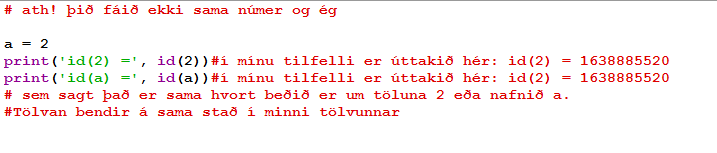
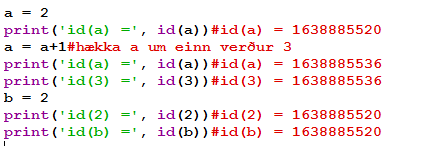
**Hlutir(objects) og nafnasöfn(namespace)**

Allt í python er hlutur(object). Til að geta unnið með hlut þarf að gefa hlutnum nafn. Þannig er hægt að nálgast hann með því að kalla á nafnið. Skoðum dæmi. Ef við skrifum kóða þar sem við látum a =2. Þá er 2 hlutur sem er geymdur í minn tölvunnar og leiðin til að nálgast hann er í gegnum nafnið á hlutnum sem í þessu tilfelli er a. Hægt er að skoða þetta með því að nota fallið id().



Skoðum þetta aðeins nánar. Segjum svo að við flækjum þetta aðeins bætum einum við breytuna a og gefum tveimur heitið b.



Hér er hægt að sjá hvað python er öflugt forritunarmál. Í byrjun er a=2 en svo bætum við einum við og þá er búinn til nýr hlutur 3 sem tengist þá breytunni a. a tengist ekki lengur 2. Næst tengjum við 2 við breytuna b. Nú sjáum við að b er komið með sama númer og a var með. Þetta sýnir að python þarf ekki að búa til nýjan hlut í hvert skipti sem nafninu er breytt á hlutnum. Dæmi



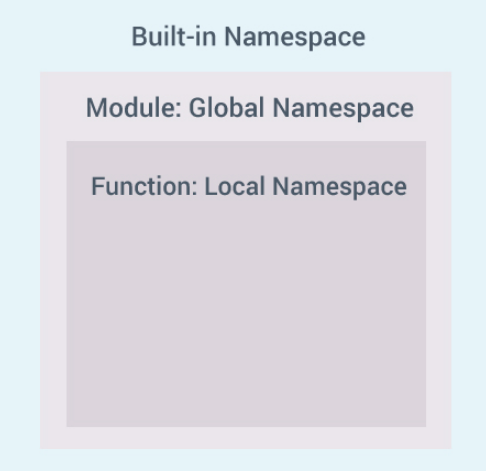
Það að hægt sé að láta breytu tengjast margskonar hlutum(strengjum,heiltölum ,listum , föllum) án þess að þurfa að skilgreina það frekar. Þetta gerir python að mjög sveigjanlegu forritunarmáli.

Nafnasöfn(namespace)

Nafnasöfn(namespace) er safn nafna. Í python má líta á nafnasafn sem skráningu á öllum hlutum(objects). Nafnasafn heldur utan um hluti sem hafa verið skilgreindir og sem hafa eitthvað sameiginlegt. Hægt er að nota mismunandi namespace á sama tíma en þau eru aðskilin frá hvert öðru. Nafnasafn sem heldur utan um öll innbyggð föll í python er búið til í hvert skipti sem python túlkurinn er keyrður. Þess vegna getum við alltaf nálgast innbyggð föll eins og id() print(). Hvert safn(module)býr til sitt eigið nafnasafn(namespace). Hvert nafnasafn er einangrað frá öðrum nafnasafni til að koma í veg fyrir að sömu nöfn í sitthvoru nafnasafninu ruglist ekki.

**Söfn(moduls)**

Hvert safn getur haft mörg föll og mismunandi klasa. Í hvert sinn sem kallað er á fall er búið til local nafnasafn sem innheldur öll nöfn sem eru skilgreind í fallinu. Sama gerist þegar kallað er í klasa. Skoðum mynd:



Þessi mynd sýnir hvernig nafnasöfnin er mynduð. Það sem þarf að hafa í huga þegar mismunandi nafnasöfn eru mynduð að ekki er hægt að nálgast hvaða breytu sem er hvar sem er í forritinu. Þetta fyrirbrigði í forritun er kölluð hulun(scope). Skoðum þetta nánar.

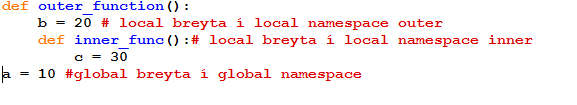
**Hulun(Scope)**

Eins og áður segir þá er ekki hægt að nálgast alla hluti nafnasafns hvar sem er í forritinu. Þar kemur fram hugtakið hulun(scope). Hulun þýðir að ef þú er staddur á ákveðnum stað í forriti þá geta sumir hlutar forritsins verið huldir (sjást ekki)á þessum tiltekna stað. Í forritum er alltaf til minnst þrjú hreiðruð scope.

* Scope fyrir fallið sem er með sínar eigin breytur(local)
* Scope fyrir safnið(module) sjálft sem hefur klasabreytur(global)
* Scope sem inniheldur innbyggð föll. Allsstaðar aðgengilegt

Ef nota á breytu í falli er fyrst athugað hvort viðkomnadi breyta er í fallinu(local scope) ef ekki þá er athugað hvort viðkomandi breyta sé í næsta scopi (global) og síðast í scopi sem hefur innbyggðu föllin. Ef það ef fall inn í öðru falli þá er búið til nýtt namespace(nýtt scope) fyrir það fall.

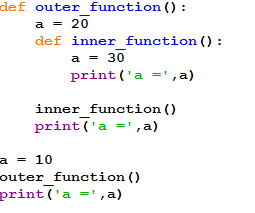
**Dæmi um hulun(scope)og nafnasöfn**



Hér er a breyta í global namspace. Breytan b er í local namespace fyrir fallið outer\_function og c er breyta inn í fallinu inner\_func sem er hreiðruð inn í falið outer\_function(er í local namespace fyrir inner\_func).

Þegar við erum stödd inn í inner\_func fallinu er c breytan local, b breytan nonlocal og a breytan global. Þegar við erum stödd í inner\_func getum við aðeins breytt gildinu á c breytunni en ekki á b eða c. Ef við reynum að breyta gildinu á a verður til ný local breyta sem heitir það sama og t.d global breytan en er allt annar hlutur.

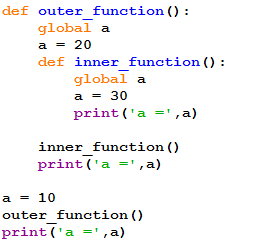
Skoðum dæmi:



Úttak :



Hér sést greinilega að breyting gerð á local a hefur ekki áhrif á global a. Oftur á móti ef við viljum að breyting gerð á local breytu a yfirfarist á global breytuna þá getum við skilgreint local a sem global a með því að hafa global fyrir framan breytuna. Sjá dæmi:



Hér setjum við global fyrir fram breytuna a. Við það breytist global breytan a varanlega, þvi tilvísunin(reference) er beint að global breytunni. Úttakið verður því:

