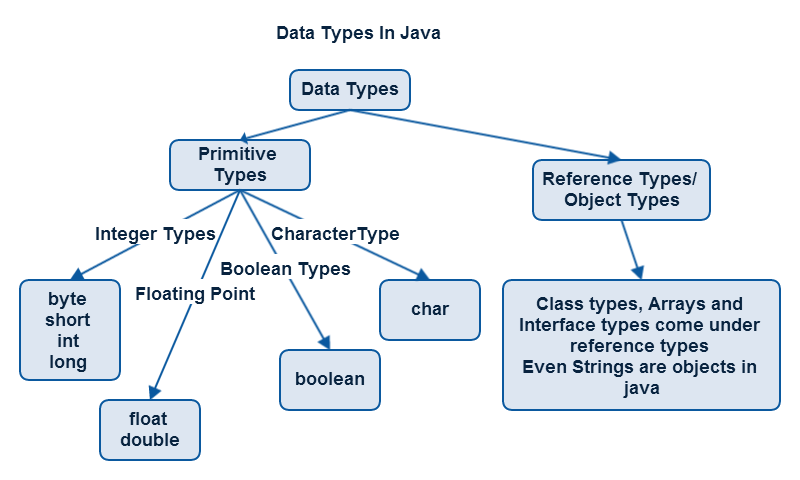
**Өгөгдлийн төрлүүд, Хувьсагчид, болон Массив**

**Өгөгдлийн төрлүүд**

Жава хэлэнд өгөгдлийн төрлийг Primitive data types болон Reference data types гэж хоёр хэсэгт хуваадаг.



Primitive data types буюу Өгөгдлийн үндсэн төрлүүд:

* byte, short, int, long, float, double, char, boolean.

Эдгээр төрлүүдийг дахин 4 бүлэгт хувааж болох юм. Эдгээр 4 бүлэг нь Integers, Floating-point, Characters, Boolean юм.

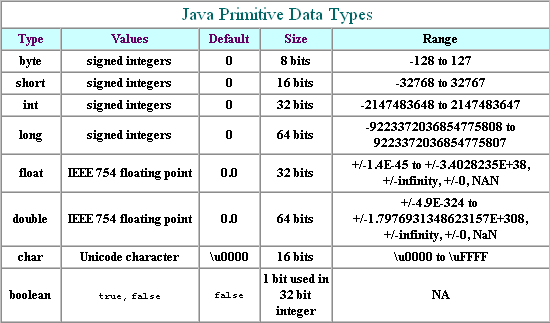
**Integers:** byte, short, int, long төрлүүд хамаардаг. Тоон утгуудыг бүхэлээр нь эерэг болон сөрөг утгатай гаргадаг. Эдгээрийн хоорондын ялгаа нь гэвэл утга авах мужийн хэмжээ юм.

**Floating-point:** float, double төрлүүд хамаардаг. Тоон утгуудыг бутархай нарийвчлалтай гаргадаг ба эдгээр нь өгөгдлийн бодит төрөлд орно. Эдгээрийн хоорондын ялгаа нь гэвэл утга авах мужийн хэмжээ юм. Хэрэвээ та бодит утгыг float төрлөөр илэрхийлэхийг хүсвэл ‘F’ тэмдэгтийг тухайн утгын хамт ашиглах хэрэгтэй.

Жишээ нь: **float radius = 2.5F;** ингэж бичих хэрэгтэй. Хэрэв ‘F’ тэмдэгтийг хасах тохиолдолд (хувиргах операторыг хэрэглээгүй бол) алдаа өгнө.

**Characters:** char төрөл хамаардаг. Өгөгдлийн энэ төрөл нь тэмдэгтийг дүрсэлдэг юм. Энэ төрөл нь бүхэл утгатай харьцаж чадах ба мөн арифметик үйлдэл, их багын харьцуулалт хийж болно. Энэ төрлийн хувьсагчид утга олгохдоо дан кавичик ашиглана. Жишээ нь ‘A’. Хэрэв давхар кавичик хэрэглэвэл мөр төрлийн утга болно.

**Boolean:** boolean төрөл хамаардаг. Энэ төрлийн хувьсагч нь true ба false гэсэн утгуудыг авах ба эдгээр утгууд нь Жавагийн онцгой утгууд болдог. Энэ утгууд нь үйлдлийн харьцаанаас үүсэж болох ба энэ чанарыг операторууд дээр ашигладаг юм.

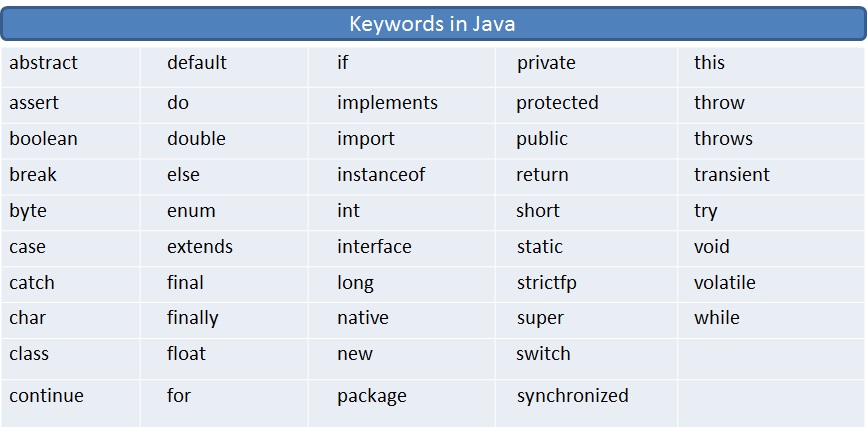
****

Reference data types буюу Заалтан төрлүүд:

* Объект, Массив, Класс, Интерфейс. Жава хэлэнд объект болон массивуудад заалтан төрлөөр ханддаг. Объект болон массивыг үүсгэх үед эдгээрт тодорхой хэмжээг санах ойд тогтоодог ба энэ санах ойн хэсгийн хаягт заалтан хувьсагчдыг хадгалдаг.

**Хувьсагчид**

Хувьсагч бол өгөгдөл хадгалах санах ойн жижигхэн орон зай юм. Хувьсагчийн нэр нь үсгээр (a…z), доогуур зураас ( \_ ), доллароор ( $) эхэлж болно. Хувьсагчдыг хэрэглэхээс өмнө тэдгээрийг заавал зарласан байх ёстой. Том жижиг үсэг ялгаатай тул **name**, **Name** хоёр хувьсагчийг ялгаатай гэж үзнэ. Яагаад гэвэл жава хэл маань case sensetive тул энэ нь юу гэсэн үг вэ гэвэл том жижиг үсгийг ялгадаг гэсэн үг юм. Жавагийн түлхүүр үгээр хувьсагчдыг нэрлэж болохгүй. Доорхи хүснэгтийг харна уу.



Жава хэлэнд түлхүүр үгээр ашиглагддаггүй боловч нөөцлөгдсөн үгс: **const**, **goto**. Жава нь програмыг хөрвүүлэхийн өмнө програмын кодыг **unicode** –руу дотроо хөрвүүлдэг. Тиймээс хувьсагчдын нэрийг **unicode** дээр бичиж болно. Өөрөөр хэлбэл монгол нэртэй хувьсагчид байж болно гэсэн үг.

Хувьсагчийн зарлалт нь

< өгөгдлийн төрөл > хувьсагч 1, хувьсагч 2, .... ; гэсэн бүтэцтэй байна.

Жишээ нь: int төрөлтэй age нэртэй хувьсагч зарлав. **int age;** эсвэл анхны утгатайгаар зарлаж болно. i**nt**  **age = 30;** ингэж бас болно. Нэг төрөлтэй олон хувьсагчдыг ар араас нь таслалаар тусгаарлана бичиж болно. **int age, door\_number, car\_number;**

**Жишээ нь:  
 int count;  
 byte hour, minute, second;**  
 **String firstName;**  
  **float $pi, \_salary, Zeel;**

Анхаарах зүйл: String –ээс бусад төрлийн нэр жижигээр эхэлсэн байна. **String** бол өгөгдлийн үндсэн төрөл биш харин класс тул томоор бичигддэг. Том жижгийг нь соливол алдаа гарна. Тэгэхээр **firstName** бол хувьсагч биш объект болж байна. Объект бол дотроо хувьсагчдыг агуулж болдог бол хувьсагч нь цаашаа задардаггүй.

**Өгөгдөлтэй ажиллах**

Бүхэл тоогоор бичигдсэн тоонууд нь int төрөлтэй байдаг. Тиймээс 1, 9, 3434, -884 тоонууд int төрөлтэй гэсэн үг. Хэрэвээ long төрөлтэй тоо ашиглая гэвэл хойно нь L үсэг нэмж бичдэг. Жишээ нь: 3434L, 9L, 1L, -884L гэх мэт.

16 –тын тооллын систем дэх тоог бичихдээ урд нь 0x нэмж бичдэг.

|  |  |  |
| --- | --- | --- |
| 16 тын системд | Тайлбар | 10 тын системд |
| 0x100 | 1 \* 162 + 0 \* 161+ 0 \* 160 | 256 |
| 0x1234 | 1 \* 163 + 2 \* 162 + 3 \* 161 + 4 \* 160 | 4660 |
| 0xDEAF | 13 \* 163 + 14 \* 162 + 10 \* 161 + 15 \* 160 | 57007 |
| 0xCAB | 12 \* 162 + 10 \* 161 + 11 \* 160 | 3243 |

8 –тын тооллын систем дэх тоог бичихдээ урд нь **0** ( тэг ) –ыг нэмж бичдэг**.**

|  |  |  |
| --- | --- | --- |
| 16 тын системд | Тайлбар | 10 тын системд |
| 010 | 1 \* 81 + 0 \* 80 | 8 |
| 012 | 1 \* 81 + 2 \* 80 | 10 |
| 023 | 2 \* 81 + 3 \* 80 | 19 |
| 033 | 3 \* 81 + 3 \* 80 | 27 |

Хэрвээ маш том тоо эсвэл маш жижиг тоо ашиглах шаардлага гарвал E тэмдэглэгээг ашиглана. Жишээ нь: дэлхийгээс сар хүртлэх зай: 149.600.000 км гэхийг 1.496E8 гэж бичвэл илүү тохиромжтой. double sunDistance = 1.496E8;

Электрон масс:

0.0000000000000000000000000009 гр хэмжээтэй гэхийг float electronMass = 9E-28F;

**Массив**

Нэгэн төрөлтэй бүлэг өгөгдлийг санах ойд дэс дарааллуулан хадгалах зүйлийг массив гэнэ. Массив дахь өгөгдөлүүд нь өгөгдлийн үндсэн төрөл эсвэл объект байж болно. Массивын элементүүдэд хандаж болно. Элементүүдэд хандахдаа массивын нэр болон индексээр нь ханддаг. Элементийн дугаар нь 0 –ээр эхэлдэг. Массивыг зарлахдаа дараах хэлбэрээр зарлаж болно. Массивыг дотор нь нэг хэмжээст болон олон хэмжээст гэж ангилдаг.

Зөвхөн зарлагаа хэлбэрээр:   
 **datatype** = массивын төрөл **identifier** = массивын нэр **size** = массивын хэмжээ

**datatype[] identifier; char[] ch;**

Зарлагаа болон үүсгэх хэлбэрээр:

**datatype[] identifier = new datatype[size] char[] ch = new char[10]**

Зарлагаа болон анхны утгатайгаар үүсгэх хэлбэрээр:

**datatype[] identifier = {value1, value2,…, valueN}; char[] ch = {‘A’,’B’,’C’};**

Хэрвээ массивт байх элементийн тоо тодорхойгүй байгаа бол доорхи хэлбэрээр зарлаж болно.

**int [ ] number = null;**

Массивын индекс хэзээ ч 0 –ээс бага байж болохгүй. Массивын индекс нь массивын элементүүдийн тооноос хэтэрч болохгүй. Массивын элементэд хандахдаа хаалтан дотор нь элементийнх нь дугаарыг тавьж хандана. Хаалтан дотор нь мөн бүхэл тоо арифметик илэрхийлэл байж болно.

Жишээ нь:

int x = 1;  
int number [ ] = new int [10];  
number [0] = 10;  
number [0] ++; // number[0] нь 11 болно  
++ number [0]; // number[0] нь 12 болно  
number[1] = -2;  
number [2+x\*3] = 7; // number[5] = 7 болно  
System.out.println(number[0]); // 12 гэж хэвлэнэ

Хэрвээ массивын бүх элементэд ямар ч утга тогтоохгүйгээр байх үед жава нь массивын бүх элементүүдэд автоматаар утга тогтоож өгсөн байдаг ба энэ нь өгөгдлийн төрлөөс хамаардаг юм. Жишээ нь: int төрөлтэй массивын элементүүдэд автоматаар утгыг нь 0 –ээр тогтоож өгдөг.

**Byte, short, int, long** төрөлтэй массив зарлахад **0** утгыг тогтоодог

**double, float** төрөлтэй массив зарлахад **0.0** утгыг тогтоодог

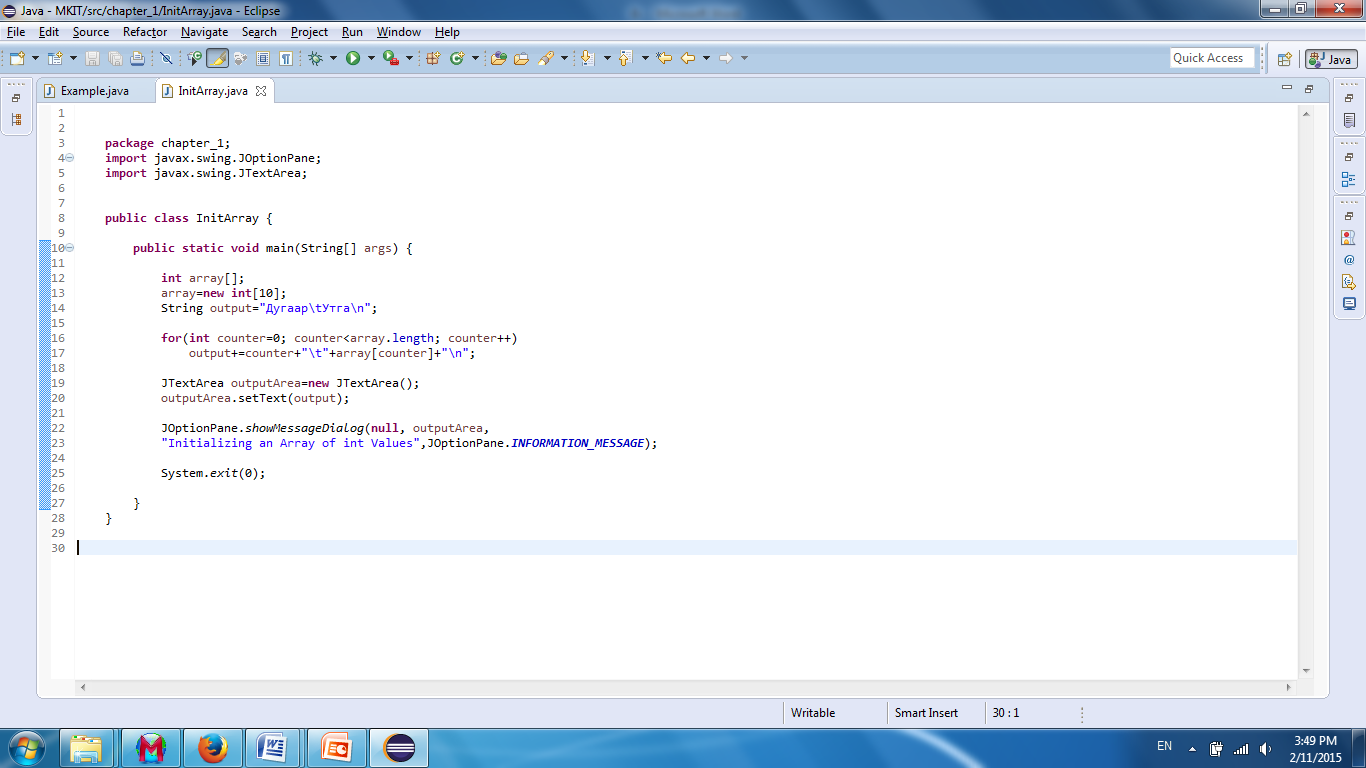
**char** төрөлтэй бол **хоосон** утгагүй

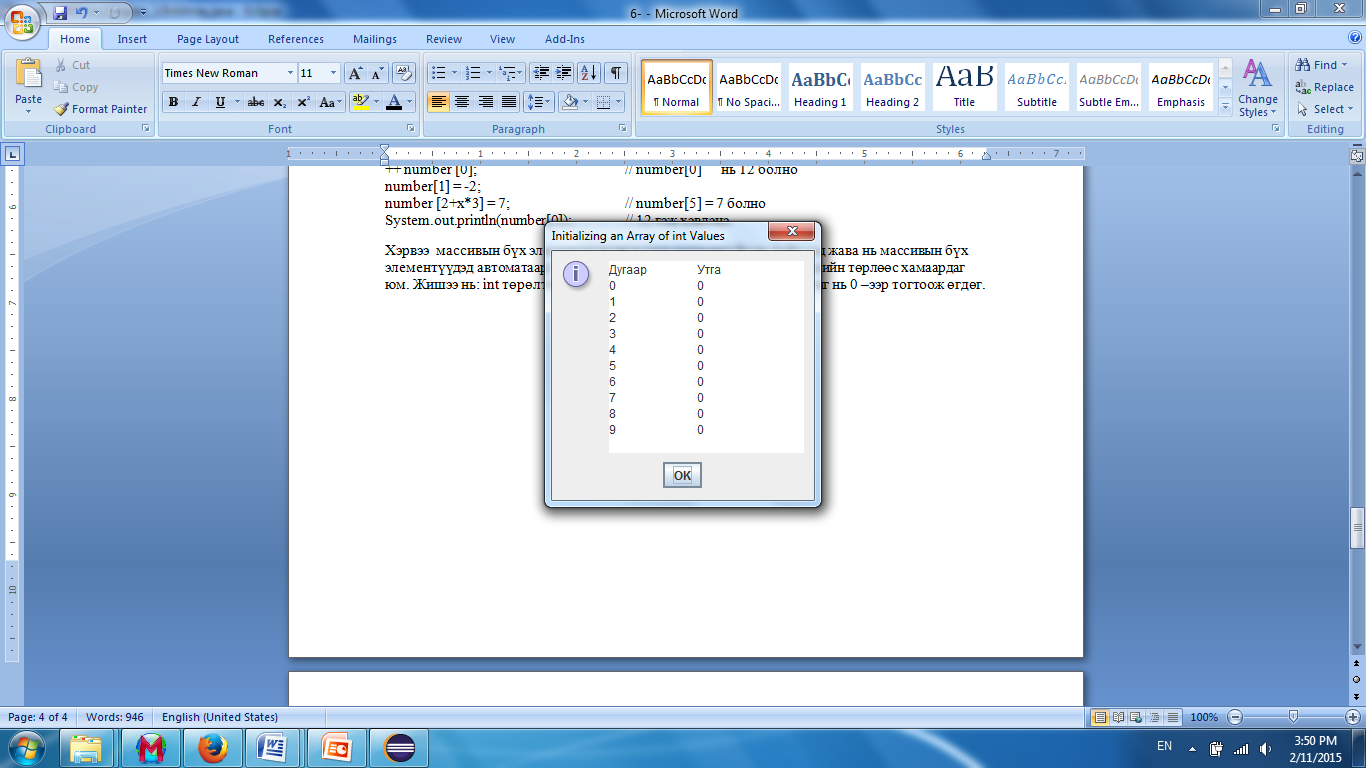
**boolean** төрөлтэй бол **false** утгыг тогтоодог

**String** төрөлтэй бол **null** утгыг тогтоодог

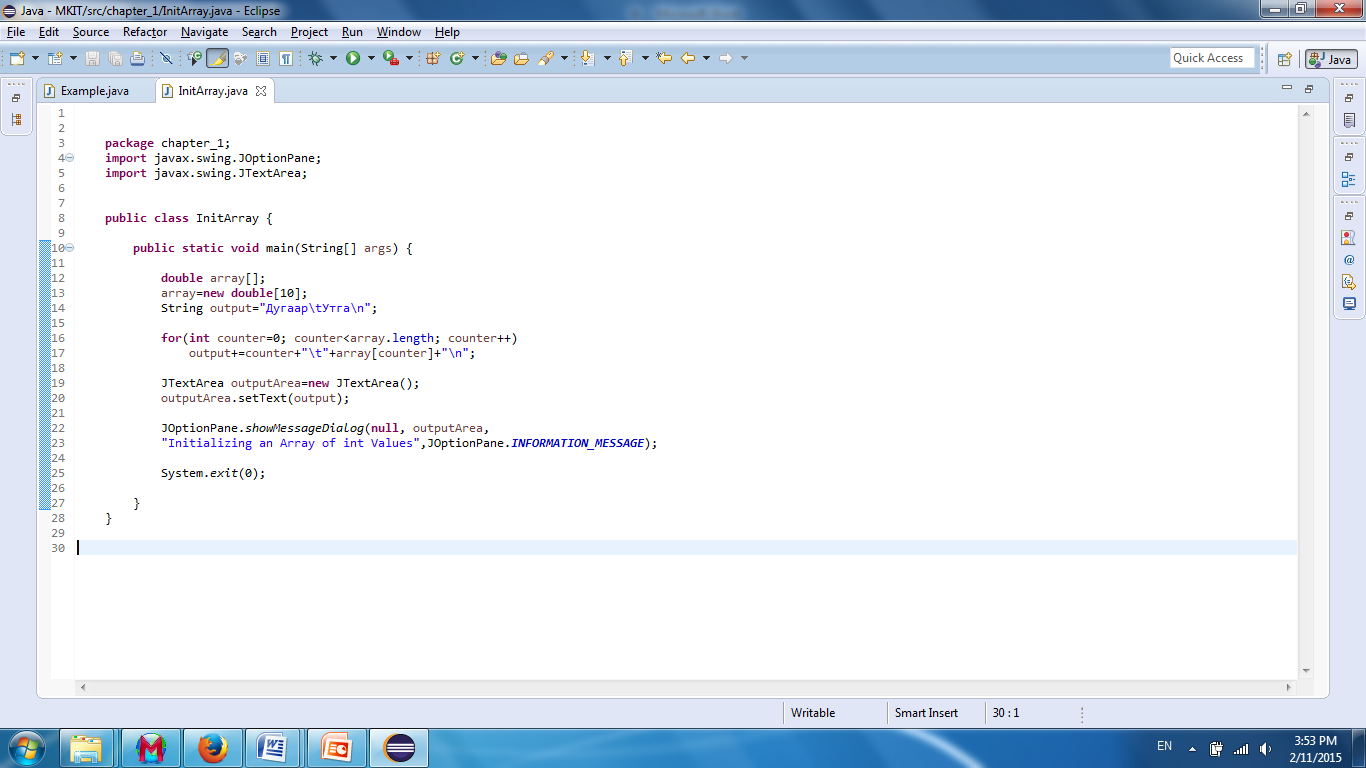
Доорхи жишээ кодуудыг харна уу.

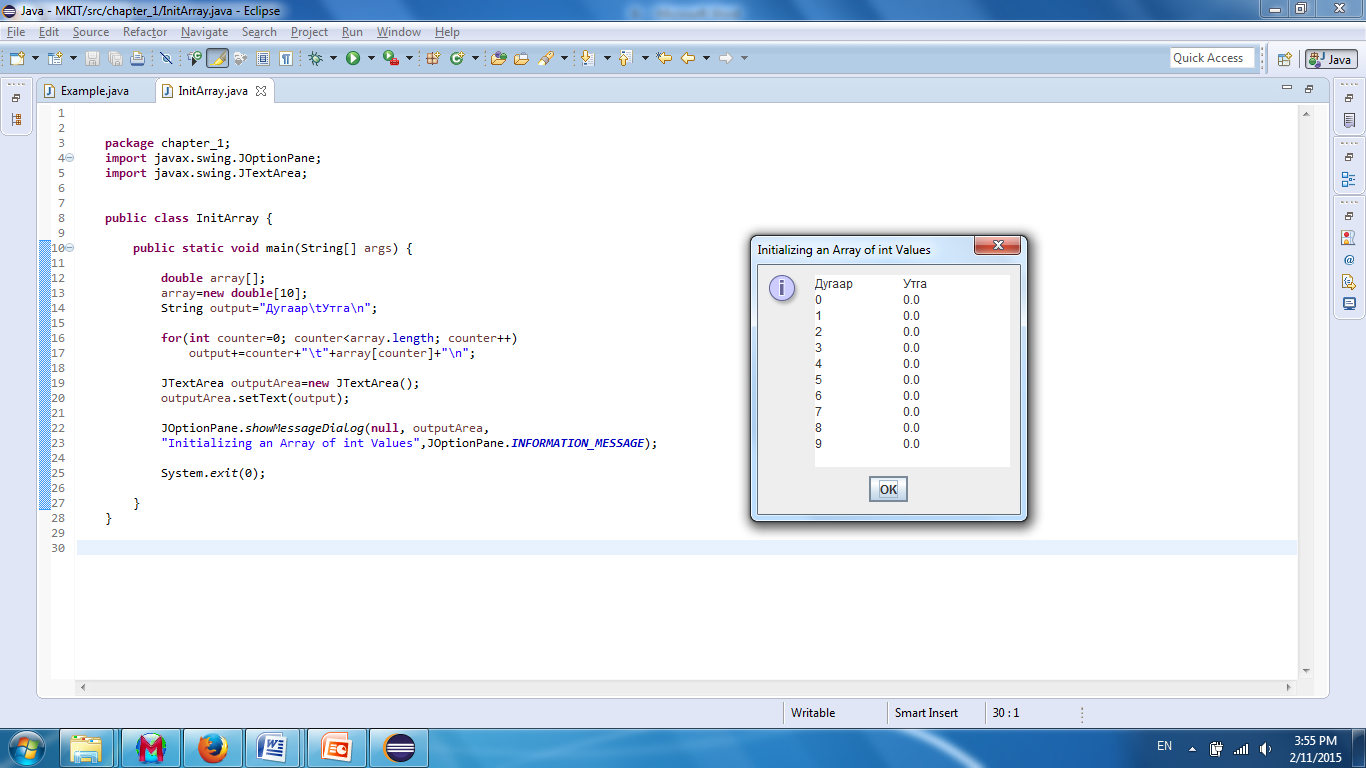
**int** төрөлтэй массив зарласан жишээ. Мөн **byte, short, long** төрлүүд дээр мөн адил байна.



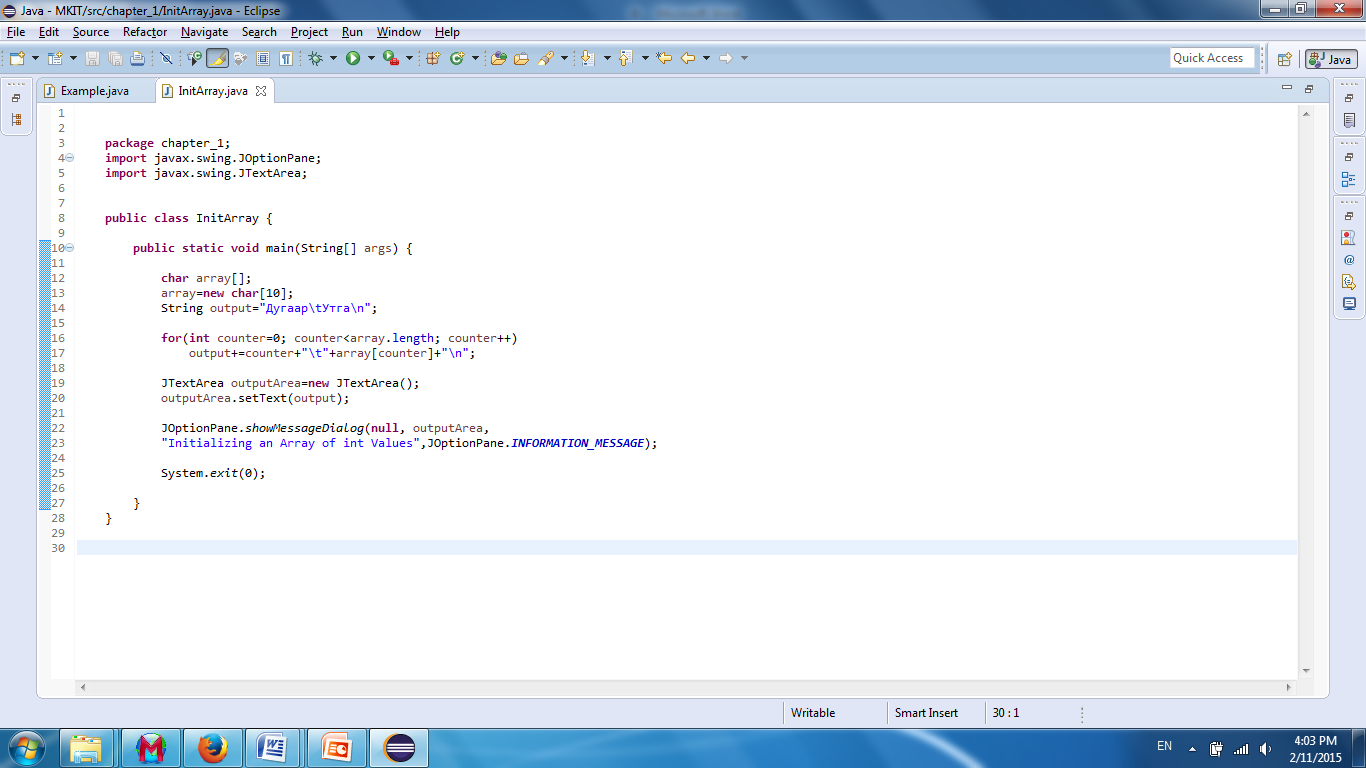


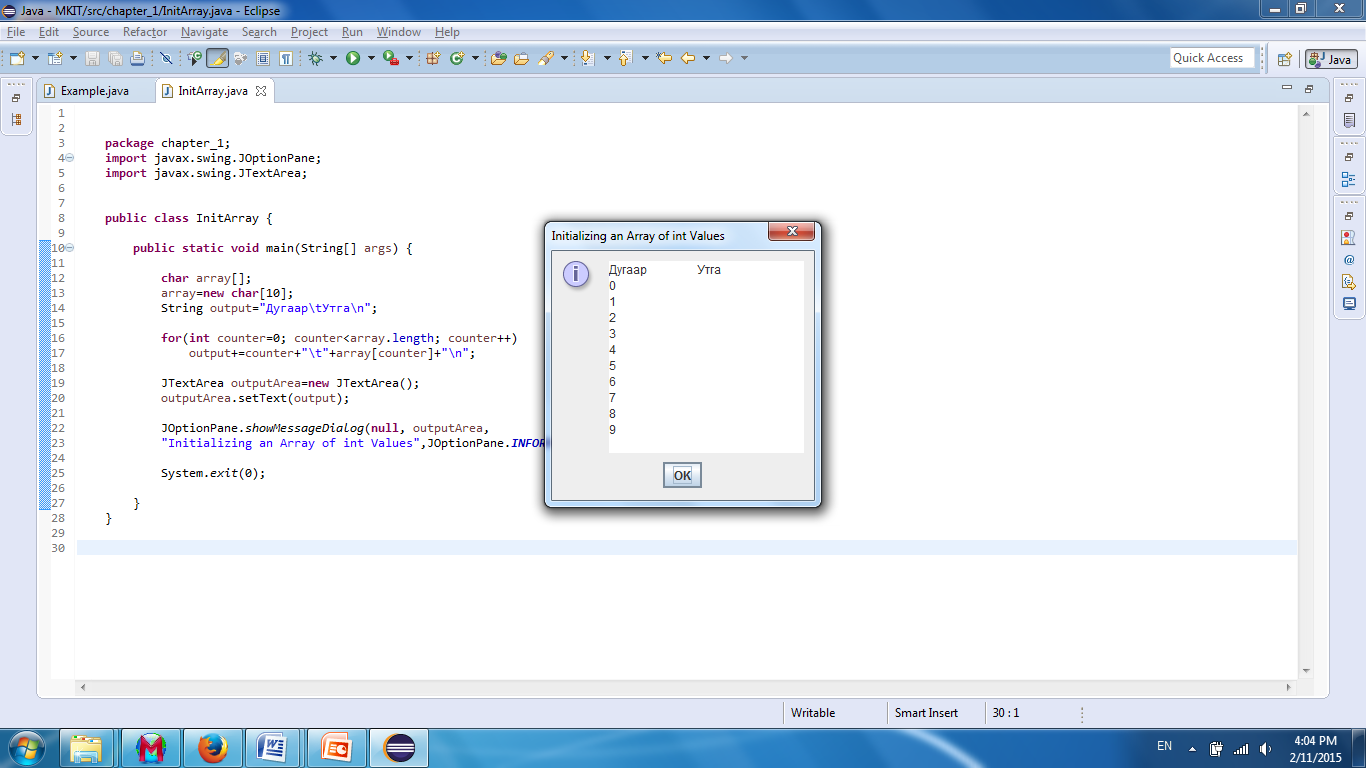
**double** төрөлтэй массив зарласан жишээ. Мөн **float** төрөл дээр адил байна.



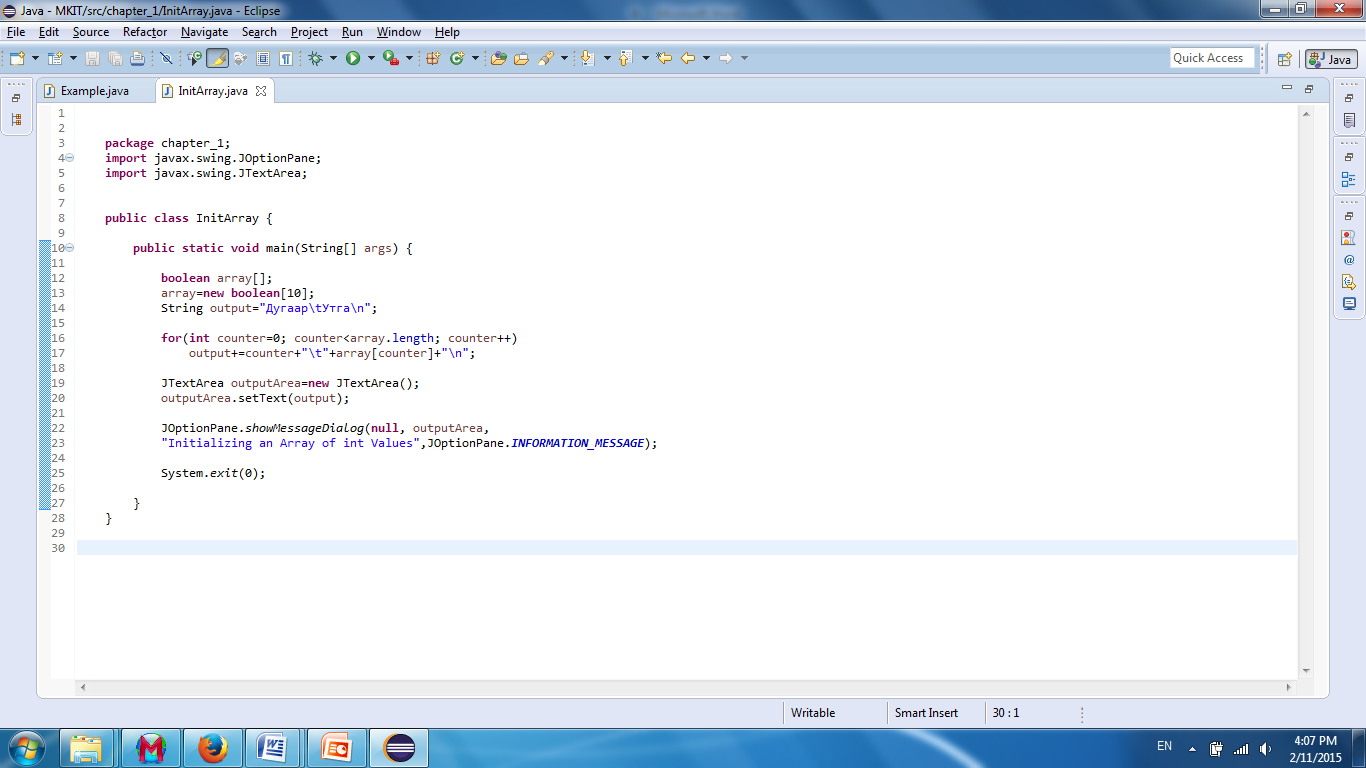


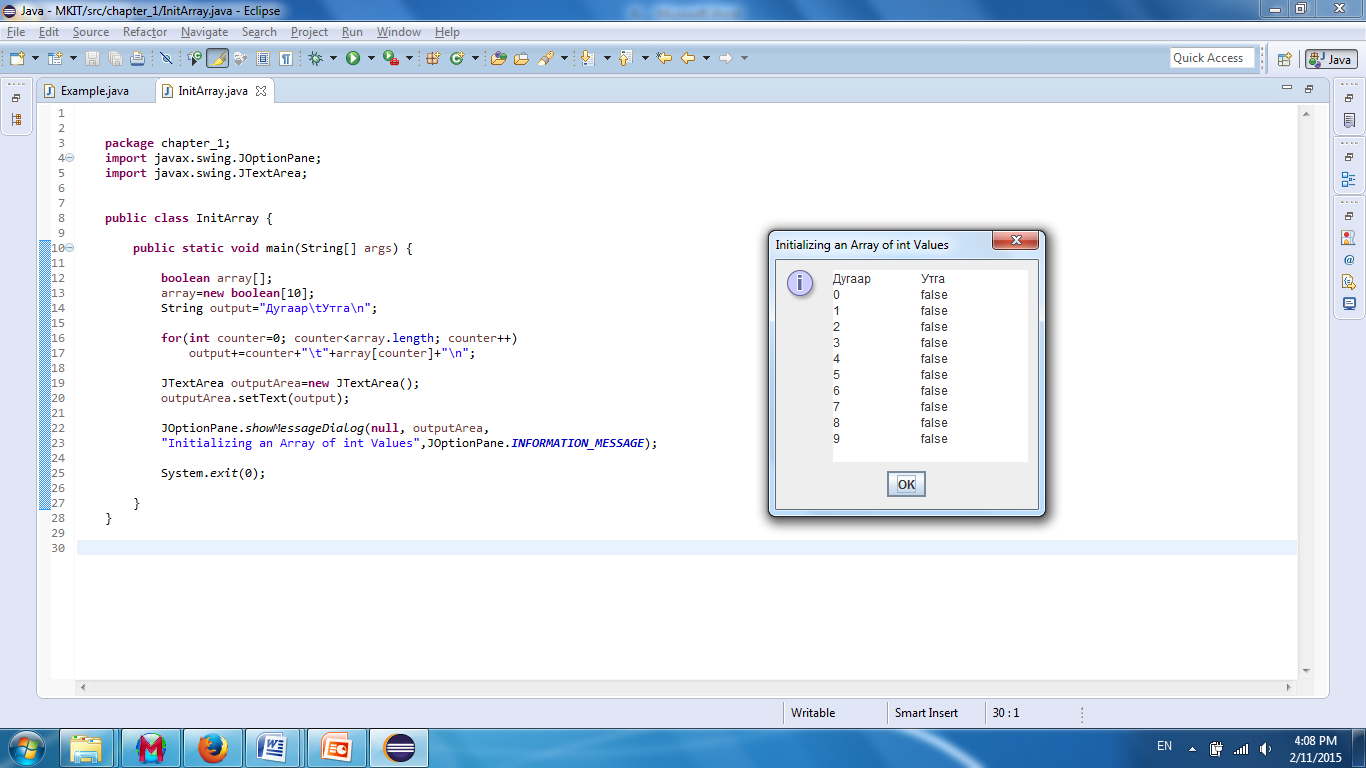
**char** төрөлтэй массив зарласан жишээ.



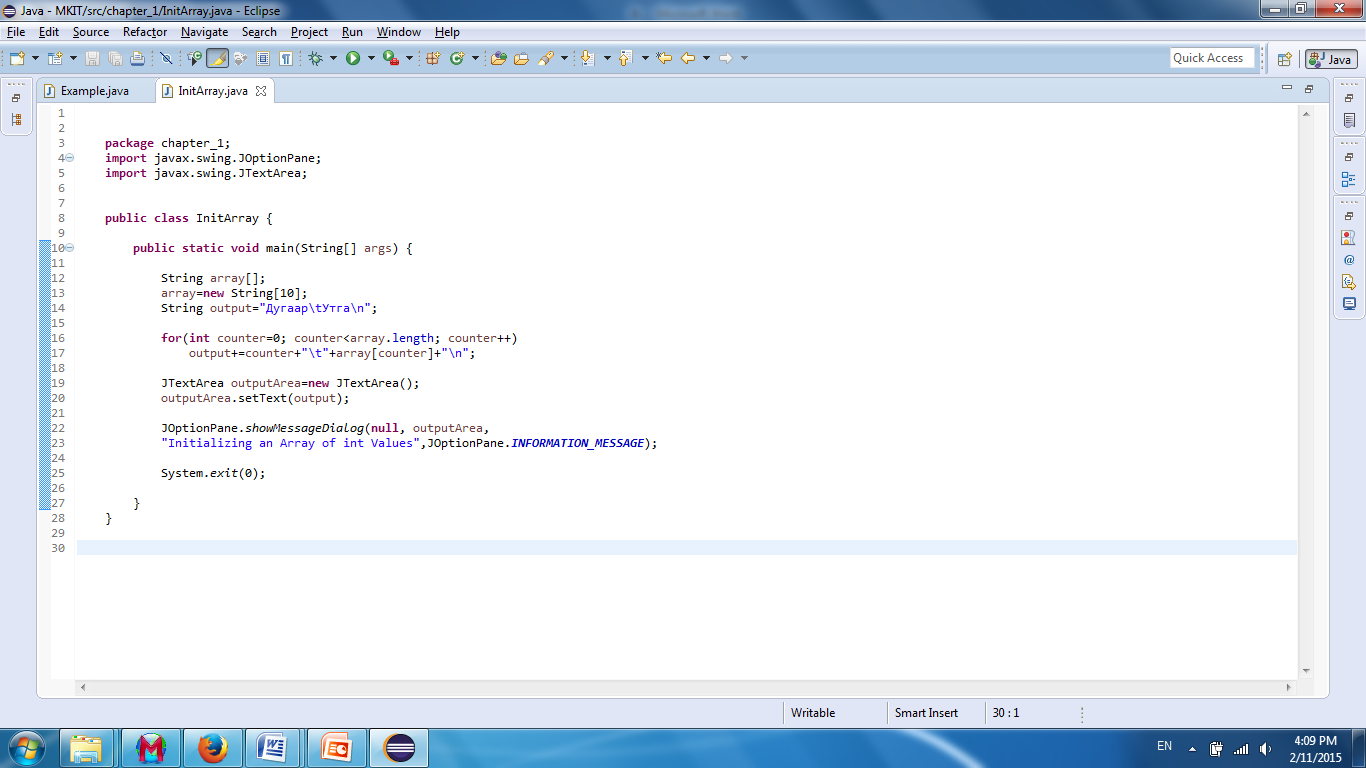


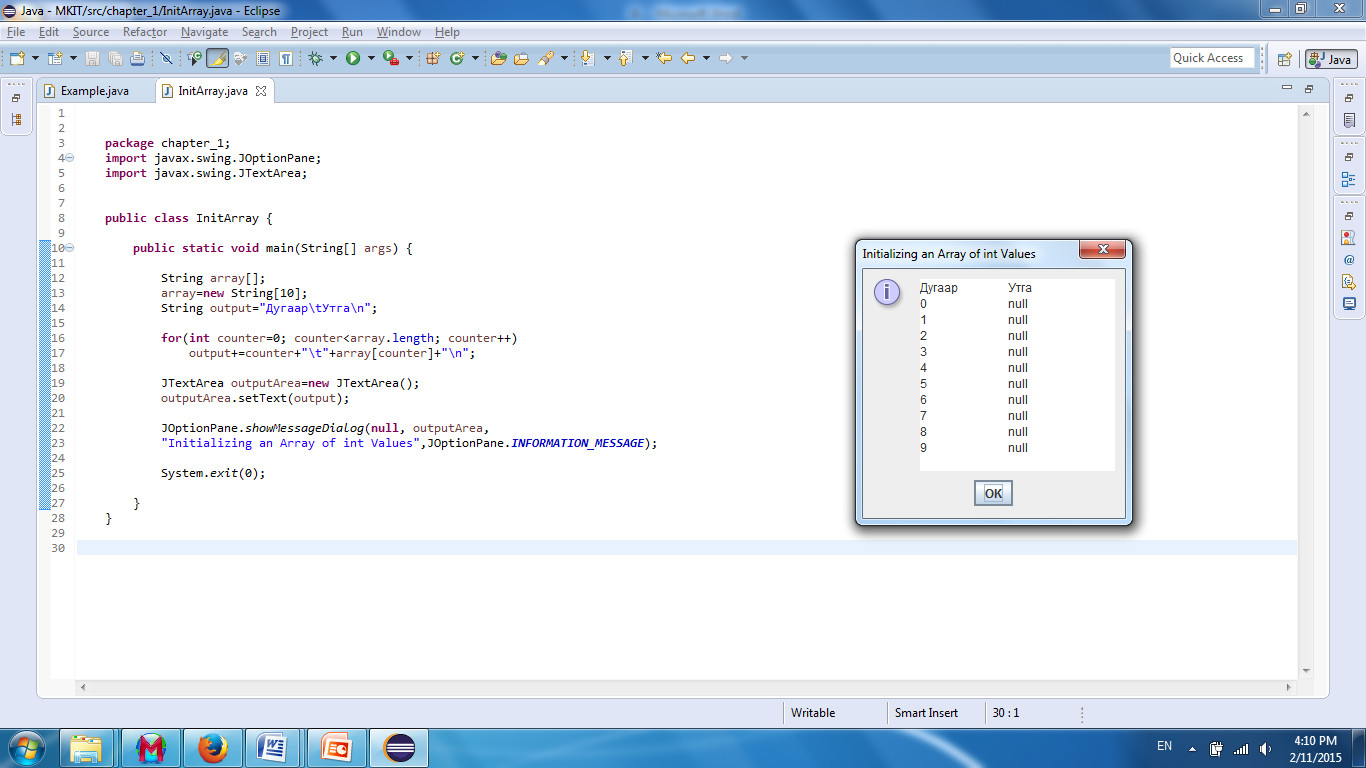
**boolean** төрөлтэй массив зарласан жишээ.





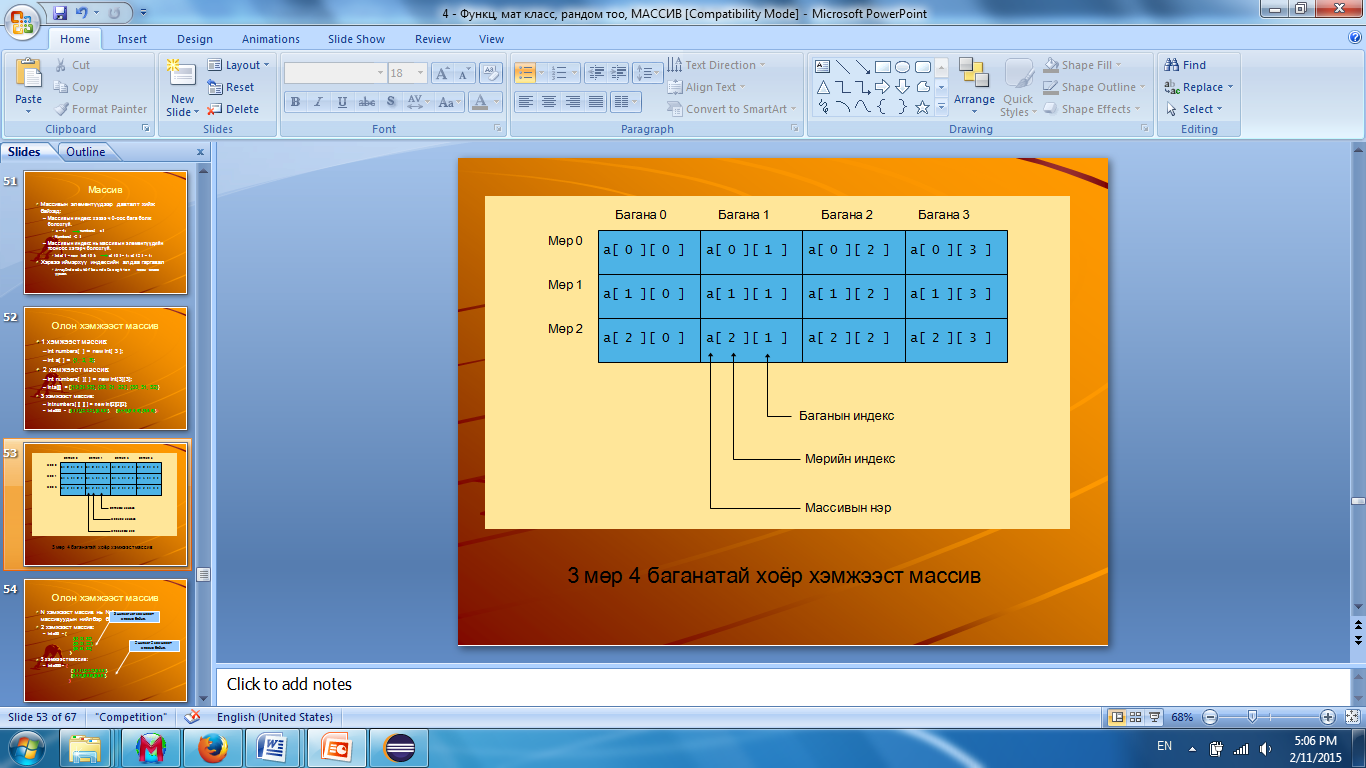
**String** төрөлтэй массив зарласан жишээ.





**Олон хэмжээст массив**

Бидний өмнө үзсэн массив бол нэг хэмжээст массив юм. Жава –д хэдэн ч тооны хэмжээтэй массивыг үүсгэж болно. Бид одоо хоёр болон гурван хэмжээст массивыг үүсгэж үзье. Доорхи зурганд 3 мөр 4 багантай 2 хэмжээст массивын зураг байна.



Хоёр хэмжээст массив.

**int number [ ][ ] = new int [3] [3];**

**int number [ ][ ] = {{10,11,12},{20,21,22},{30,31,32}};**

Гурван хэмжээст массив.

**int number [ ][ ][ ] = new int [2][3][3];**

**int number [2][3][3] = {**

**{{11,12,13},{21,22,23},{31,32,33}},**

**{{41,42,43},{51,52,53},{61,62,63}}**

**};**

**number[1][0][2] = 43**