Exercise Solutions - Sessions 1.1 & 1.2

Please, Complete Python Code Solutions for All Exercises

SESSION 1.1: Introduction to Python & Git

This session includes 4 exercises covering Python basics and Git workflow.

Exercise 1: About Me Program (10 minutes)

Task: Create a program that introduces yourself using variables and print statements.

Solution: about_me.py

```
# Exercise 1: About Me Program
name = "Sarah Johnson"
age = 28
city = "Chicago"
favorite_language = "Python"
reason_for_ml = "I want to build AI systems that help people"

print("Hello! Let me introduce myself:")
print()
print(f"My name is {name}")
print(f"I am {age} years old")
print(f"I live in {city}")
print(f"My favorite programming language is {favorite_language}")
print(f"I want to learn ML because: {reason_for_ml}")
print()
print("Nice to meet you! 🎉")
```

Expected Output:

```
Hello! Let me introduce myself:

My name is Sarah Johnson
I am 28 years old
I live in Chicago
My favorite programming language is Python
I want to learn ML because: I want to build AI systems that help people

Nice to meet you! 🎉
```

Exercise 2: Simple Budget Calculator (10 minutes)

Task: Calculate total expenses and remaining budget.

Solution: budget_calculator.py

```
# Exercise 2: Budget Calculator
monthly_budget = 3000
```

```
rent = 1200
groceries = 400
transportation = 150
entertainment = 200
utilities = 180

total_expenses = rent + groceries + transportation + entertainment + utilities
remaining = monthly_budget - total_expenses
percent_spent = (total_expenses / monthly_budget) * 100

print("=== Monthly Budget Report ===")
print(f"\nTotal Budget: ${monthly_budget}\n")
print("Expenses Breakdown:")
print(f"  Rent:           ${rent}")
print(f"  Groceries:      ${groceries}")
print(f"  Transportation: ${transportation}")
print(f"  Entertainment:  ${entertainment}")
print(f"  Utilities:      ${utilities}")
print(f"  Total Expenses: ${total_expenses}\n")
print(f"Remaining Budget: ${remaining}")
print(f"You spent {percent_spent:.1f}% of your budget")
if remaining > 0:
    print(f"✓ Good job! You have ${remaining} left!")
```

Expected Output:

```
=== Monthly Budget Report ===

Total Budget: $3000

Expenses Breakdown:
  Rent:           $1200
  Groceries:      $400
  Transportation: $150
  Entertainment:  $200
  Utilities:      $180
  Total Expenses: $2130

Remaining Budget: $870
You spent 71.0% of your budget
✓ Good job! You have $870 left!
```

Exercise 3: ML Program Countdown (10 minutes)

Task: Calculate days until ML bootcamp starts.

Solution: ml_countdown.py

```
# Exercise 3: ML Program Countdown
course_name = "Machine Learning Bootcamp"
start_date = "March 1, 2026"
duration_weeks = 18
days_until_start = 45
```

```
weeks_until_start = days_until_start / 7
total_days = duration_weeks * 7

print("════════════════════════════════════════")
print(f"   {course_name}")
print("════════════════════════════════════════")
print(f"\n📅 Start Date: {start_date}")
print(f"⏱  Duration: {duration_weeks} weeks ({total_days} days)\n")
print("⏳ TIME UNTIL START:")
print(f"   {days_until_start} days")
print(f"   {weeks_until_start:.1f} weeks\n")
print("💪 You've got this! Start preparing now!")
```

Exercise 4: Complete Git Workflow (20 minutes)

Task: Initialize Git repository and save all exercise files.

Solution: Git Commands

```
# Create and initialize repository
mkdir python_week1
cd python_week1
git init
git status

# Save about_me.py
git add about_me.py
git commit -m "Add about me program"

# Save budget_calculator.py
git add budget_calculator.py
git commit -m "Add budget calculator"

# Save ml_countdown.py
git add ml_countdown.py
git commit -m "Add ML program countdown"

# Make a change and commit
git diff
git add about_me.py
git commit -m "Add comment to about me program"

# View history
git log --oneline
```

SESSION 1.2: Working with Text (Strings)

This session includes 4 exercises covering string manipulation.

Exercise 1: Name Formatter (8 minutes)

Solution: name_formatter.py

```
# Exercise 1: Name Formatter
name1 = 'john smith'
name2 = '  ALICE JOHNSON  '
name3 = 'bob   JONES'

formatted_name1 = name1.strip().title()
formatted_name2 = name2.strip().title()
formatted_name3 = name3.strip().title()

print("=== Name Formatter ===\n")
print("Formatted Names:")
print(f"1. {formatted_name1}")
print(f"2. {formatted_name2}")
print(f"3. {formatted_name3}")
```

Expected Output:

```
=== Name Formatter ===

Formatted Names:
1. John Smith
2. Alice Johnson
3. Bob Jones
```

Exercise 2: Email Validator (9 minutes)

Solution: email_validator.py

```
# Exercise 2: Email Validator
email1 = 'user@company.com'
email2 = 'invalid.email'
email3 = '  USER@EMAIL.COM  '

# Process email1
cleaned1 = email1.strip().lower()
if '@' in cleaned1 and '.' in cleaned1:
    print(f"{cleaned1} - Valid")
    at_pos = cleaned1.index('@')
    username = cleaned1[:at_pos]
    domain = cleaned1[at_pos+1:]
    print(f"  Username: {username}, Domain: {domain}")
else:
    print(f"{cleaned1} - Invalid")

# Process email2
cleaned2 = email2.strip().lower()
if '@' in cleaned2 and '.' in cleaned2:
    print(f"{cleaned2} - Valid")
else:
    print(f"{cleaned2} - Invalid")
```

```
# Process email3
cleaned3 = email3.strip().lower()
if '@' in cleaned3 and '.' in cleaned3:
    print(f"{cleaned3} - Valid")
    at_pos = cleaned3.index('@')
    username = cleaned3[:at_pos]
    domain = cleaned3[at_pos+1:]
    print(f"  Username: {username}, Domain: {domain}")
else:
    print(f"{cleaned3} - Invalid")
```

Expected Output:

```
user@company.com - Valid
  Username: user, Domain: company.com
invalid.email - Invalid
user@email.com - Valid
  Username: user, Domain: email.com
```

Exercise 3: Text Statistics Tool (8 minutes)

Solution: text_stats.py

```
# Exercise 3: Text Statistics Tool
text = 'I am learning Python and I love it!'

# Calculate statistics
total_chars_with_spaces = len(text)
text_no_spaces = text.replace(' ', '')
total_chars_no_spaces = len(text_no_spaces)
words = text.split()
word_count = len(words)
average_word_length = total_chars_no_spaces / word_count
uppercase_version = text.upper()
lowercase_version = text.lower()

# Display results
print("=" * 50)
print("TEXT STATISTICS ANALYZER")
print("=" * 50)
print(f"\nOriginal Text: {text}\n")
print("——— BASIC STATISTICS ———")
print(f"Total characters (with spaces):    {total_chars_with_spaces}")
print(f"Total characters (without spaces): {total_chars_no_spaces}")
print(f"Word count:                        {word_count}")
print(f"Average word length:               {average_word_length:.2f}\n")
print("——— TEXT TRANSFORMATIONS ———")
print(f"Uppercase: {uppercase_version}")
print(f"Lowercase: {lowercase_version}")
```

Expected Output:

```
==================================================
TEXT STATISTICS ANALYZER
==================================================
Original Text: I am learning Python and I love it!

─── BASIC STATISTICS ───
Total characters (with spaces):    36
Total characters (without spaces): 30
Word count:                        7
Average word length:               4.29

─── TEXT TRANSFORMATIONS ───
Uppercase: I AM LEARNING PYTHON AND I LOVE IT!
Lowercase: i am learning python and i love it!
```

Exercise 4: Git Workflow for Session 1.2 (20 minutes)

Solution: Git Commands

```
# Create repository
mkdir session_1_2_work
cd session_1_2_work
git init

# Save exercises
git add name_formatter.py
git commit -m "Add name formatter program"

git add email_validator.py
git commit -m "Add email validation tool"

git add text_stats.py
git commit -m "Add text statistics analyzer"

# Make a change
git diff
git add name_formatter.py
git commit -m "Add additional test name"

# View history
git log --oneline
```