Data Types and Structures - Solutions

Exercise 1: Check Data Types

In [ ]:

```
a = 42
b = 3.14
c = "Python"
d = [1, 2, 3]
e = (1, 2, 3)
f = {'name': 'John', 'age': 30}
g = {1, 2, 3}

print(type(a))  # int
print(type(b))  # float
print(type(c))  # str
print(type(d))  # list
print(type(e))  # tuple
print(type(f))  # dict
print(type(g))  # set
```

Exercise 2: List Operations

In [ ]:

```
my_list = [10, 20, 30, 40, 50]
my_list.append(60)    # Adding 60
my_list.remove(30)     # Removing 30
my_list.reverse()      # Reversing the list

print(my_list)  # Output: [60, 50, 40, 20, 10]
```

Exercise 3: Tuple Unpacking

In [ ]:

```
my_tuple = (100, 200, 300)
a, b, c = my_tuple

print(a)  # 100
print(b)  # 200
print(c)  # 300
```

Exercise 4: Dictionary Manipulation

In [ ]:

```
person = {'name': 'Alice', 'age': 25, 'city': 'New York'}
person['age'] = 26  # Updating age
person['profession'] = 'Engineer'  # Adding new key-value pair

print(person)
# Output: {'name': 'Alice', 'age': 26, 'city': 'New York', 'profession': 'Engineer'}
```

Exercise 5: Set Operations

In [ ]:

```
set1 = {1, 2, 3, 4}
set2 = {3, 4, 5, 6}

union_set = set1.union(set2)
intersection_set = set1.intersection(set2)
```

```
difference_set = set1.difference(set2)

print("Union:", union_set)  # Output: Union: {1, 2, 3, 4, 5, 6}
print("Intersection:", intersection_set)  # Output: Intersection: {3, 4}
print("Difference:", difference_set)  # Output: Difference: {1, 2}
```

Exercise 6: String Slicing

In [ ]:

```
s = "Hello, Python!"

# Extract "Python"
substring = s[7:13]
print(substring)  # Output: Python

# Reverse the string
reversed_string = s[::-1]
print(reversed_string)  # Output: !nohtyP ,olleH
```

Exercise 7: Boolean Logic

In [ ]:

```
a = 10
b = 5
c = 5

# Check if 'a' is the largest and 'b' is not equal to 'c'
result = (a > b and a > c) and (b != c)
print(result)
# Output: False (because b == c)
```

Exercise 8: Complex Boolean Logic with Multiple Conditions

In [ ]:

```
x = 8
y = 6
z = 6

# Condition 1: x > y or y == z
condition1 = (x > y) or (y == z)

# Condition 2: The sum of x and y is even
condition2 = ((x + y) % 2 == 0)

# Condition 3: x, y, and z are all positive numbers
condition3 = (x > 0) and (y > 0) and (z > 0)

# Check if all conditions are true
result = condition1 and condition2 and condition3
print(result)
# Output: True
```

Exercise 9: Checking for Keys in a Dictionary

In [ ]:

```
person = {'name': 'Bob', 'age': 25}

if 'age' in person.keys():
    print("Key 'age' exists in the dictionary.")
# Output: Key 'age' exists in the dictionary.
```

Exercise 10: Nested Data Structures

In [ ]:

```
students = {
    'Alice': {'age': 24, 'grades': [88, 92, 85]},
    'Bob': {'age': 23, 'grades': [75, 80, 89]}
}

# Print Bob’s age
print(students['Bob']['age'])  # Output: 23

# Print Alice’s grades
print(students['Alice']['grades'])  # Output: [88, 92, 85]
```