**Constuctori**

* Secventa Nume obj = new Nume(par1,par2..) face referire la crearea unui obiect si la apelarea constructorului direct la crearea sa.
* Daca nu exista nici-un constructor,Compilatorul creaza un constructor default default(de 2 ori default) automat care va initiliza cu valori default toate atributele neinitializate. Daca noi cream un constructor, nu se mai creaza un constructor default default, si deja noul constructor, desi nu setam sa faca asta, va avea grija sa initializeze cu valori default atributele neinitializate la finalul executiei sale. Daca nu avem noi definit un constructor fara parametri,ci doar cu parametri, Java oricum nu creaza constructorul default default, pentru a nu ne strica intentia de a nu permite construirea unor obiecte cu constructor default.
* Daca avem mai multi constructori, putem apela un constructor din alt constructor prin this(par1,par2…) si se va apela constructorul ce are acel nr de parametri. Apelarea unui alt constructor trebuie sa fie mereu prima instructiune din constructorul ce apeleaza altul si poate fi doar una!!!

**class** abc{

**public** **int** a;

**public** **int** b;

**public** abc(**int** x,**int** y) {

a = x;

b = y;

}

**public** abc() {

***this(10,20);***

a = 0;

b = 0;

}

}

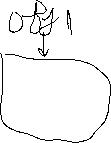
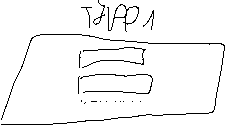
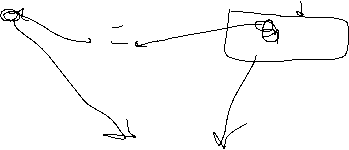
* Spre deosebire de alte limbaje, Java nu ofera un constructor default de copiere, ci trebuie creat unul.

**Ciclul unui obiect**

* Un obiect este sters de catre Java atunci cand nu mai exista nici-o referinta la el.
* Metodele, chiar daca au si obiecte definite in ele, odata ce termina executia, fac ca membrii lor sa piarda referintele si astfel obiectele si datele primitive sunt sterse din metoda si din memorie.
* **Garbage Collection** – proces de recuperare a memoriei care este ocupata de obiecte ce nu mai sunt referentiate de alte variabile.
* Garbage Collection scaneaza regular pentru a verifica daca nu trebuie sterse anumite obiecte.
* Type1 obj1 = new Type1();

Type1 obj2 = obj1;

Aici obj1 si obj2 au referinta la unul si acelasi obiect! Obj2 va crea o referinta identica cu cea pe care o are obj1, si obj1 are referinta la obiect. Obj2 va copia valoarea, adica referinta lui obj1, asa cum fac si datele primitive cand isi copie valorile, doar ca aici se copie referinta pe care o stocheaza obj1.



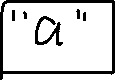
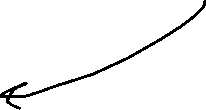
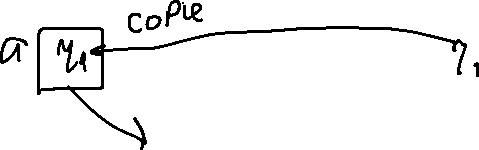
**Referintele variabilelor**

* variabila stocheaza doar referinta la un obiect. Daca o variabila va fi egala cu alta, acea variabila **va copia** referinta stocata de cealalta, si atat, **nu va crea o referinta la variabila data**. In Java totul se copie, inclusiv si referintele.
* String a = “a”;

String b = a;

a = “altceva”;

Aici, desi **a** si-a modificat referinta, b continua sa aiba referinta pe care a avut-o **a** cand b a copiat-o de la ea, de aceea b inca are referinta la obiectul “a”;



Dar mai apoi:

