یکی از مهمترین دلایلی که لازم است تست بنویسیم این است که اگر بعد از اضافه کردن کدهای جدید تست هایی که از قبل وجود داشته است را یکبار اجرا کنیم و این تست ها به درستی اجرا شوند آنوقت مطمئن می‌شویم که کدهای جدید باعث باگ در کدهای قبلی نشده‌اند

------------------------------------------------------------------------------------------

در دایرکتوری unittest فایل‌های func.py و test\_func.py مربوط به این قسمت میباشد

# Unittest از standard library پایتون میباشد و لازم نیست به وسیله pip نصب شود

# یکی از بدی‌های unittest این است که دارای متدهای خیلی زیادی میباشد

# برای اجرای تست ها توسط unittest میبایست از دو دستور زیر استفاده کرد

python -m unittest file\_name.py ====> selected file

python -m unittest discover ====> all file that start with test\_

# توجه شود که در فانکشنهایی که برای تست نوشته می‌شود حتماً باید با test شروع شود

--------------------------------------------------------------------------------------

در دایرکتوری unittest فایل‌های person.py و test\_person.py مربوط به این قسمت میباشد

# با استفاده از fixture ها میتوانیم قبل و بعد از اجرا شدن تست ها کارهایی خاصی مثل ساخت آبجکت را برنامه‌ریزی کنیم

# setUp برای قبل و tearDown برای بعد از اجرا شدن تست میباشد

-------------------------------------------------------------------------------------

# برای اجرای تست ها به وسیله nose میبایست از دستورهای زیر استفاده کنیم

nosetests ======> all file

nosetests file\_name.py ======> selected file

# با استفاده از دو دستور بالا میتوانیم تست هایی که براساس unittest نوشته‌ایم رو هم تست کنیم

------------------------------------------------------------------------------------

# pytest همانند nose با unittest پایتون سازگار است و میتواند تست هایی که براساس unittest نوشته شده است را اجرا کند

# برای اجرای تست ها به وسیله pytest میبایست از دستورهای زیر استفاده کرد

pytest ====> all file

pytest file\_name.py =====> selected file

-----------------------------------------------------------------------------------

# fixture در pytest با unittest کمی متفاوت است به این صورت که فانکشن با استفاده از دکوریتور به fixture تبدیل می‌شود و میتوانیم این function را هم به عنوان setup و هم به عنوان teardown استفاده کنیم

# دستوراتی که بعد از yield نوشته می‌شود به عنوان teardown عمل میکند ( جلوی دستور yield باید اسم خوده فانکشن fixture نوشته شود )

# برای اینکه فانکشن fixture به عنوان setup برای فانکشنی عمل کند کافیست آن را به عنوان ورودی به فانکشن مورد نظر بدهیم

# با استفاده از آپشن s – میتوانیم پرینت هایی در تست استفاده کرده‌ایم را در کنسول نمایش دهیم

# میتوانیم خروجی تست ها را در فایلی ذخیره کنیم :pytest test\_func.py - -resultlog==file\_name.log

---------------------------------------------------------------------------------------------