МОВА ПРОГРАМУВАННЯ
JAVA

Лекція 9. Основи мови програмування Java

План лекції:

1. Виникнення мови Java
2. Особливості мови Java
3. Змінні та типи даних
4. Основні оператори
5. Керуючі інструкції
6. Масиви
7. Класи та об’єкти
8. Конструктори

Технології Java та література

Література:

1. П. Ноутон, Г. Шилдт. Java 2. Наиболее полное руководство
2. К.С. Хорстманн, Г. Корнелл. Java 2 (том 1 и 2)
3. И. Хабибуллин. Java. Самоучитель

Технології

та підтримка лекційного курсу:

1. Sun Microsystems
   - www.sun.com
   - www.oracle.com
   - www.java.com
   - Java (jdk & jre)
   - NetBeans

Виникнення мови Java

Люди та події:

Ґенеза мов:
B =► C =► C++ =► Java

Мова Java створювалась з 1991 по 1995 роки, корпорація Sum Microsystems

Авторы мови Java (Oak):

Джеймс Гослінг, Патрік Ноутон, Кріс Варт, Ед Франк, Майк Шерідан

Ідеї:

Java: необхідність створення ефективних програм для Internet

Незалежність від конкретної комп'ютерної платформи та типу процесору

Компілятор Java видає не код виконання, а байт-код, який виконується не операційною системою, а віртуальною машиною Java (JVM)

Відмінності між мовами C++ та Java

На відміну від C++

В мові Java немає:

1. Вказівників!!!
2. Структур і об'єднань
3. Перевантаження операторів!!!
4. Автоматичного приведення типів з втратою точності
5. Глобальних змінних і функцій
6. Значень аргументів за умовчанням!!!
7. Деструкторів!!!
8. Оператора goto
9. Передачі об'єктів за значенням (тільки за посиланням!)

В мові Java є:

1. Багатопотоковість
2. Пакети
3. Інтерфейси (аналог абстрактного класу в C++)
4. Вбудований рядковий тип String
5. Документаційний коментар
6. Всі масиви динамічні!

Відмінність властивостей:

1. В C++ true і false – можуть бути числами, в Java – тільки літерали!
2. В C++ специфікатор рівня доступу застосовується до груп полів, в Java – для кожного поля окремо

Особливості мови Java

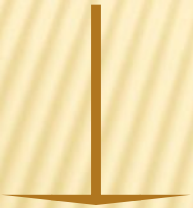
Мова Java повністю об’єктно-орієнтована!!!

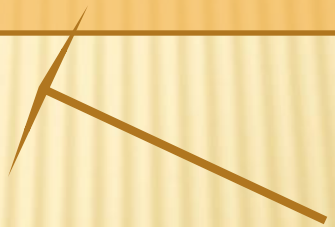
Базові принципи ООП

- **Інкапсуляція** — Програмний код + дані
- **Поліморфізм** — Єдиний інтерфейс
- **Наслідування** — Передача властивостей

Приклад програми на Java

```
class Intro{
public static void main(String[] args) {
System.out.println("Ми програмуємо на Java!");
}
}
```





Ми програмуємо на Java!

ЗАПУСТИТИ

Типи даних Java

Цілі числа

byte	8 біт
short	16 біт
int	32 біт
long	64 біт

Числа з плаваючою точкою

float	32 біт
double	64 біт

Символи

char	16 біт

Логічний тип

boolean	true або false

Літерали

1, 2, 3, …	int
125L	long
012	8-ричне
0x12	16-ричне
2.0	double
2.0F	float
‘\xxx’	8-ричний символ Unicode
‘\uxxx’	16-ричний символ Unicode

Система Unicode –
повний набір символів

Змінні

Оголошення

```
int a,b,c;
int d=3,f,k=5;
char x='x';
```

Автоматичне приведення типів

1. Два типи мають бути сумісними
2. Цільовий тип має бути "більшим" за вихідний

Динамічна ініціалізація

```
double a=3.0,b=4.0;
double c=a*a+b*b;
```

Правила розширення типів

byte і short => int

Явне приведення типу

Вихід за межі діапазону: виконується перетворення до типу int

ПОМИЛКА: літерал 2 має тип int

```
byte
b=50;
b=b*2;
```

ПРАВИЛЬНО!!!

```
byte b=50;
b=(byte)(b*2);
```

Область доступності змінних

визначається межами блока (пара дужок { та }), де вони визначені. Коли запускається новий блок, створюється відповідна область доступності

Основні оператори Java

Групи операторів

- Арифметичні
- Логічні
- Побітові (порозрядні)
- Оператори порівняння

Порозрядні оператори

~, &, |, ^, >>, <<, >>>, &=, |=, ^=, >>=, <<=, >>>=

Операція присвоєння

змінна=вираз;

Арифметичні

+, -, *, /, %, ++, --, +=, -=, *=, /=, %=, --

Умовна тернарна операція

умова?вираз1:вираз2;

Логічні

& (скорочена форма &&), | (скорочена форма ||), ^, !

Пріоритет операторів

1. () [] .
2. ++ -- ~ !
3. * / %
4. + -
5. >> << >>>
6. > >= <= <
7. == !=
8. &
9. ^
10. |
11. &&
12. ||
13. ?:
14. = op=

Оператори відношення

==, !=, >, <, >=, <=

Керуючі інструкції Java

Оператор if()

```
if(умова) оператор1;
else оператор2;
```

Оператор switch()

```
switch(вираз){
case значення1:
//...
break;
case значення2:
//...
break;
.....
default:
//...
}
```

Оператор for()

```
for(ініціалізація;умова;ітерації ){
//...
}
```

Оператор while()

```
while(умова){
//...
}
```

Оператор do-while()

```
do{
//...
} while(умова);
```

Аналогічно до відповідних операторів мови C++

Масиви в Java

Оголошення масиву

тип ім’я[ ];

Виділення пам’яті

ім’я=new тип[розмір];

Приклад 1

```
int data[ ];
data=new int[12];
```

Приклад 2

```
int data[ ]=new int[12];
byte[ ] a=new byte[100];
```

Приклад 3

```
int data[ ]={1,2,3,...};
```

Індексація масиву починається з нуля!

Багатомірні масиви

```
int data1[ ][ ]=new int[4][5];
int data2[ ][ ]=new int[3][ ];
data2[0]=new int[5];
data2[1]=new int[3];
data2[2]=new int[1];
```

Ініціалізація

```
int m[ ][ ]={
{1,2,3},
{4,5,6}
};
```

В Java ВИКОНУЄТЬСЯ перевірка на предмет виходу за межі масива!!!

В Java всі масиви ДИНАМІЧНІ!!!

Класи та об’єкти

Оголошення класу

```
class ім’я_класу{
тип змінна;
.....
тип метод(){
//тіло методу}
.....
}
```

Приклад оголошення класу

```
class MyClass{
double x;
int m;
void set(double z, int n){
x=z;
m=n;}
}
```

Створення об’єкту

```
клас об’єкт;
об’єкт=new клас();
або
клас об’єкт=new клас();
```

Приклад створення об’єкту

```
MyClass obj;
obj=new MyClass();
```

або те саме:

```
MyClass obj=new MyClass();
```

- Любий java-клас має бути повністю визначений в одному файлі
- Оголошення і реалізація методів розміщені разом
- Метод main() визначається в класі, якщо він є стартовою точкою програми

Класи та об’єкти – приклади 1

```
class Box{
double width;
double height;
double depth;
}
class BoxDemo{
public static void main(String[] args){
Box mybox=new Box();
double vol;
mybox.width=10;
mybox.height=20;
mybox.depth=15;
vol=mybox.width*mybox.height*mybox.depth;
System.out.println(“Об’єм дорівнює ”+vol);}
}
```

- Любий java-клас має бути повністю визначений в одному файлі
- Оголошення і реалізація методів розміщені разом
- Метод main() визначається в класі, якщо він є стартовою точкою програми

Після компіляції буде створено 2 файли (з розширенням class): Box.class та BoxDemo.class (кожен клас компілюється в окремий файл). Виконувати треба файл BoxDemo.class

ЗАПУСТИТИ

Класи та об’єкти – приклади 2

```
class Box{
double width;
double height;
double depth;
void volume(){
System.out.print(“Об’єм дорівнює ”);
System.out.println(width*height*depth);
}
}
class BoxDemo2{
public static void main(String[] args){
Box mybox=new Box();
mybox.width=5;
mybox.height=16;
mybox.depth=25;
mybox.volume();
}
}
```

ЗАПУСТИТИ

Конструктори в Java

- Конструктором виконується ініціалізація об’єктів після створення
- Конструктори не мають специфікатора типу
- Ім’я конструктора співпадає з іменем класу

```
class Box{
double width, height, depth;
double volume(){
return width*height*depth;}
Box(){
System.out.println(“Створення об’єкту!”);
width=10;
height=20;
depth=30;}}
class BoxDemo3{
public static void main(String[] args){
Box mybox=new Box();
System.out.println(“Об’єм дорівнює ”+mybox.volume());}
}
```

ЗАПУСТИТИ

Конструктор з аргументами

- Ключове слово this – посилання на об’єкт, метод якого викликано
- В Java імена об’єктів є фактично, посиланнями на ці об’єкти

```
class Box{
double width, height, depth;
double volume(){
return width*height*depth;}
Box(double x,double y,double depth){
System.out.println(“Створення об’єкту!”);
width=x;
height=y;
this.depth=depth;}}
class BoxDemo4{
public static void main(String[] args){
Box mybox1=new Box(5,6,8);
Box mybox2=new Box(10,10,10);
mybox2=mybox1;
System.out.println(“Об’єм дорівнює ”+mybox2.volume());}
}
```

ЗАПУСТИТИ