Project Name : DesignPatternGenerator

Goal : Create a Module Can auto Generate Some Langauge Code , include Design Pattern

Language : Python

File Structure:

DesignPatternGenerator

| Project.docx

| README.md

| DesignPatternGenerator

| \_\_init\_\_.py

| modules

| | \_\_init\_\_.py

| | basic.py

| | base.py

| | variable.py

| | operator.py

| | structure.py

| | function.py

| | class\_.py

| | designpattern.py

| | generator.py

| | generator\_variable.py

| | generator\_operator.py

| | generator\_structure.py

| | generator\_function.py

| | generator\_class.py

| | generactor\_designpattern.py

| tests

| | \_\_init\_\_.py

| | test\_basic.py

| | test\_base.py

| | test\_variable.py

| | test\_operator.py

| | test\_structure.py

| | test\_function.py

| | test\_class\_.py

| | test\_designpattern.py

| | test\_generator.py

| | test\_generator\_variable.py

| | test\_generator\_operator.py

| | test\_generator\_structure.py

| | test\_generator\_function.py

| | test\_generator\_class.py

| | test\_generactor\_designpattern.py

| example

| | \_\_init\_\_.py

| | createvariable.py

| | createstructure.py

| | createfucntion.py

| | createclass.py

| | createdesignpattern.py

| doc

TodoList

建立資料夾架構跟內容

讓所有基礎類別生成並讓通過測試

~~確認basic.py 建立完成，並通過測~~

~~確認 base.py 建立完成，並通過測~~

確認 variable.py 建立完成，並通過測

確認 operator.py 建立完成，並通過測

確認 structure.py 建立完成，並通過測

確認 function.py 建立完成，並通過測

確認 class\_.py 建立完成，並通過測

確認 generator.py 建立完成，並通過測

確認 generator \_variable.py 建立完成，並通過測

確認 generator \_operator.py 建立完成，並通過測

確認 generator \_structure.py 建立完成，並通過測

確認 generator \_function.py 建立完成，並通過測

確認 generator \_class\_.py 建立完成，並通過測

建立使用範例createvariable.py，並確認結果可用

建立使用範例createstructure.py，並確認結果可用

建立使用範例createfucntion.py，並確認結果可用

建立使用範例createclass.py，並確認結果可用

建立使用範例createdesignpattern.py，並確認結果可用

類別架構

系統資訊類別 basic 儲存要生成的語言基本內容

語法類別 base 定義各類型語法資訊

生成類別 generator 各種生成器

basic.py

class  Language

variable:

language = [‘Python’]

uselang = ‘’

method:

changeStringFormat

checkIsSupport

setUseLang

construct

setUserLang

class  Keyword

variable:

(static)

datatype = ['int', 'str', 'None', 'list', 'dict', 'set', 'tuple', 'other’]

set\_value\_operaotr ['=', '+=', '-=', '\*=', '/=‘]

compute\_operator = ['+', '-', '\*' '/' , '%','//', '\*\*’]

compare\_operaotor = ['==', '!=', '>', '>=', '<', '<=‘]

logic\_operator = ['and', 'or', 'not', 'is', 'is not','in’]

bitwise\_operator = ['&', '|', '~', 'a^b']

structure = ['if', 'elif', 'else’]

function = ['def’]

loop = [‘while’, for]

class\_ = ['class’]

design\_pattern = []

base.py

class Base

variable:

name

class GeneratorBase

variable

type

method

generate

variable.py

class Variable inherit  base.basic

variable

datatype

value

        class G\_Int inherit Variable

variable

datatype = int

        class G\_Str inherit Variable

variable

datatype = str

        class G\_List inherit Variable

variable

datatype = ;ist

        class G\_Dict inherit Variable

variable

datatype = dict

        class G\_Set inherit Variable

variable

datatype = set

        class G\_Tuple inherit Variable

variable

datatype = tuple

        class G\_Other inherit Variable

variable

datatype = other

class VariableFacotry

method

setVariable

setInt

setList

setDict

setSet

setTuple

setOther

doc

method is user to return Variable Object and manage Variable Object

operator.py

class Operator inherit base.Base

variable

str1

operator

str1

method

setOperator

class SetValueOperator inherit Operator

variable

method

checkOperatorIs SetValueOperator

class ComputeOperator inherit Operator

variable

method

checkOperatorIs ComputeOperator

class CampareOperator inherit Operator

variable

method

checkOperatorIsCompareOperator

class LoginOperator inherit Operator

variable

method

checkOperatorIs LoginOperator

class BitwiseOperator inherit Operator

variable

method

checkOperatorIs BitwiseOperator

structure.py

class Structure inhert base.Base

variable

conditionList

contentList

method

addCondition need CompareOperator or LoginOperator

addContent

class If inhert Structure

variable

name = if

method

class Elif inhert Structure

variable

name = elif

method

class Else inhert Structure

variable

name = else

method

addCondition need CompareOperator or LoginOperator

addContent

loop.py

class Loop inhert base.Base

variable

condition

contentList

method

addCondition need CompareOperator or LoginOperator

addContent

class While inhert Loop

variable

method

class For inhert Loop

variable

method

function.py

class Function inhert base.Base

variable

parameterList

contentList

method

addParameter need Variable

addContent

class\_.py

class Class inhert base.Base

variable

parameterList

methodList

method

addCondition need CompareOperator or LoginOperator

addContent

design pattern

class DesignPattern inhert base.Base

variable

type

classList

method

addClass need Class

base\_generator.py

class GeneratorBase

variable

type

method

generate

generatorvariable.py

class GeneratorVariable inherit GeneratorBase

variable

type = variable

method

generatorstructure.py

Generatorstructure

generator.py

class Generator

variable

obj\_generate

Method

SetGenerate

Generate

| **Operation** | **Syntax** | **Function** |
| --- | --- | --- |
| Addition | a + b | add(a, b) |
| Concatenation | seq1 + seq2 | concat(seq1, seq2) |
| Containment Test | obj in seq | contains(seq, obj) |
| Division | a / b | truediv(a, b) |
| Division | a // b | floordiv(a, b) |
| Bitwise And | a & b | and\_(a, b) |
| Bitwise Exclusive Or | a ^ b | xor(a, b) |
| Bitwise Inversion | ~ a | invert(a) |
| Bitwise Or | a | b | or\_(a, b) |
| Exponentiation | a \*\* b | pow(a, b) |
| Identity | a is b | is\_(a, b) |
| Identity | a is not b | is\_not(a, b) |
| Indexed Assignment | obj[k] = v | setitem(obj, k, v) |
| Indexed Deletion | del obj[k] | delitem(obj, k) |
| Indexing | obj[k] | getitem(obj, k) |
| Left Shift | a << b | lshift(a, b) |
| Modulo | a % b | mod(a, b) |
| Multiplication | a \* b | mul(a, b) |
| Matrix Multiplication | a @ b | matmul(a, b) |
| Negation (Arithmetic) | - a | neg(a) |
| Negation (Logical) | not a | not\_(a) |
| Positive | + a | pos(a) |
| Right Shift | a >> b | rshift(a, b) |
| Slice Assignment | seq[i:j] = values | setitem(seq, slice(i, j), values) |
| Slice Deletion | del seq[i:j] | delitem(seq, slice(i, j)) |
| Slicing | seq[i:j] | getitem(seq, slice(i, j)) |
| String Formatting | s % obj | mod(s, obj) |
| Subtraction | a - b | sub(a, b) |
| Truth Test | obj | truth(obj) |
| Ordering | a < b | lt(a, b) |
| Ordering | a <= b | le(a, b) |
| Equality | a == b | eq(a, b) |
| Difference | a != b | ne(a, b) |
| Ordering | a >= b | ge(a, b) |
| Ordering | a > b | gt(a, b) |