Python Fundamentals Practice Exercise

Objective

Complete a series of tasks that demonstrate your understanding of basic Python concepts including:

- Variable assignment
- Data types
- Type conversion
- String manipulation
- List operations
- Tuple and dictionary usage
- Basic functions
- Basic exception handling

Instructions

Write Python code to solve each of the following challenges. You may use built-in functions, methods, and standard Python operations.

Task 1: Data Type Manipulation

Create variables for the following:

- A string representing your full name
- An integer representing your age
- A float representing your height in meters
- A boolean indicating whether you are a student

Convert these variables to different types and print the results. Demonstrate type conversion using the following functions:

- `str()`
- `int()`
- `float()`
- `bool()`

In [1]:
```
#A string representing my full name
full_name = "Keshen A/L Nareshchandra"

#An integer representing my age
age = 26

#A float representing my height in meters
height = 1.8
```

```
#A boolean indicating if I am a student
is_student = True
```

In [7]:
```
#Conversion to different types

str(full_name) #full_name cannot be changed to int, float or bool
```

Out[7]: 'Keshen A/L Nareshchandra'

In [8]:
```
str(age), int(age), float(age), #age cannot be changed to bool
```

Out[8]: ('26', 26, 26.0)

In [9]:
```
str(height), int(height), float(height), #height cannot be changed to bool
```

Out[9]: ('1.8', 1, 1.8)

In [10]:
```
str(is_student), bool(is_student) #is_student cannot be changed to int and float
```

Out[10]: ('True', True)

Task 2: String Operations

Given the following string: "Python Programming is Fun!" Perform the following operations:

1. Print the length of the string
2. Convert the string to uppercase
3. Convert the string to lowercase
4. Replace the word "Fun" with "Exciting"
5. Split the string into a list of words
6. Check if the string starts with "Python"
7. Check if the string ends with "Fun!"

In [12]:
```
string = "Python Programming is Fun!"
```

In [13]:
```
#1
print(len(string))
```

```
26
```

In [16]:
```
#2
string.upper()
```

Out[16]: 'PYTHON PROGRAMMING IS FUN!'

In [17]:
```
#3
string.lower()
```

Out[17]: 'python programming is fun!'

In [22]:
```
#4
string.replace("Fun", "Exciting")
```

Out[22]: 'Python Programming is Exciting!'

In [23]:
```
#5
string.strip().split(" ")
```

Out[23]: ['Python', 'Programming', 'is', 'Fun!']

In [24]:
```
#6
string.startswith("Python")
```

Out[24]: True

In [25]:
```
#7
string.endswith("Fun!")
```

Out[25]: True

Task 3: List Manipulation

Create a list of your favorite fruits:

1. Add three more fruits to the list
2. Remove the second fruit from the list
3. Find the index of a specific fruit
4. Create a new list by copying the original list
5. Reverse the list
6. Sort the list alphabetically

In [238…
```
fruits = ["apple", "banana", "orange"]
```

In [237…
```
#1 Method 1: Using Append
fruits.append("kiwi"),
fruits.append("pear"),
fruits.append("grape"),
fruits
```

Out[237… ['apple', 'banana', 'orange', 'kiwi', 'pear', 'grape']

In [239…
```
#1 Method 2: Using [start:stop:step]
fruits[3:3] = ["kiwi", "pear", "grape"]
fruits
```

Out[239… ['apple', 'banana', 'orange', 'kiwi', 'pear', 'grape']

In [119…
```
#2
del fruits[1],
fruits
```

Out[119… ['apple', 'orange', 'kiwi', 'pear', 'grape']

In [120…
```
#3
fruits.index("apple")
```

Out[120… 0

In [121…
```
#4
fruits_copy = fruits.copy()
fruits_copy
```

Out[121…
```
['apple', 'orange', 'kiwi', 'pear', 'grape']
```

In [122…
```
#5
fruits.sort(reverse=True),
fruits
```

Out[122…
```
['pear', 'orange', 'kiwi', 'grape', 'apple']
```

In [123…
```
#6
fruits.sort(),
fruits
```

Out[123…
```
['apple', 'grape', 'kiwi', 'orange', 'pear']
```

Task 4: Tuple and Dictionary Exploration

Create a tuple with 5 different integers Create a dictionary representing a person with the following keys:

- "name"
- "age"
- "city"
- "occupation"

Perform these operations:

1. Access individual elements from the tuple
2. Add a new key-value pair to the dictionary
3. Remove a key-value pair from the dictionary
4. Get all keys from the dictionary
5. Get all values from the dictionary

In [116…
```
numbers_tuple = (1, 2, 3, 4, 5)
numbers_tuple
```

Out[116…
```
(1, 2, 3, 4, 5)
```

In [133…
```
person_dict = {
    "name":"Keshen",
    "age":26,
    "city":"George Town",
    "occupation":"Student"
}
person_dict
```

Out[133…
```
{'name': 'Keshen', 'age': 26, 'city': 'George Town', 'occupation': 'Student'}
```

In [139…
```
#1
person_dict["name"], person_dict["age"], person_dict["city"], person_dict["occup
```

Out[139… ('Keshen', 26, 'George Town', 'Student')

In [142…
```
#2
person_dict["height (in m)"] = 1.8
person_dict
```

Out[142…
```
{'name': 'Keshen',
 'age': 26,
 'city': 'George Town',
 'occupation': 'Student',
 'height (in m)': 1.8}
```

In [143…
```
#3
del person_dict["age"]
person_dict
```

Out[143…
```
{'name': 'Keshen',
 'city': 'George Town',
 'occupation': 'Student',
 'height (in m)': 1.8}
```

In [146…
```
#4
person_dict.keys()
```

Out[146… dict_keys(['name', 'city', 'occupation', 'height (in m)'])

In [148…
```
#5
person_dict.values()
```

Out[148… dict_values(['Keshen', 'George Town', 'Student', 1.8])

Task 5: Function Creation

Write a function that does the following:

1. Takes two parameters (a string and a number)
2. Concatenates the string with the number
3. Returns the final concatenated result

In [215…
```
def function(string, number):
    #Converting number to a string for the concatenation as only strings can be
    return string + str(number)
```

In [216…
```
function("Hey", 10)
```

Out[216… 'Hey10'

Bonus Challenge: Combination Task

Create a function that:

- Takes a list of numbers as input
- Calculates the sum of the list
- Calculates the average of the list

- Returns both the sum and average as a tuple

Hints

- Refer to Python documentation for built-in methods
- Use `print()` to verify your results
- Don't hesitate to experiment with different approaches

In [242…
```python
#Method 1: For Loop
def numbers1(numbers_list):
    number_sum = 0
    number_average = 0
    for number in numbers_list:
        number_sum += number
        number_average = number_sum/len(numbers_list)
    #Returning two things with a commna = auto tuple
    return number_sum, number_average
```

In [243…
```python
#Method 2: Built-in Sum Function
def numbers2(numbers_list):
    #Returning two things with a commna = auto tuple
    return sum(numbers_list), sum(numbers_list)/len(numbers_list)
```

In [244…
```python
#Method 3: Using Built-In Sum Function and the Mean Function From Statistics Lib
import statistics

def numbers3(numbers_list):
    #Returning two things with a commna = auto tuple
    return sum(numbers_list), statistics.mean(numbers_list)
```

In [245…
```python
#Method 1: Answer
numbers1((1,2,3,4,5))
```

Out[245… (15, 3.0)

In [246…
```python
#Method 2: Answer
numbers2((1,2,3,4,5))
```

Out[246… (15, 3.0)

In [247…
```python
#Method 3: Answer
numbers3((1,2,3,4,5))
```

Out[247… (15, 3)