

# Основи програмування

## Заняття №3

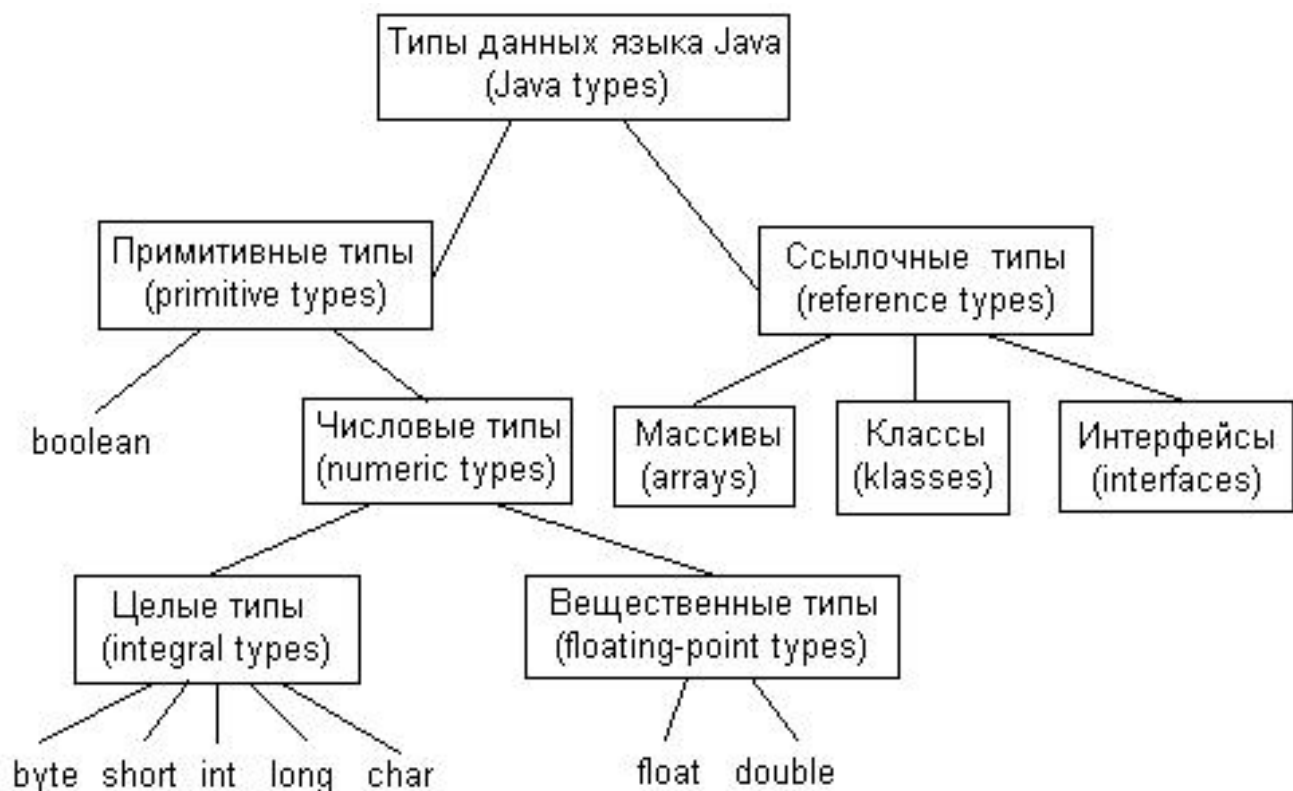
### 1. Компільовані та інтерпретовані мови програмування?

Відомі два основні різновиди трансляторів: **компілятори** та **інтерпретатори**. **Компілятори** спочатку повністю перекладають весь текст програми з мови високого рівня мовою машинних команд, щоб потім можна було запускати отриману в результаті цього машинну програму. При використанні компілятора етап трансляції програми чітко відділено від етапу її виконання: достатньо один раз відкомпілювати програму, щоб потім запускати отриманий результат безліч разів. Фактично, після компіляції вихідний текст програми, який був написаний мовою високого рівня, більше не потрібен. **Інтерпретатори** натомість читають текст програми мовою високого рівня та виконують його по мірі прочитання. Переклад програми на машинну мову не запам'ятовується, отже, щоб виконати ту ж програму вдруге, її потрібно знов пропустити через інтерпретатор.

**Компілятори** з одного боку та **інтерпретатори** з іншого мають свої недоліки та переваги. Навіть якщо текст програми недописаний чи містить деякі помилки, інтерпретатор може виконати хоча б частину її тексту, поки не натрапить на першу помилку. Компілятор мусить спочатку перекласти машинною мовою весь текст програми, і лише після цього її можна запустити на виконання. Якщо програма містить хоча б одну помилку, компілятор не створить машинної програми, отже зробити пробний запуск такої програми неможливо. Таким чином, інтерпретатори набагато більш гнучкі. Але оборотною стороною є ненадійність інтерпретаторів: компілятор принаймні гарантує, що програму можна буде запустити на виконання лише тоді, коли в усьому її тексті немає граматичних помилок, а коли програма виконується під інтерпретатором, в ній можуть міститися «міни уповільненої дії» — помилки, які проявлять себе лише тоді, коли до них дійде виконання.

Крім того, запуск відкомпільованої програми вимагає набагато менших машинних ресурсів (пам'яті та процесорного часу): компілятор один раз і наперед виконує всю складну роботу з розбору, аналізу вихідного тексту програми та підготовки машинного коду, тому в подальшому запускається програма, вже перетворена у найзручнішу для машинного виконання форму. Якщо ж одну й ту саму програму кілька разів запускати під інтерпретатором, то інтерпретатор щоразу буде змушений заново читати та аналізувати її текст.

## 2. Типи даних Java?



Розглянемо детальніше примітивні типи даних :

<b>datatype</b>	<b>size</b>	<b>range</b>	<b>Default value</b>	<b>Wrapper class</b>
<b>byte</b>	1 byte	-128 to 127	0	Byte
<b>short</b>	2 bytes	-32768 to 32767	0	Short
<b>int</b>	4 bytes	$-2^{31}$ to $2^{31}-1$	0	Integer
<b>long</b>	8 bytes	$-2^{63}$ to $2^{63}-1$	0	Long
<b>float</b>	4 bytes	-3.4e38 to 3.4e38	0.0	Float
<b>double</b>	8 bytes	-1.7e308 to 1.7e308	0.0	Double
<b>boolean</b>	NA	NA(But allowed values are true, false)	false	Boolean
<b>char</b>	2 bytes	0 to 65535	0(blank spaces)	Character

### 3. Розгалуження в коді?

Отже, ключове запитання, як же записувати умову в коді java.

#### А) Конструкція if-else

```
if(boolean expression){  
    //Блок коду, що буде виконуватись, якщо умова буде true.  
}else{  
    // Блок коду, що буде виконуватись, якщо умова буде false.  
}
```

#### Б) Конструкція if-else-if

```
if (boolean expression 1){  
    do something...  
} else if (another boolean expression 2) {  
    do something else...if expression2 is true  
}
```

#### В) Конструкція if-else-if(безліч разів)

```
if (boolean expression){  
    do something...  
} else if (another boolean expression) {  
    do something else... if expression is true  
} else if (another boolean expression2) {  
    do something else... if expression2 is true  
} else if (another boolean expression3) {  
    do something else... if expression3 is true  
} else{  
    do something else...  
}
```

### 4. Що таке функції , призначення, область видимості, повернення значення функції?

Отже функції, а їх так рідко називають в Java, а частіше зустрічається слово **методи** - це аналог підпрограм, функцій, процедур в інших мовах програмування. За допомогою методів ми виносимо текст повторюваного коду програми окремо в тіло методу, після чого можна викликати даний метод з будь-якого місця програми, безліч разів.

Спрощене оголошення та визначення методу, який ми зараз будемо використовувати, має вигляд:

```
тип_повернення  назва_методу(параметри) {  
  
    //тіло методу;  
    інструкція1;  
    інструкція2;  
    ....  
    інструкціяN;  
}
```

тип\_повернення – результат виконання методу, наприклад він може повертати об’єм сейфу, тоді тип\_повернення буде double, int ... (будь-який примітив або клас). Якщо метод нічого не повертає, то вказується слово ключове слово void.

### *Оголошення та визначення*

```
int methodName (int parameterName){  
return 5;  
}
```

Спершу оголошується **тип повернення** (int, double, boolean, String, Object, etc.), **назва методу**. Якщо у метод потрібно передати дані – це можна зробити через **параметри**, які вказуються у дужках після оголошення назви методу.

### **Приклади запису методів :**

```
int sum(int a, int b){  
return a + b;  
}
```

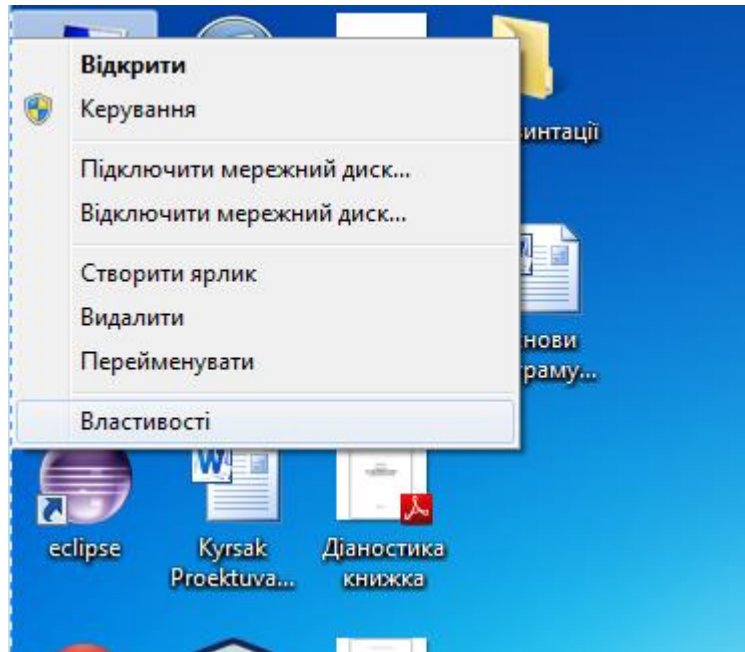
```
double division (int a, int b){  
return a/b;  
}
```

## **5. Створюємо програму HelloWorld в блокноті.**

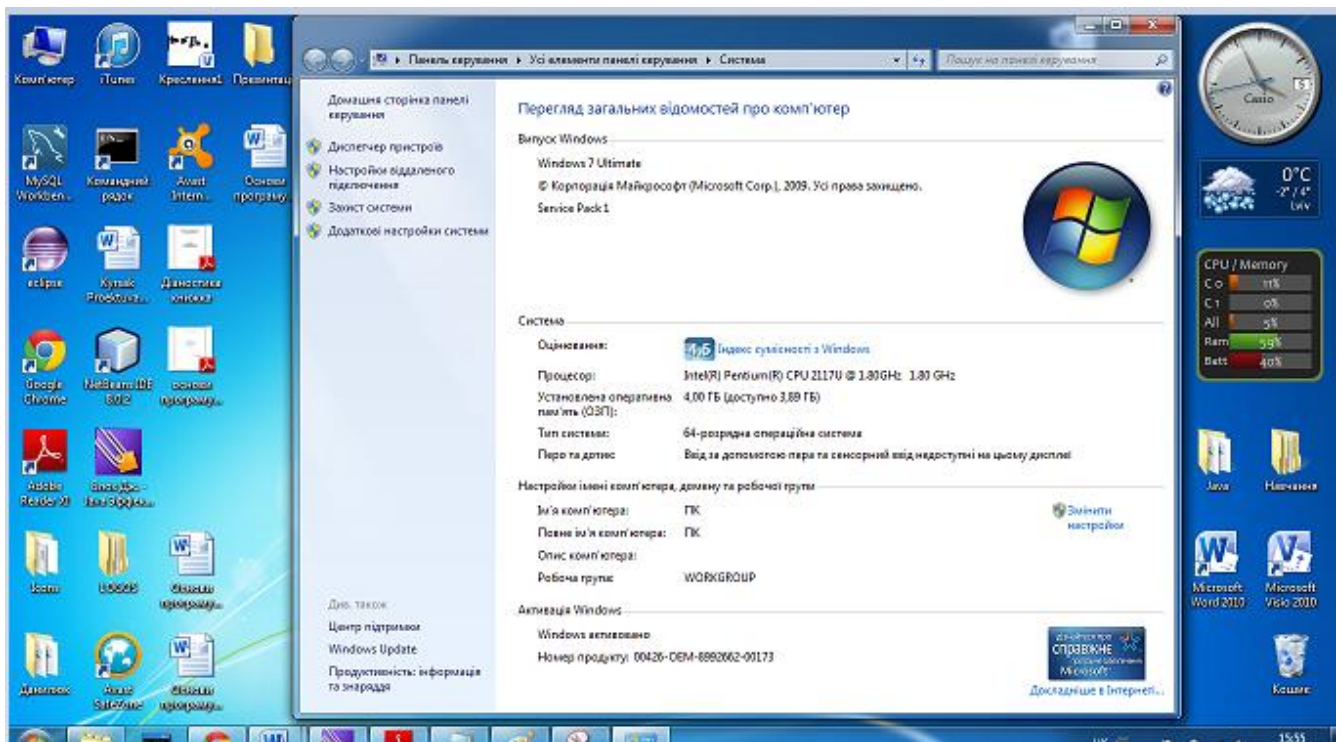
1. Встановлюємо JdK. В мене воно встановилось в папці C:\Program Files\Java\jdk1.7.0\_67

2. Необхідно вказати системі де наш javac і java.exe . Вони знаходяться тут C:\Program Files\Java\jdk1.7.0\_67

3. Створюємо системну змінну і додаємо її до системної змінної PATH.  
Натискаємо праву кнопку миші на ярлику «мій компютер»

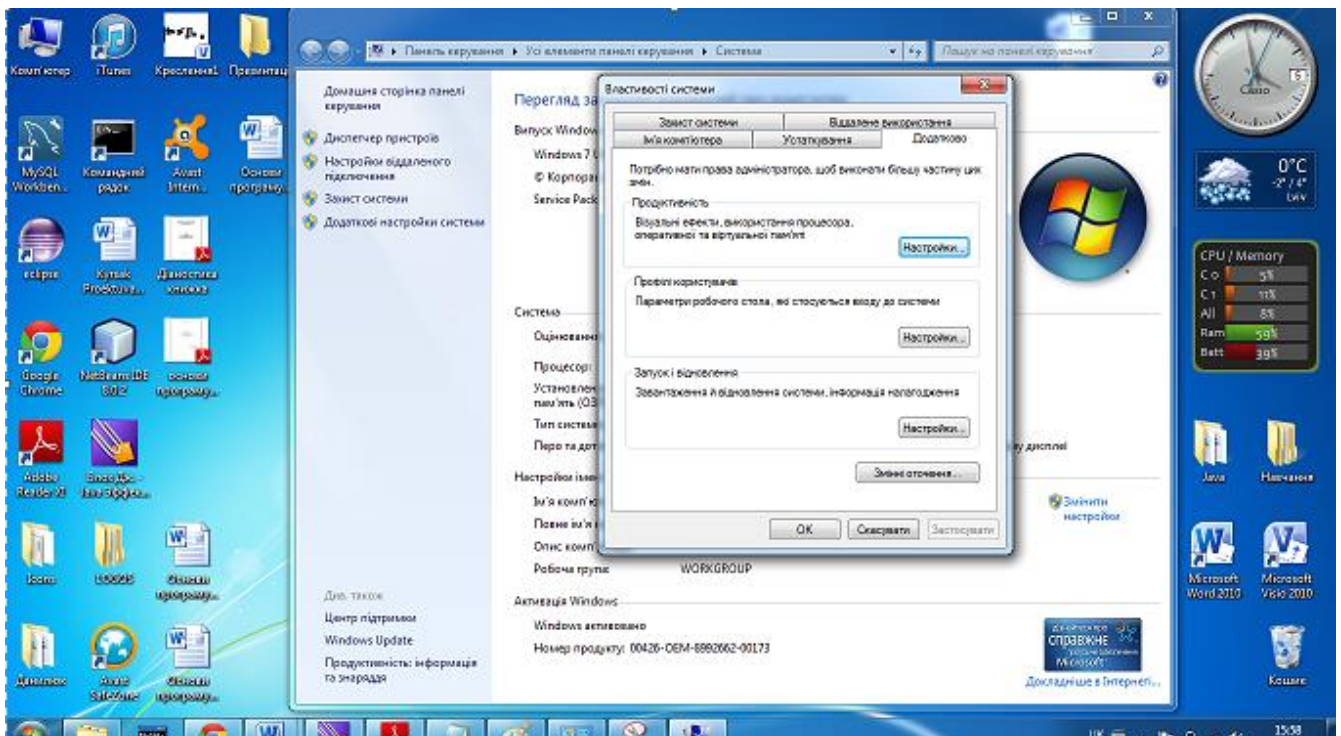


Обираємо властивості

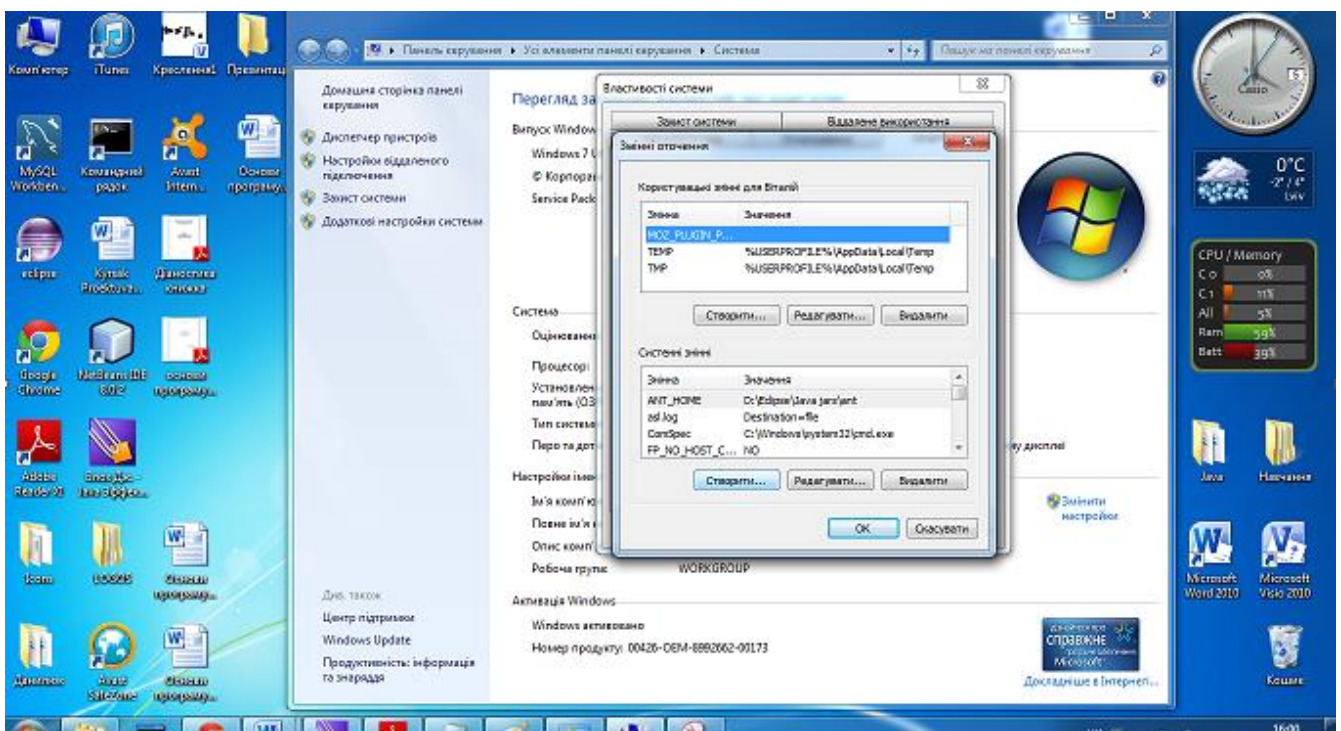


Обираємо «Додаткові настройки системи»

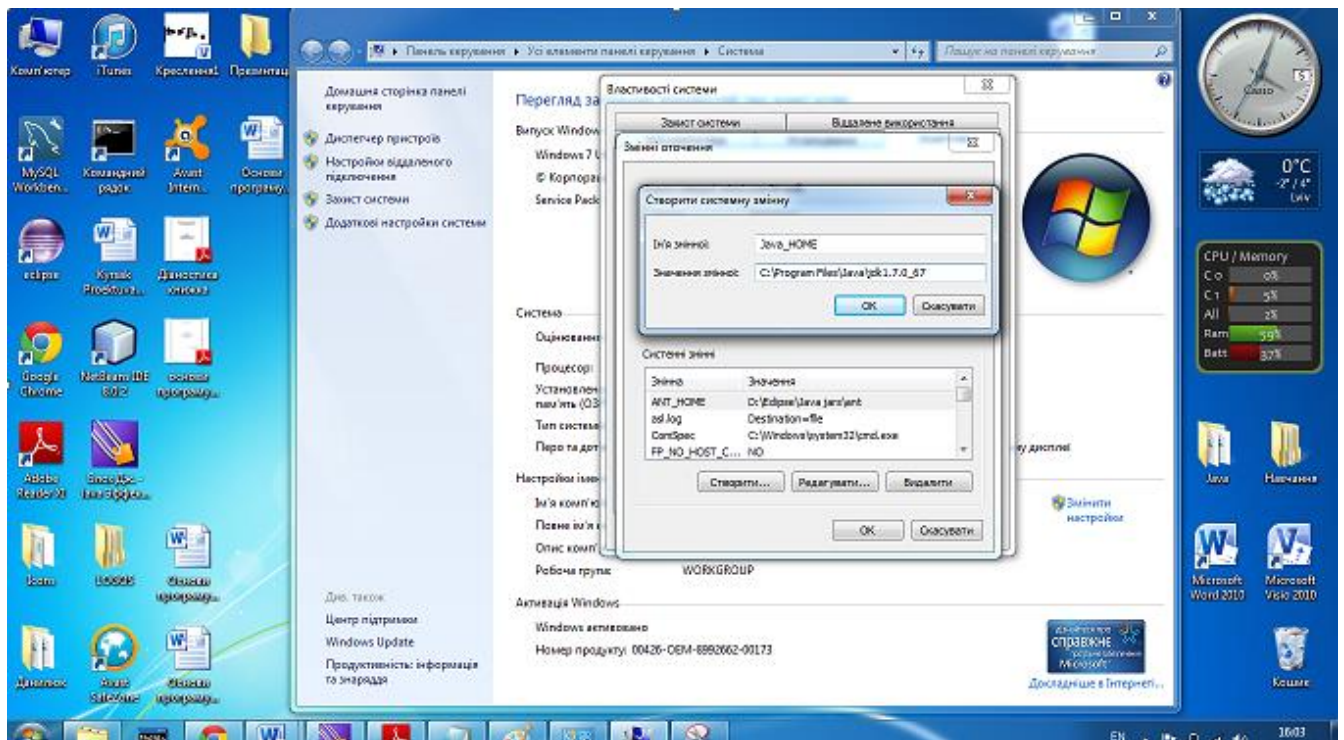




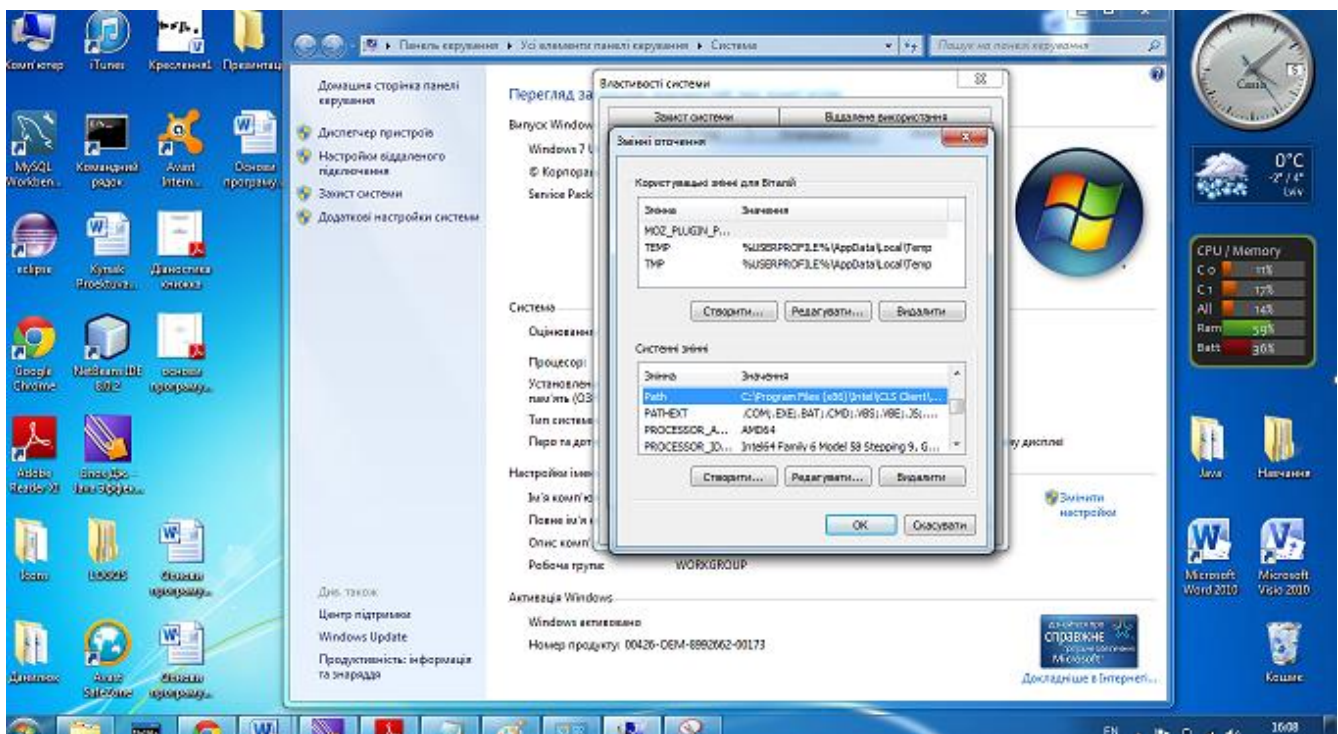
Обираємо «Змінні оточення»



Під пунктом «Системні змінні », натискаємо кнопку «створити»

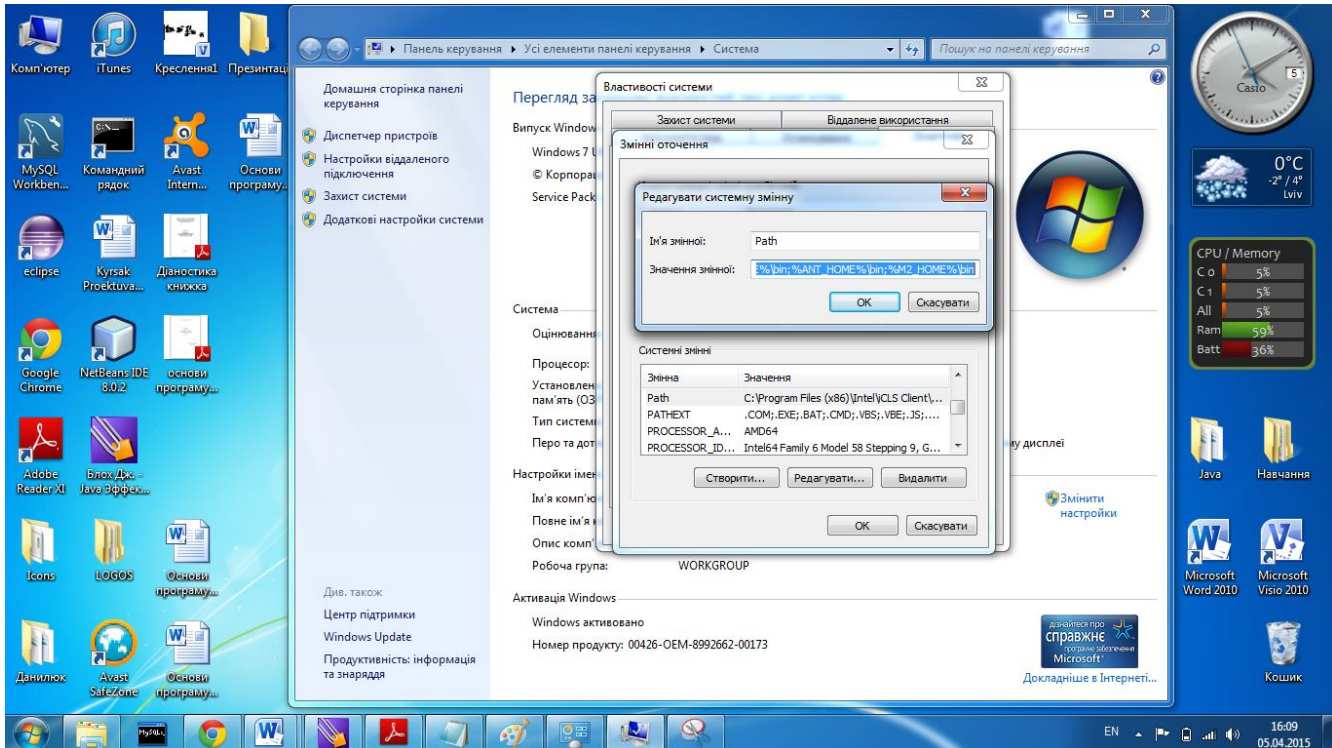


В полі «Ім'я змінної» вводимо назву, яку захотів щоб вона була Java\_Home, в полі «значення змінної» вводимо адресу ношої папки з jdk, в моєму випадку це «C:\Program Files\Java\jdk1.7.0\_67». Натискаємо «OK»

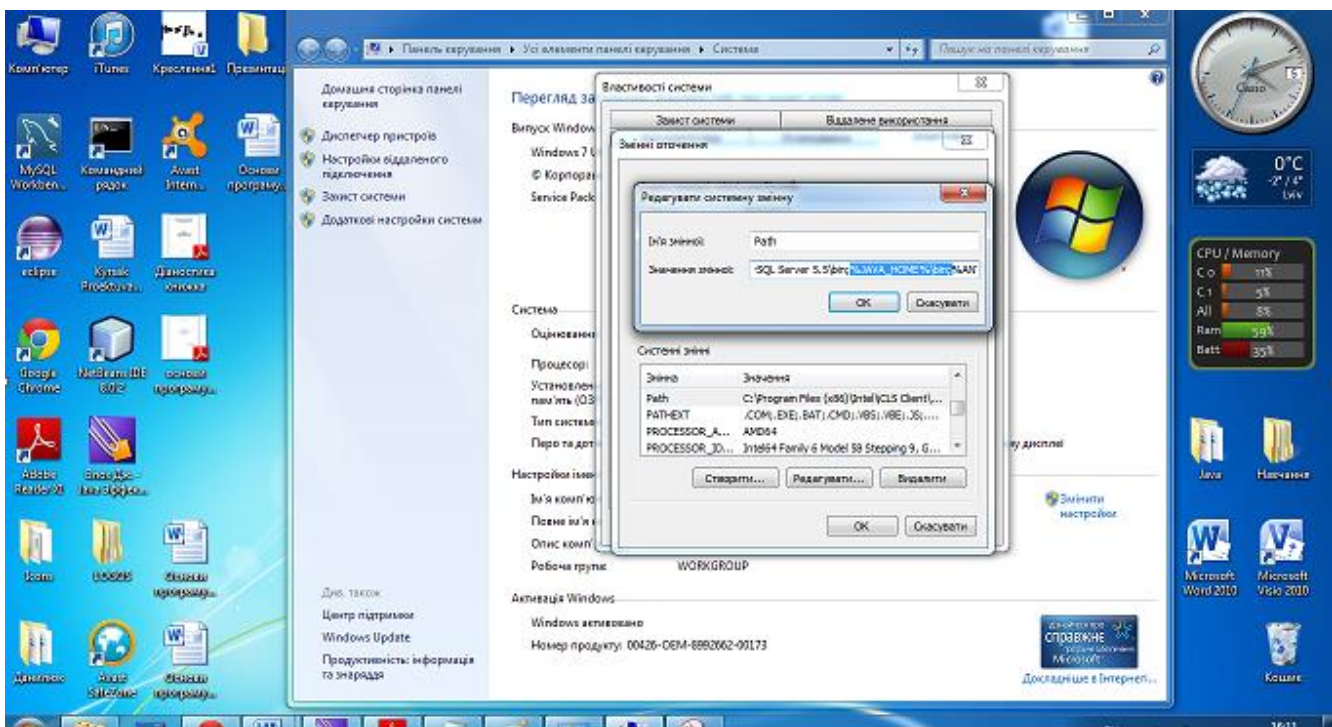


Серед багатьох змінних обираємо змінну «Path».





Обираємо кнопку «Редагувати».



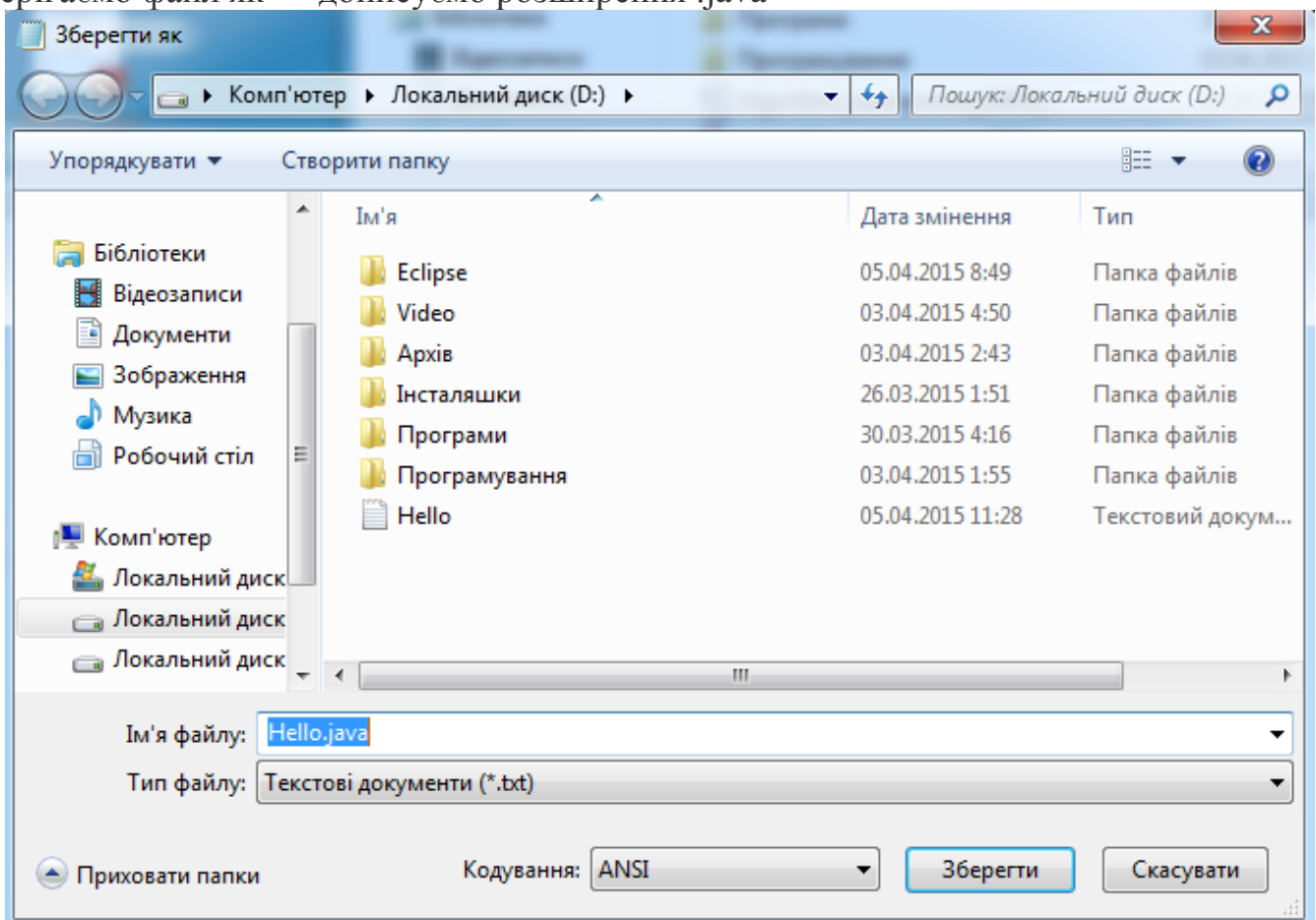
Синтаксис наступний «;%JAVA\_HOME%\bin». Після останньої змінної ставимо крапку з комою і прописуємо нашу змінну, після bin нічого не ставимо. Натискаємо «ОК», потім знову «ОК». Все, тепер необхідно перезавантажити комп'ютер.

4. Створюємо собі файл блокноту, скажімо на диску D:.\.
5. Пишемо в ньому наступний код

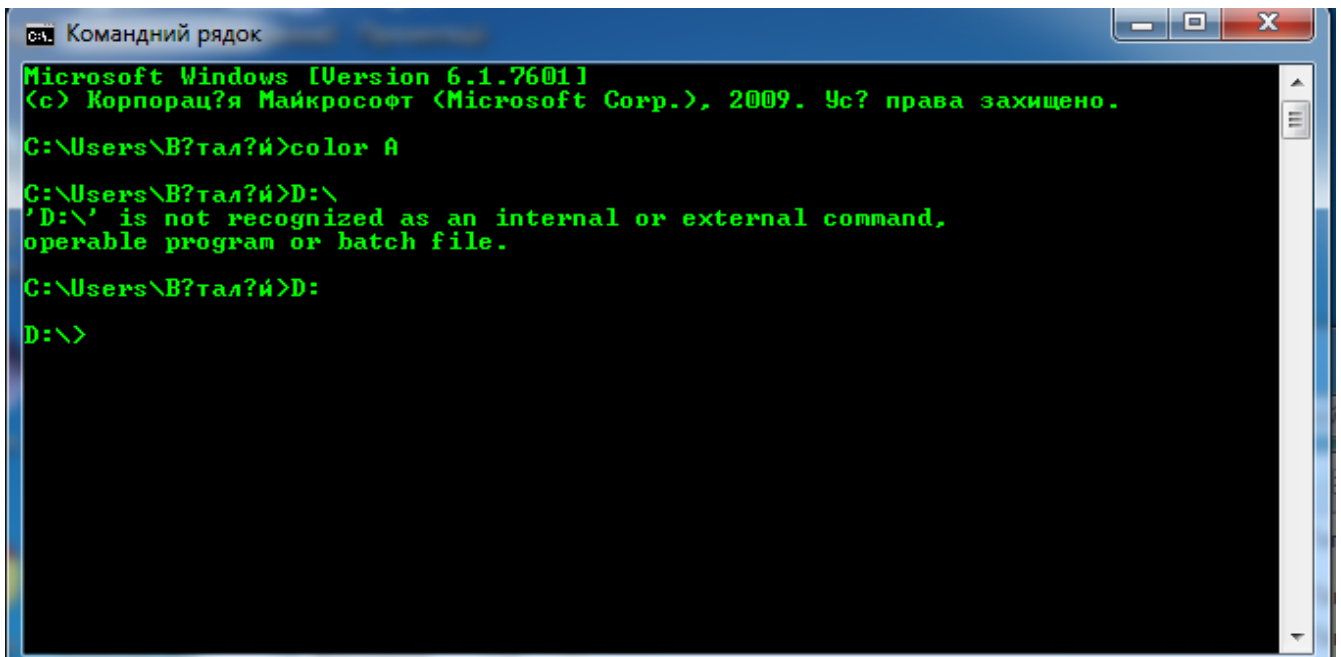


```
1. public class Hello
2. {
3.     public static void main(String [] args)
4.     {
5.         System.out.println("Hello, World!");
6.     }
7. }
```

Зберігаємо файл як--> дописуємо розширення .java

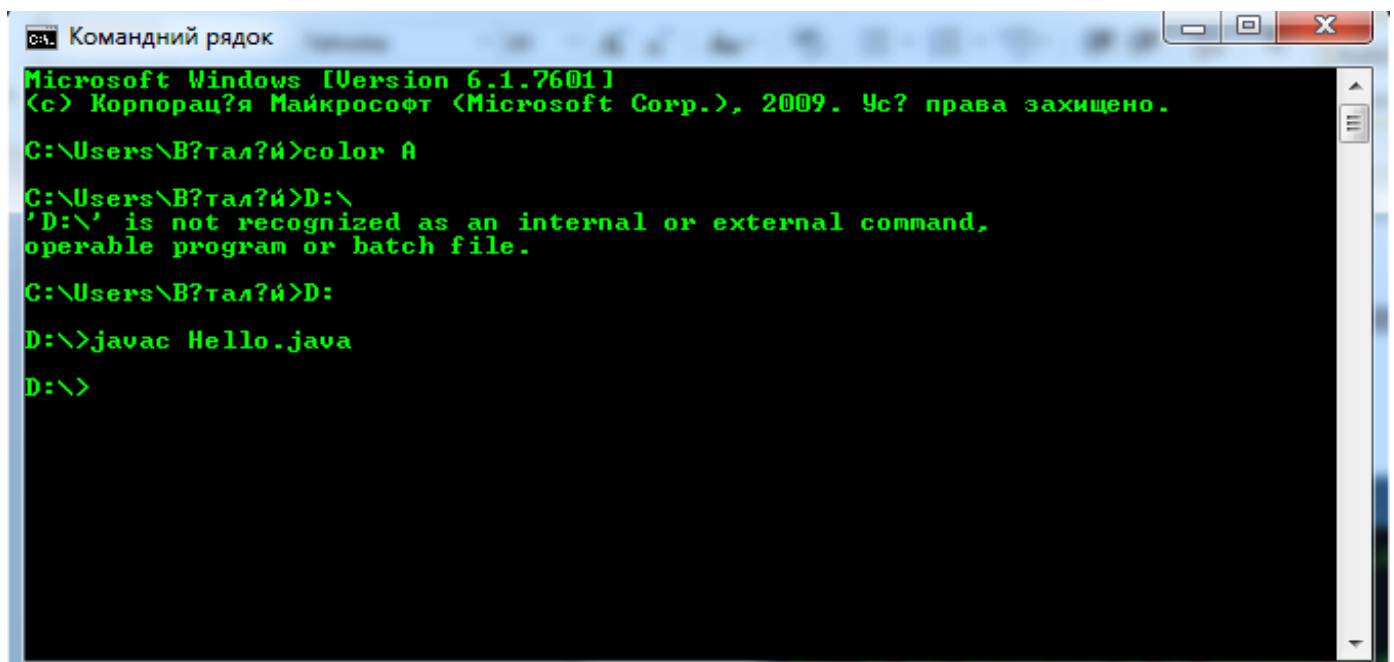


6. Запускаємо командний рядок . Обираємо диск D , командою "D:"



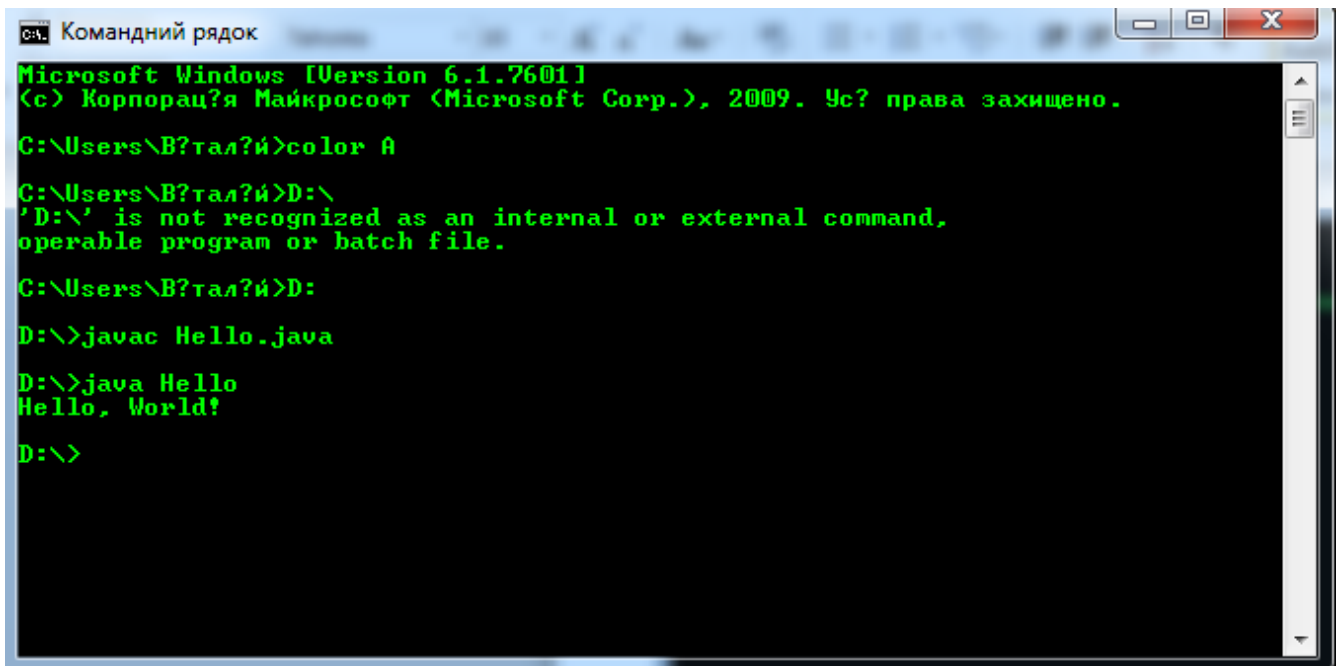
```
Командний рядок
Microsoft Windows [Version 6.1.7601]
(c) Корпорація Майкрософт (Microsoft Corp.), 2009. Усі права захищено.
C:\Users\B?тал?й>color A
C:\Users\B?тал?й>D:\
'D:\' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\B?тал?й>D:
D:\>
```

Пишемо команду `javac` ( для того що скомпілювати наш файл) +пробіл+ назва нашого файлу (Обов'язково щоб назва вашого файлу відповідала назві вашого класу)+ «.java»(додаємо розширення java). + натискаємо Enter



```
Командний рядок
Microsoft Windows [Version 6.1.7601]
(c) Корпорація Майкрософт (Microsoft Corp.), 2009. Усі права захищено.
C:\Users\B?тал?й>color A
C:\Users\B?тал?й>D:\
'D:\' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\B?тал?й>D:
D:\>javac Hello.java
D:\>
```

7.Оскільки компіляція вже відбулася , необхідно провести запуск нашого файлу за допомогою команди «`java`»+ пробіл +`Hello`(назва нашого класу)+Enter



```
Microsoft Windows [Version 6.1.7601]
(c) Корпорація Майкрософт (Microsoft Corp.), 2009. Усі права захищено.
C:\Users\B?тал?й>color A
C:\Users\B?тал?й>D:\
'D:\' is not recognized as an internal or external command,
operable program or batch file.
C:\Users\B?тал?й>D:
D:\>javac Hello.java
D:\>java Hello
Hello, World!
D:\>
```

Як бачимо нам виводиться саме те що ми хотіли .