**Айнымалылар**

Java бағдарламалау тілінде төмендегідей айнымалалар түрлері болады:

* **Дана айнымалылар(Статикалық емес)( Instance Variables(Non static fields))**

Техникалық көзқараспен айтқанда, объектер статикалық емес айнымалаларда, яғни static cөзінсіз жарияланған айнымалылар, өздеріне тән жеке мәндерді сақтайды. Статикалық емес айнымалылар дана айнымалылар деп те аталады, себебі олардың мәндері әрбір класс данасына дара мәнге ие, басқалай әрбір объект десе де болады. Мысалы, бір велсипедтің жылдамдығы currentSpeed келесі бір велосипедтің жылдамдығына тәуелді емес.

* **Класс айнымалалары(Class Variables(static fields))**

Класс айнымалылары кез келген static модификаторымен жарияланған айнымалы. Бұл компиляторға кластың қанша данасы бар екендігіне қарамастан, мұнда айнымалының тек бір ғана көшірмесі болатындығын білдіреді. Мысалы, анықталған велосипед түрінің жылдамдықтарын ауыстыру тісшелері саны айнымалысын static деп анықталса болады, себебі кластың барлық данасында бірдей жылдамдық ауыстыру тісшелері болады. Код static int numGears = 6; статикалық айнымалыны құрады. Қосымша ретінде, тісшелер саны ешқашан өзгермейтіндей көрсету үшін final модификаторын қосуға да болады.

* **Жергілікті айнымалылары(Local Variables)**

Объект қалай өзінің күйін атауларда сақтайтындығына ұқсас. Әдістер көбінесе өзінің уақытша күйін жергілікті айнымалыларда сақтайды. Жергілікті айнымалыларды жариялай синтаксисі жай атауды жариялаумен бірдей. Мысалы, int count = 0; Мұнда айнымалыны жергілікті етіп көрсететін арнайы модификатор қолданылмайды. Айнымалы жарияланған жердің бәрінде, яғни әдістің жақшасы ашылып жабылған аралықта, анықтала алады. Сондықтан жергілікті айнымалылар тек қана жарияланған әдіс ішінде ғана белгілі, ал кластың басқа бөліктерінде белгісіз, анықталмаған болып табылады.

* **Параметрлер(Parameters)**

Параметр – әдіске, болмасы класс құраушысына берілетін айнымалы. Параметрлерге байланысты бастапқы мысалдардан байқауға болады. Басты әдіс public static void main(String[] args) екендігін еске түсірейік. Мұнда args айнымалысы main әдісінің параметрі болып тұр.

Әрбір бағдарламалау тілі өзіне тән айнымалыларды қалай қолдану ережелері мен келісімдерге ие. Айнымалыларды атау туралы ережелерді алдыңғы түсіндірмеде таныстырғанбыз.

**Қарапайым деректер түрлері(Primitive data types)**

Java бағдарламалау тілі статикалық терілген, яғни барлық айнымалылар қолданылмастан бұрын жариялануы тиіс.

int gear = 1;

Жоғарыда көріп тұрғаныңыздай айнымалыны жариялау айнымалы түрінен және атауынан тұрады. Бұл дегеніміз бағдарламада gear атауындағы айнымалы бар, санды мәндерді сақтайтындығын және бастапқы мәні 1-ге тең екендігін білдіреді. Айнымалы түрі қандай мәнді сақтайтындығын және оған қандай мүмкін операцияларды орындауға болатындығын көрсетеді. Сонымен қатар int айнымалы түрінен басқа Java бағдарламалау тілі жеті қарапайым деректер түрін қолдайды. Қарапайым деректер түрлері алдын ала айқындалған және өзіне тән жеке атаулармен аталған. Қарапайым деректер түрлері басқаларымен күйін бөлісе алмайды.

Java бағдарламалау тілі қолдайтын сегіз түрлі қарапайым деректер түрлері:

* **byte:** деректер түрі 8-битті екі таңбалы бүтін сандар. Ең кіші мәні -128 және ең үлкен мәні 127 тең. Байтты деректер түрі үлкен деректерді сақтауда жадты үнемдеу үшін пайдалы болуы мүмкін. Одан басқа int орнына да қолдануға болады.
* **short:** деректер түрі 16-битті екі таңбалы бүтін сандар. Ең кіші мәні -32 768 және ең үлкен мәні 32 767 тең. Байтты деректер түрі сияқты нұсқаулықтар қолданыла алады.
* **int:** әдепкідей, int деректер түрі 32-битті екі таңбалы, ең кіші мәні -231 және ең үлкен мәні 231-1 тең бүтін сандар. Java SE 8 және одан кейінгі нұсқаларында int деректер түрін 32-битті таңбасыз, ең кіші мәні 0 және ең үлкен мәні 232-1 тең бүтін сандарды көрсету үшін де қолдануға болады. Көбірек ақпарат алу үшін Сандар класы бөлімін қараңыз. Integer класына таңбасыз бүтін сандарға қолданатын арифметикалық амалдарды қолдайтын compareUnsigned, divideUnsigned сияқты static әдістер қосылған-ды.
* **long:** деректер түрі 64-битті екі таңбалы бүтін сандар. Ең кіші мәні -263 және ең үлкен мәні 263-1 тең. Java SE 8 және одан кейінгі нұсқаларында long деректер түрін 64-битті таңбасыз, ең кіші мәні 0 және ең үлкен мәні 264-1 тең бүтін сандарды көрсету үшін де қолдануға болады. Int көзделгеннен кеңірек мәндер ауқымын қажет еткен жағдайда осы деректер түрін пайдаланыңыз.
* **float:** деректер түрі 32-битті IEEE 754 бөлшек сандар. Оның мәндер ауқымы айтылған шекаралардан асып өтеді, бірақ Java Language Specification құжатының Floating-Point Types, Formats, және Values бөлімдерінде нақтыланған. Бөлшек сандарды сақтау үшін қолданылады. Бұл деректер түрі ешқашан ақша бірлігі сияқты нақты сандар үшін қолданылмауы тиіс. Оның орнына сізге java.math.BigDecimal класын қолдану керек. BigDecimal туралы сандар бөлімінде қарастырылған.
* **double:** деректер түрі 64-битті IEEE 754 бөлшек сандар. Оның мәндер ауқымы айтылған шекаралардан асып өтеді, бірақ Java Language Specification құжатының Floating-Point Types, Formats, және Values бөлімдерінде нақтыланған. Ондық жүйедегі мәндер үшін бұл деректер түрі әдепкі таңдау болып табылады. Жоғарыда айтылғандай, бұл деректер түрі ешқашан нақты сандар үшін қолданылмауы тиіс.
* **boolean:** деректер түрі тек екі ықтимал мәндерге ие: true және false. Бұл деректер түрін тек қарапайым true/false(рас/өтірік) шарттарын қадағалау жағдайларында қолданыңыз. Бұл деректер түрі көлемі тек 1 бит.
* **char:** деректер түрі тек бір ғана 16-битті Unicode таңбасы болып табылады. Кіші мәні '\u0000' (немесе 0) және үлкен мәні '\uffff' (немесе 65,535) тең.

Жоғарыда айтылған сегіз қарапайым деректер түрінен басқа, Java бағдарламалау тілі таңбалар тізіміне(character strings) java.lang.String класы арқылы арнайы қолдау көрсетеді. Таңбалар тізімін екі жақты тырнақшаға алу арқылы автоматты түрде жаңа String объекті құрылады. Мысалы:

String s = "this is a string";

String объектері өзгертілмейді, яғни құрылғаннан кейін мәні өзгере алмайды. String класы туралы көбірек ақпарат Simple Data Objects бөлімінен ала аласыз.

**Әдеттегі мәндер**

Айнымалы жарияланған кезде бастапқы мәнді беру барлық жағдайда міндетті емес. Ондай айнымалылар компилятор көмегімен әдеттегі мәнге ие болады. Деректер түріне тәуелді бұл мәндер 0 немесе null бола алады. Әдетте бұндай мәндерге арқа арту нашар бағдарламалау стилі болып есептеледі.

Келесі кестеде деректер түрінің әдеттегі мәндері көрсетілген:

|  |  |
| --- | --- |
| **Деректер түрі** | **Әдеттегі мәндер** |
| byte | 0 |
| short | 0 |
| int | 0 |
| long | 0L |
| float | 0.0f |
| double | 0.0d |
| char | ‘\u0000’ |
| String (немесе кез-келген object) | null |
| boolean | false |

Жергілікті айнымалылар біраз ерекшеленеді. Компилятор инициализацияланбаған жергілікті айнымалыларға ешқашан әдеттегі мәнді бермейді. Егер сіз жергілікті айнымалыңыз жарияланғанда инициализация жасай алмасаңыз, онда айнымалыны қолданбас алдын оған мән берілгенін қадағалаңыз. Инициализацияланбаған жергілікті айнымалыны қолдануға тырысу compile-time error қателігіне алып келеді.