 14
a, b, c, d, e = my_tuple

# 15
my_set = {1, 2, 3}

# 16
my_set.remove(2)

# 17
print(3 in my_set)

# 18
set1 = {1, 2, 3}
set2 = {3, 4, 5}
print(set1.union(set2))

# 19
print(set1.intersection(set2))

# 20
nested_dict = {"person": {"name": "John", "age": 30}}
print(nested_dict["person"]["name"])

```

---

4. Basic Operations

1. Add two numbers.

2. Subtract two numbers.

3. Multiply two numbers.
4. Divide two numbers and print the result as a float.
5. Perform integer division.
6. Find the remainder of a division.
7. Raise a number to a power.
8. Use the floor division operator.
9. Compare two numbers using ==.
10. Compare two numbers using !=.
11. Use logical AND on two boolean values.
12. Use logical OR on two boolean values.
13. Use a bitwise AND on two integers.
14. Use a bitwise OR on two integers.
15. Increment a variable using +=.
16. Decrement a variable using -=.
17. Use a conditional statement to compare two values.
18. Check if a number is divisible by 3 and 5.
19. Write a program to find the maximum of three numbers.
20. Write a program to find the factorial of a number using a loop.

Solution:

basic_operation.py

Python Programming > Week 01 > basic_operation.py > ...

```python
# 1 - 7
print(3 + 5, 10 - 2, 3 * 4, 10 / 3, 10 // 3, 10 % 3, 2 ** 3)

# 8
print(10 // 3)

# 9 - 10
print(5 == 5, 5 != 4)

# 11 - 12
print(True and False, True or False)

# 13 - 14
print(5 & 3, 5 | 3)

# 15 - 16
var = 5
var += 1
var -= 2

# 17
if 5 > 3:
    print("5 is greater")

# 18
if 15 % 3 == 0 and 15 % 5 == 0:
    print("Divisible by both")

# 19
print(max(10, 20, 15))

# 20
factorial = 1
num = 5
for i in range(1, num + 1):
    factorial *= i
print(factorial)
```

5. String Manipulations

1. Print the length of a string.
2. Concatenate two strings.
3. Repeat a string three times.
4. Access the first character of a string.
5. Access the last character of a string.
6. Slice a string to get the first three characters.
7. Convert a string to uppercase.
8. Convert a string to lowercase.
9. Check if a string starts with a specific letter.
10. Check if a string ends with a specific letter.
11. Find the index of a substring in a string.
12. Replace a word in a string with another word.
13. Split a string into a list of words.
14. Join a list of strings into a single string.
15. Remove whitespace from both ends of a string.
16. Check if a string is alphanumeric.
17. Reverse a string.
18. Count the occurrences of a character in a string.
19. Format a string using f-strings.
20. Write a program to check if a string is a palindrome.

Solution:

string_manipulations.py

Python Programming > Week 01 > string_manipulations.py > ...

```python
# 1 - 4
string = "hello"
print(len(string), string + " world", string * 3, string[0])

# 5 - 8
print(string[-1], string[:3], string.upper(), string.lower())

# 9 - 10
print(string.startswith("h"), string.endswith("o"))

# 11 - 12
print(string.find("e"), string.replace("hello", "hi"))

# 13 - 14
words = "Hello world".split()
print(" ".join(words))

# 15
print("  hello  ".strip())

# 16
print("hello123".isalnum())

# 17
print(string[::-1])

# 18
print(string.count("l"))

# 19
name, age = "John", 30
print(f"My name is {name}, and I am {age} years old.")

# 20
def is_palindrome(s):
    return s == s[::-1]
print(is_palindrome("madam"))
```

These exercises cover a broad spectrum of foundational Python concepts and encourage problem-solving skills. Let me know if you'd like any explanations or solutions for these exercises!