

# Coding Standard Guidelines

Copyright (2025) Madeline and Laura, UC3M Students Software Development Group 16

Created on: 2025-02-05

Guided Exercise 1 Coding Standards

## TABLE OF CONTENTS:

Structured Directory Organization	3
Modified Rules	4
Basic	4
Design	5
Format	8
Imports	9
Logging	9
Messages control	9
String	10
Screenshots for Pylint	11

## Structured Directory Organization

G88.2024.TXX.GE1/	<i># Root directory of your project</i>
— uc3m_money/	<i># Python package</i>
— __init__.py	<i># Package initialization file</i>
— transaction_manager.py	<i># File for managing transactions</i>
— transaction_request.py	<i># File for handling requests</i>
— transaction_management_exception.py	<i># Custom exceptions</i>
— docs/	<i># Documentation</i>
— CodingStandard.pdf	<i># Our coding standard document</i>
— Screenshots.pdf	<i># File for all of the screenshots of Pylint</i>
— tests/	<i># Tests package</i>
— test.json	<i># Test file for valid IBAN</i>
— invalid_test.json	<i># Test file for invalid IBAN</i>
— main.py	<i># Main program entry point</i>
— README.md	<i># Project overview or documentation</i>
— .gitignore	<i># Git ignore rules</i>

## Modified Rules

### [BASIC]

- Attributes should follow camelCase.

Changed rule:

```
# Naming style matching correct attribute names.  
# MODIFIEDRULE  
# attr-naming-style=snake_cases  
attr-naming-style=camelCase
```

Accepted: *totalNum*

Rejected: *totalnum*

- The minimum length for lines of functions/classes which require docstrings cannot be shorter than 5.

Changed rule:

```
# Minimum line length for functions/classes that require docstrings, shorter  
# ones are exempt.  
# MODIFIEDRULE  
# docstring-min-length=-1  
docstring-min-length=5
```

Accepted: *def add(a, b):*

```
    """Adds two numbers and returns the result."""  
    return a + b
```

Rejected: *def f():*

```
    """A function that does something."""  
    pass
```

- Good variable names that should always be accepted are i, j and k.

Changed rule:

```
# Good variable names which should always be accepted, separated by a comma.  
# MODIFIEDRULE  
# good-names=i,  
#     j,  
#     k,  
#     ex,  
#     Run,  
#     -  
good-names=i,  
    j,  
    k
```

Accepted: *for i in range(10)*

*print(i)*

Rejected: *for \_ in range(10)*

*print("loop")*

## [DESIGN]

→ Max Branches for a function/method body should be less than or equal to 10.

Changed rule:

```
# Maximum number of branch for function / method body.  
# MODIFIEDRULE  
# max-branches=12  
max-branches=10
```

Accepted: *def is\_positive(n):*

*"""Returns True if the number is positive, otherwise False."""*

*if n > 0:*

*return True*

*return False*

Rejected: *def analyze\_number(n):*

*"""Analyzes a number based on multiple conditions."""*

*if n < 0:*

*return "Negative"*

*elif n == 0:*

*return "Zero"*

*if n % 2 == 0:*

*print("Even")*

*else:*

*print("Odd")*

*if n % 3 == 0:*

*print("Divisible by 3")*

*if n % 5 == 0:*

*print("Divisible by 5")*

```

if n % 7 == 0:
    print("Divisible by 7")
for i in range(3):
    if i % 2 == 0:
        print(f"{i} is even")
    else:
        print(f"{i} is odd")
if n > 100:
    print("Number is too large")
return "Done"

```

→ Max number of returns per function/method is less than or equal to 5 returns.

Changed rule:

```

# Maximum number of return / yield for function / method body.
# MODIFIEDRULE
# max-returns=6
max-returns=5

```

Accepted: `def easy_func(a: int):`

```

    """Prints num."""
    if a == 2:
        return True
    return False

```

Rejected: `def complex_function(n):`

```

    """Evaluates num."""
    if n < 0: return "Negative"
    if n == 0: return "Zero"
    if 1 <= n <= 10: return "Small"
    if 11 <= n <= 50: return "Medium"
    if 51 <= n <= 100: return "Large"
    if n > 100: return "Extra Large"

```

→ Methods should not exceed 60 lines of code excluding comments and blank lines.

Changed rule:

```
# Maximum number of statements in function / method body.
# MODIFIEDRULE
# max-statements=50
max-statements=60
```

Accepted: *def process\_data(data):*

```
    """Processes a list of numbers and returns a dictionary with total, count and
    average."""
    if not data:
        return {"error": "No data provided"}
    total = 0
    count = 0
    for num in data: # Iterate through data
        total += num
        count += 1
    average = total / count if count > 0 else 0
    return { "total": total, "count": count, "average": average }
```

Rejected: *def long\_function():*

```
    """Performs a complex operation on a dataset."""
    result = []
    # Loop through a large dataset
    for i in range(100):
        value = i * 2
        result.append(value)
    for i in range(100):
        value = i * 3
        result.append(value)
    for i in range(100):
        value = i * 4
        result.append(value)
    for i in range(100):
        value = i * 5
        result.append(value)
```

```

for i in range(100):
    value = i * 6
    result.append(value)
total = sum(result)          # Complex calculation done
count = len(result)
average = total / count if count > 0 else 0
return {"total": total, "count": count, "average": average}

```

## [FORMAT]

→ 2 spaces indentation should be used for all code; not tab.

Changed rule:

```

# String used as indentation unit. This is usually "    " (4 spaces) or "\t" (1
# tab).
# MODIFIEDRULE
# indent-string='    '
indent-string='    '

```

Accepted:

```

def hello():
    if True:
        print("Hello world!")

```

Rejected:

```

def hello():
    if True:
        print("Hello world!")

```

→ Maximum number of characters on a single line is less than or equal to 150 characters.

Changed rule:

```

# Maximum number of characters on a single line.
# MODIFIEDRULE
# max-line-length=100
max-line-length=150

```

Accepted: `def calculate_total(price, tax_rate):`

```

    total = price + (price * tax_rate)

```



*return total*

Rejected: *def calculate\_total(price, tax\_rate): return price + (price \* tax\_rate) + (price \* 0.05) + (price \* 0.1) + (price \* 0.2) + (price \* 0.3) + (price \* 0.4) + (price \* 0.5)*

## [IMPORTS]

→ “enchant” and “requests” should be treated as third-party libraries.

Changed rule:

```
# Force import order to recognize a module as part of a third party library.  
# MODIFIEDRULE  
# known-third-party=enchant  
known-third-party=enchant, requests
```

Accepted: *import requests*

*import enchant*

Rejected: *import enchant*

*import requests*

## [LOGGING]

→ When formatting a logging module, {}-formatting or str.format() should be used.

Changed rule:

```
# The type of string formatting that logging methods do. `old` means using %  
# formatting, `new` is for `{}` formatting.  
# MODIFIEDRULE  
# logging-format-style=old  
logging-format-style=new
```

Accepted: *logging.info("User {} logged in at {}".format(username, timestamp))*

Rejected: *logging.info("User %s logged in at %s", username, timestamp)*

## [MESSAGES CONTROL]

→ “type-check” should be enabled to verify type-correctness within the code; “type-check” detects issues such as invalid assignments, using objects incorrectly, etc.

Changed rule:

```
# Enable the message, report, category or checker with the given id(s). You can
# either give multiple identifier separated by comma (,) or put this option
# multiple time (only on the command line, not in the configuration file where
# it should appear only once). See also the "--disable" option for examples.
# MODIFIEDRULE
# enable=
enable=typecheck
```

Accepted: *enable=typecheck,unused-import*

Rejected: *enable=*

*enable=typecheck*

*enable=unused-import*

## [STRING]

→ “ ” type quotes and ' ' type quotes should not be mixed together.

Changed rule:

```
# This flag controls whether inconsistent-quotes generates a warning when the
# character used as a quote delimiter is used inconsistently within a module.
# MODIFIEDRULE
# check-quote-consistency=no
check-quote-consistency=yes
```

Accepted: *name = "Alice" greeting = "Hello, world!"*

Rejected: *name = "Alice" greeting = 'Hello, world!'*

# Screenshots for Pylint

★ Running Pylint after rules have been modified, without correcting the code.

```
***** Module UC3MONEY.TransactionManager
UC3MONEY.TransactionManager.py:6:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionManager.py:7:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:9:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionManager.py:12:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:14:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionManager.py:16:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:17:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:18:0: W0311: Bad indentation. Found 16 spaces, expected 8 (bad-indentation)
UC3MONEY.TransactionManager.py:19:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:20:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:21:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:22:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:25:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:26:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:27:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:28:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:29:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:30:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:31:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:32:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:33:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:34:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:35:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
UC3MONEY.TransactionManager.py:36:0: W0311: Bad indentation. Found 16 spaces, expected 8 (bad-indentation)
UC3MONEY.TransactionManager.py:37:0: C0304: Final newline missing (missing-final-newline)
UC3MONEY.TransactionManager.py:37:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionManager.py:1:0: C0114: Missing module docstring (missing-module-docstring)
UC3MONEY.TransactionManager.py:1:0: C0103: Module name "TransactionManager" doesn't conform to snake_case naming style (invalid-name)
```

```
***** Module UC3MONEY.TransactionRequest
UC3MONEY.TransactionRequest.py:6:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:7:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:8:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:9:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:10:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:11:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:13:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:14:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:16:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:17:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:18:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:19:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:20:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:21:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:23:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:24:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:25:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:26:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:27:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:28:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:30:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:31:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:32:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:33:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:34:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
UC3MONEY.TransactionRequest.py:35:0: C0304: Final newline missing (missing-final-newline)
UC3MONEY.TransactionRequest.py:35:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:1:0: C0114: Missing module docstring (missing-module-docstring)
UC3MONEY.TransactionRequest.py:1:0: C0103: Module name "TransactionRequest" doesn't conform to snake_case naming style (invalid-name)
```

```
UC3MONEY.TransactionRequest.py:35:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
UC3MONEY.TransactionRequest.py:1:0: C0114: Missing module docstring (missing-module-docstring)
UC3MONEY.TransactionRequest.py:1:0: C0103: Module name "TransactionRequest" doesn't conform to snake_case naming style (invalid-name)
UC3MONEY.TransactionRequest.py:5:0: C0115: Missing class docstring (missing-class-docstring)
UC3MONEY.TransactionRequest.py:7:8: C0103: Attribute name "__receptorName" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:8:8: C0103: Attribute name "__IBANFrom" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:9:8: C0103: Attribute name "__IBANTo" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:11:8: C0103: Attribute name "__timeStamp" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:6:23: C0103: Argument name "IBANFROM" doesn't conform to snake_case naming style (invalid-name)
UC3MONEY.TransactionRequest.py:6:32: C0103: Argument name "IBANTo" doesn't conform to snake_case naming style (invalid-name)
UC3MONEY.TransactionRequest.py:6:40: C0103: Argument name "RECEPTORName" doesn't conform to snake_case naming style (invalid-name)
UC3MONEY.TransactionRequest.py:17:4: C0103: Attribute name "receptorName" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:20:4: C0103: Attribute name "receptorName" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:24:4: C0103: Attribute name "IBAN_FROM" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:27:4: C0103: Attribute name "IBAN_FROM" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:31:4: C0103: Attribute name "IBAN_To" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:34:4: C0103: Attribute name "IBAN_To" doesn't conform to '^_[a-z][a-z0-9_]*$' pattern (invalid-name)
UC3MONEY.TransactionRequest.py:11:8: W0238: Unused private member 'TransactionRequest.__timeStamp' (unused-private-member)

-----
Your code has been rated at 0.00/10 (previous run: 5.76/10, -5.76)
```

Selected examples created with OpenAI (2024). *ChatGPT* (version 4o) [Large size language model].  
11 <https://chat.openai.com/chat>. Specified in the Student Statement of Generative AI.

```
(.venv) PS C:\Users\Owner\PycharmProjects\688.2024.TXX.GE1> pylint main.py
***** Module main
main.py:10:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:11:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:12:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
main.py:13:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
main.py:14:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
main.py:15:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
main.py:16:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
main.py:17:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:20:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:21:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:22:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
main.py:23:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
main.py:24:0: W0311: Bad indentation. Found 8 spaces, expected 4 (bad-indentation)
main.py:25:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
main.py:26:0: W0311: Bad indentation. Found 12 spaces, expected 6 (bad-indentation)
main.py:27:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:32:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:33:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:34:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:35:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:36:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:37:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:38:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:39:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:40:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:41:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)

main.py:39:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:40:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:41:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:44:0: W0311: Bad indentation. Found 4 spaces, expected 2 (bad-indentation)
main.py:12:0: W1405: Quote delimiter ' is inconsistent with the rest of the file (inconsistent-quotes)
main.py:13:0: W1405: Quote delimiter ' is inconsistent with the rest of the file (inconsistent-quotes)
main.py:22:0: W1405: Quote delimiter ' is inconsistent with the rest of the file (inconsistent-quotes)
main.py:23:0: W1405: Quote delimiter ' is inconsistent with the rest of the file (inconsistent-quotes)
main.py:1:0: C0114: Missing module docstring (missing-module-docstring)
main.py:5:0: C0103: Constant name "letters" doesn't conform to UPPER_CASE naming style (invalid-name)
main.py:6:0: C0103: Constant name "shift" doesn't conform to UPPER_CASE naming style (invalid-name)
main.py:9:0: C0103: Function name "Encode" doesn't conform to snake_case naming style (invalid-name)
main.py:19:0: C0103: Function name "Decode" doesn't conform to snake_case naming style (invalid-name)
main.py:30:0: C0116: Missing function or method docstring (missing-function-docstring)
main.py:34:4: C0103: Variable name "strRes" doesn't conform to snake_case naming style (invalid-name)
main.py:34:13: C2801: Unnecessarily calls dunder method __str__. Use str built-in function. (unnecessary-dunder-call)
main.py:36:4: C0103: Variable name "EncodeRes" doesn't conform to snake_case naming style (invalid-name)
main.py:38:4: C0103: Variable name "DecodeRes" doesn't conform to snake_case naming style (invalid-name)
main.py:2:0: C0411: standard import "string" should be placed before first party import "UC3MMoney.TransactionManager" (wrong-import-order)

-----
Your code has been rated at 0.00/10 (previous run: 0.00/10, +0.00)
```

★ Running Pylint after code has been corrected with the new implemented rules.

```
(.venv) PS C:\Users\Owner\PycharmProjects\688.2024.TXX.GE1> pylint main.py

-----
Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)

(.venv) PS C:\Users\Owner\PycharmProjects\688.2024.TXX.GE1>
```

```
(.venv) PS C:\Users\Owner\PycharmProjects\688.2024.TXX.GE1> pylint uc3m_money/

-----
Your code has been rated at 10.00/10 (previous run: 10.00/10, +0.00)
```