# Learn Python

3강 - 간단한 게임만들기

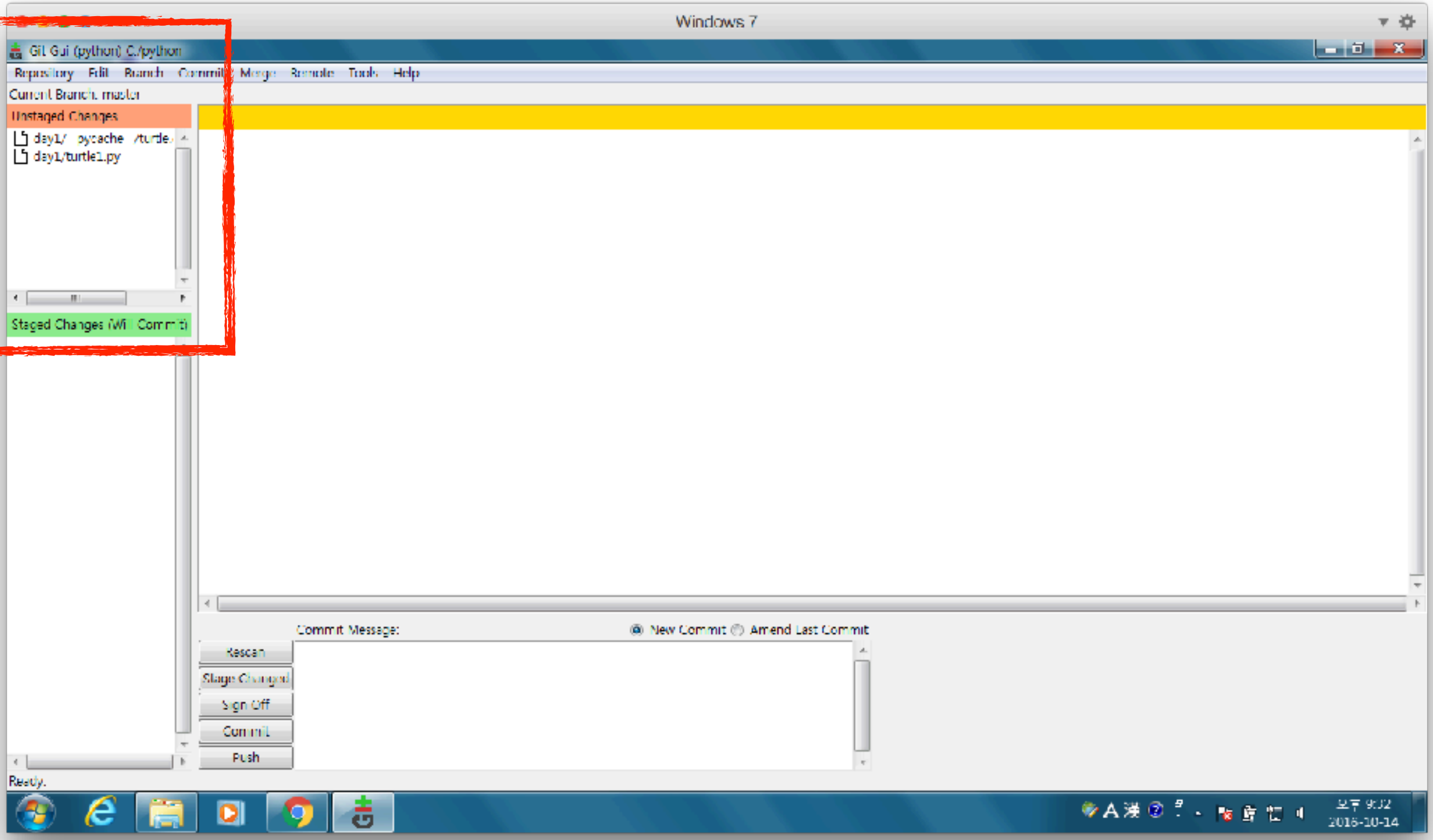
# 1. Git 간단 설명(1)

# 1.1 git clone

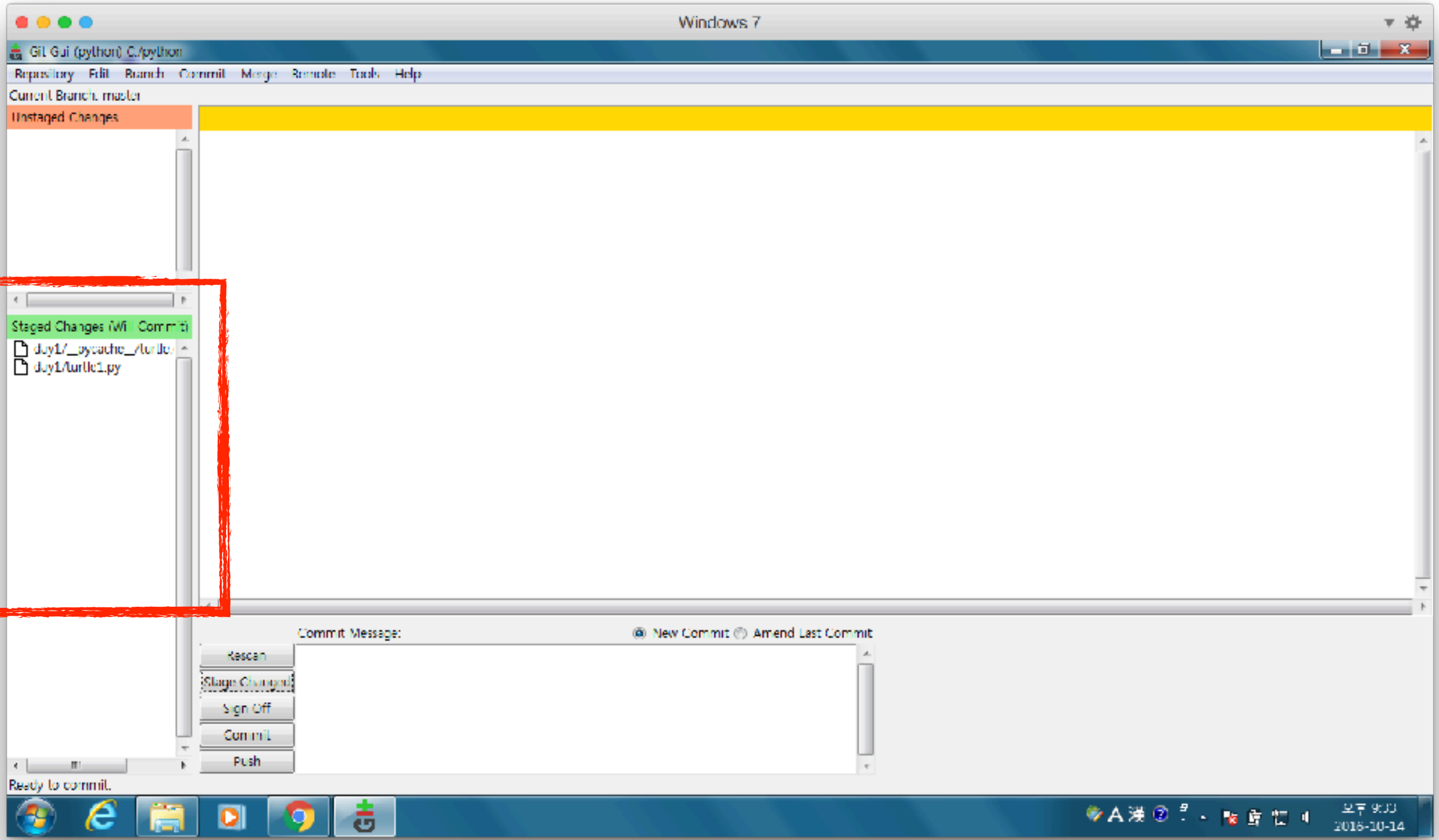
- 원격 저장소의 복제합니다.
- git clone /로컬/저장소/경로
- git clone 사용자명@호스트:/원격/저장소/경로

# 1.2 git 작업의 흐름

- 작업 디렉토리 - 인덱스 - HEAD
- 작업 디렉토리의 변경 내용은 unstaging area
- add를 해주면 인덱스 영역으로 staging area
- commit을 해주면 HEAD 영역
- 즉, 코드의 변경이 생시면 add를 한 뒤에 commit 한다.


Windows 7
Git Gui (python) C:/python
Repository Edit Branch Commit Merge Remote Tools Help
Current Branch: master
Unstaged Changes
day1/__pycache__/turtle.
day1/turtle1.py
Staged Changes (Wi Commit)
Commit Message:
New Commit
Amend Last Commit
Rescan
Stage Changed
Sign Off
Commit
Push
Ready.
2016-10-14

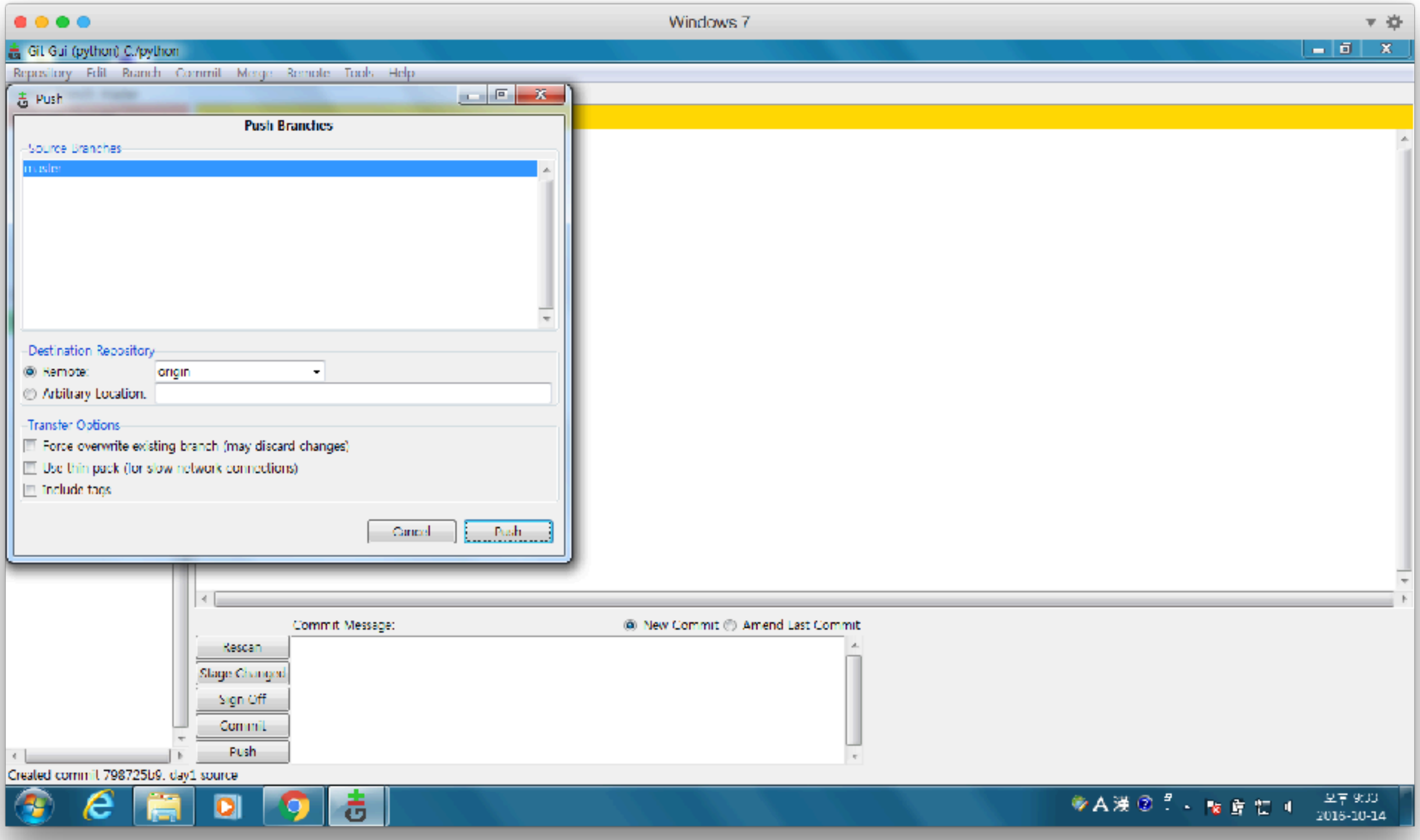
unstaged


Windows 7
Git Gui (python) C:/python
Repository Edit Branch Commit Merge Remote Tools Help
Current Branch: master
Unstaged Changes
Staged Changes (Wi Commit)
day1/__pycache__/turtle
day1/turtle1.py
Commit Message:
New Commit
Amend Last Commit
Rescan
Sign Off
Commit
Push
Ready to commit.
2016-10-14

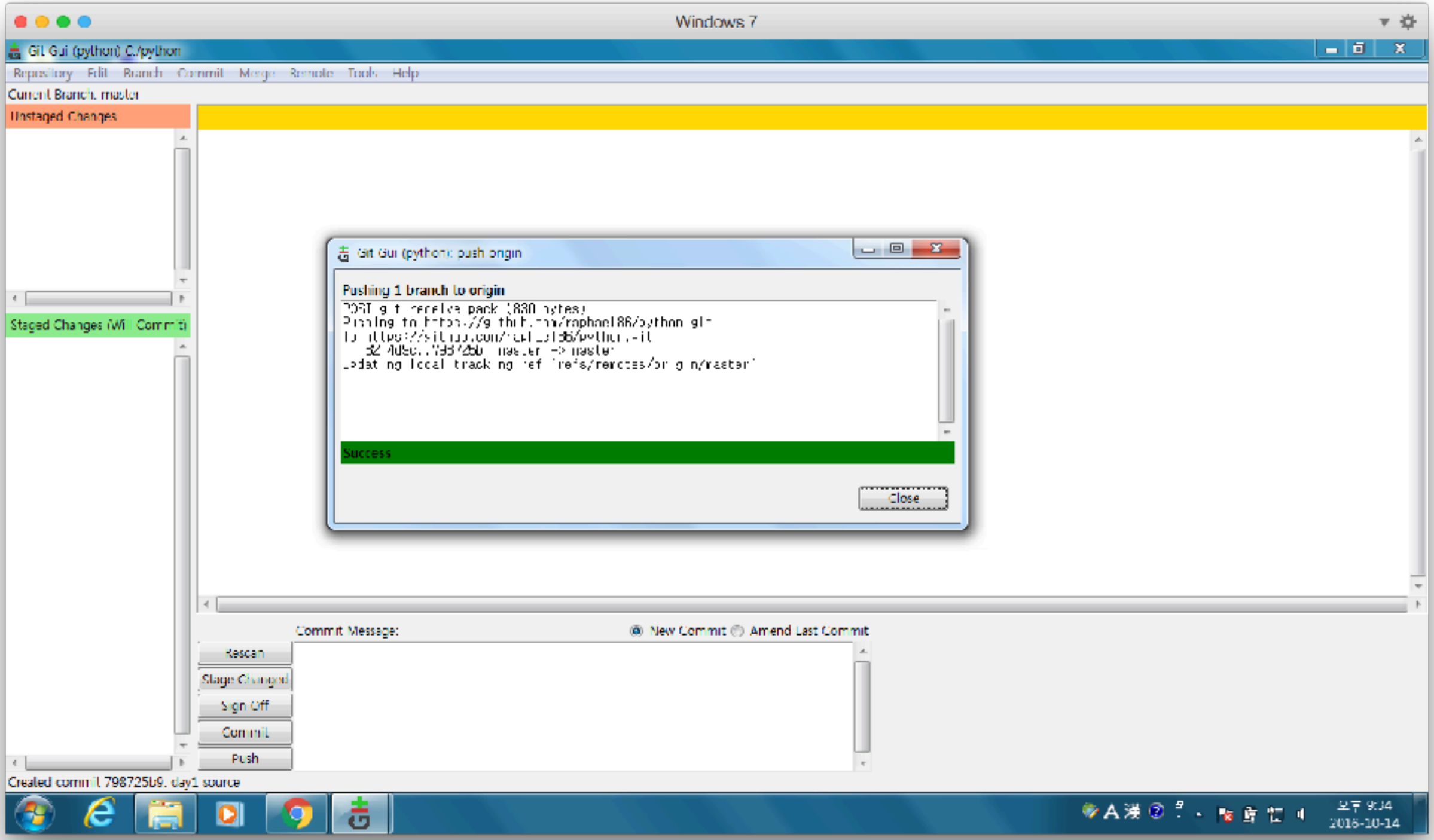
staged

# 1.3 git push

- commit한 내용을 원격 저장소로 올리는 작업
- git push (원격저장소) (브랜치)
- git push origin(깃헙저장소) master(기본 브랜치명)




origin master로 push 합니다.


Windows 7
Git Gui (python) C:/python
Repository Edit Branch Commit Merge Remote Tools Help
Current Branch: master
Unstaged Changes
Staged Changes (Wi Commit)
Git Gui (python): push origin
Pushing 1 branch to origin
Success
Close
Commit Message:
New Commit
Amend Last Commit
Rescan
Stage Changed
Sign Off
Commit
Push
Created commit 798725b9: day1 source

push 성공

# 1.4 command line 실습

- README.md 파일을 수정하면서 진행하기
- git clone
- git status
- git commit
- git push

# 2. 파이썬 클래스

# 2.1 클래스와 객체

- 객체란?
    - 데이터, 행위, 아이덴티티를 가지고 있는 것
    - 데이터는 객체의 상태를 기술하는 정보를 저장
    - 행위들은 메세지를 받았을 때 객체가 어떻개 해야하는지 알고 있는 것
    - 어떠한 객체를 다른 객체와 구분하는 것을 가능

# 2.1 클래스와 객체

- 클래스란?
    - 클래스란 객체를 생성하는 청사진
    - 클래스는 객체가 소유하게 될 속성 attributes 과 행위 behaviors 를 정의
    - 객체를 생성하기 위한 형판(틀)
    - 클래스 이름은 PEP 8 Coding Convention에 가이드된 대로 각 단어의 첫 문자를 대문자로 하는 CapWords 방식으로 명명
    - 클래스는 메서드(method), 속성(property), 클래스 변수 (class variable), 초기자(initializer), 소멸자(destructor) 등의 여러 맴버들을 가질 수 있음

# 2.1 클래스와 객체

- 클래스란?

```
class 클래스이름[(상속 클래스명)]:
    <클래스 변수 1>
    <클래스 변수 2>
    ...
    def 클래스함수1(self[, 인수1, 인수2,,,]):
        <수행할 문장 1>
        <수행할 문장 2>
        ...
    def 클래스함수2(self[, 인수1, 인수2,,,]):
        <수행할 문장1>
        <수행할 문장2>
        ...
...
```

# 2.1 클래스와 객체

- 클래스란?

```
class 클래스이름[(상속 클래스명)]:
    <클래스 변수 1>
    <클래스 변수 2>
    ...
    def 클래스함수1(self[, 인수1, 인수2,,,]):
        <수행할 문장 1>
        <수행할 문장 2>
        ...
    def 클래스함수2(self[, 인수1, 인수2,,,]):
        <수행할 문장1>
        <수행할 문장2>
        ...
...
```

# 2.1 클래스와 객체

- 사칙연산 클래스

```
class FourCal:
    def setdata(self, first, second):
        self.first = first
        self.second = second
    def sum(self):
    result = self.first + self.second
    return result
    def mul(self):
    result = self.first * self.second
    return result
    def sub(self):
    result = self.first - self.second
    return result
    def div(self):
    result = self.first / self.second
    return result
```

# 2.1 클래스 생성자와 소멸자

- 생성자 __init__()
    - 객체가 생성될 때 호출됨
- 소멸자 __del__()
    - 객체가 소멸될 때 호출됨
    - 레퍼런스 카운트가 0이 되는 순간 호출됨

# 2.2 클래스 변수

- 클래스 정의에서 메서드 밖에 존재하는 변수를 클래스 변수(class variable)라 하는데, 이는 해당 클래스를 사용하는 모두에게 공용으로 사용되는 변수

- "클래스명.변수명" (클래스 내외부에서) 혹은 "인스턴스변수명.변수명" (클래스 외부에서) 으로 엑세스 할 수 있다.

# 2.2 인스턴스 변수

- 클래스 정의에서 메서드 안에서 사용되면서 "self.변수명"처럼 사용되는 변수를 인스턴스 변수(instance variable)라 하는데, 이는 각 객체별로 서로 다른 값을 갖는 변수

- 인스턴스 변수는 클래스 내부에서는 self.width 과 같이 "self." 을 사용하여 엑세스하고, 클래스 밖에서는 "객체변수.인스턴스변수"와 같이 엑세스

# 2.2 클래스 변수

```
class Rectangle:
    count = 0  # 클래스 변수

    # 초기자(initializer)
    def _init_(self, width, height):
        # self.* : 인스턴스변수
        self.width = width
        self.height = height
        Rectangle.count += 1

# 메서드
def calcArea(self):
    area = self.width * self.height
    return area
```

# 2.3 클래스 메소드

- 메서드는 클래스의 행위를 표현하는 것으로 클래스 내의 함수
- 메서드는 해당 클래스와 관련된 어떤 행위를 표현하는 함수인데, 함수와 다르게 항상 첫번째 파라미터로 해당 클래스 객체를 나타내는 "self" 를 갖는다.
- 메서드는 여러 파라미터를 가질 수 있지만, 첫번째 파라미터는 항상 self 를 갖는다.
- public 메소드는 외부에서 실행이 가능한 함수이다.
- private 메소드는 내부에서만 실행이 가능한 함수이다.

# 2.3 클래스 상속

- 부모 클래스와 자식 클래스라는 것이 존재하여 부모 클래스의 멤버를 자식 클래스가 물려받을 수 있다.

- 클래스를 상속 받기 위해서는 파생클래스(자식클래스)에서 클래스명 뒤에 베이스클래스(부모클래스) 이름을 괄호와 함께 넣어 주면 된다.

- 파이썬은 다중 상속을 지원하지만, 수업에서는 Skip

```
#-*- coding: utf-8 -*-

class Rectangle:
    count = 0   # 클래스 변수

    # 초기자(initializer)
    def __init__(self, width, height):
        # self.* : 인스턴스변수
        self.width = width
        self.height = height
        Rectangle.count += 1

    # 메서드
    def calcArea(self):
        area = self.width * self.height
        return area

# public 메서드
def publicFunction(self):
    return "public function"

# public 메서드
def publicFunctionForPrivateFunction(self):
    return self.__privateFunction()

# private 메서드
def __privateFunction(self):
    return "private function"
```

```
#-*- coding: utf-8 -*-

class Animal:
    def __init__(self, name):
        self.name = name
    def move(self):
        print("move")
    def speak(self):
        print("mm..")

class Dog (Animal):
    def speak(self):
        print("bark")

class Duck (Animal):
    def speak(self):
        print("quack")
```