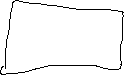
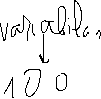
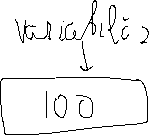
In Java, variabilele pot fi de 2 tipuri:

* De date primitive
* De referinta

**Variabilele de date primitive** se comporta exact ca in C++. Ele ocupa o zona din memorie in care pastreaza direct propria valoare. Orice modificare asupra unei astfel de variabile face ca valoarea ei veche sa fie inlocuita pur si simplu cu una noua. O egalare a doua variabile de date primitive doar face ca valaorea unei sa fie copiata de alta.Chiar si daca mai multe variabile de date primitive stocheaza aceeasi valoare, adresele ce le au nu vor fi aceleasi ca in Python!De ex:

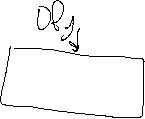
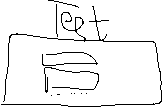
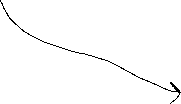
int variabila1 = 100;

in**t** variabila2 = 100;



**Variabilele de referinte** deja se comporta diferit si fata de C++(cu unel exceptii) si dfierit fata de Python. In Python, nici nu exista variabile, ci doar nume ce sunt legate de un obiect. In C++, doar pointerii pot stoca adrese de memorie, si variabilele ce sunt declarate din start sa stocheze referinte pot stoca referinte. In java, variabilele ce fac trimitire la un obiect mereu pastreaza doar referinta lui! Nu sunt legate de obiect ca in Python si nu pastreaza locatia din memorie, ci referinta.

Test Obj1 = new Test();

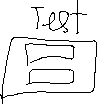
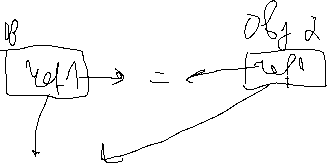


Deci, utilizarea variabilei Obj1 nu va face direct trimitirele la zona de memorie in care e stocat obiectul, ci la o referinta la ea! Anume constructorul e cel care trimite o referinta inapoi catre variabila.

Daca egalam doua variabile de referinta, primul obiect pur si simplu va copia referinta pe care o pastreaza a doua:

Test obj1 = new Test();

Test obj2 = obj1;



Deci, se respecta principiul copierii ca si la variabilele de date primitive! De asta, in Java variabilele la fel copie ce stocheaza, daca se egaleaza, ca in C++, doar ca diferenta e ca variabilele de obiecte pastreaza referinte la obiecte, nu propriu zis obiectul.