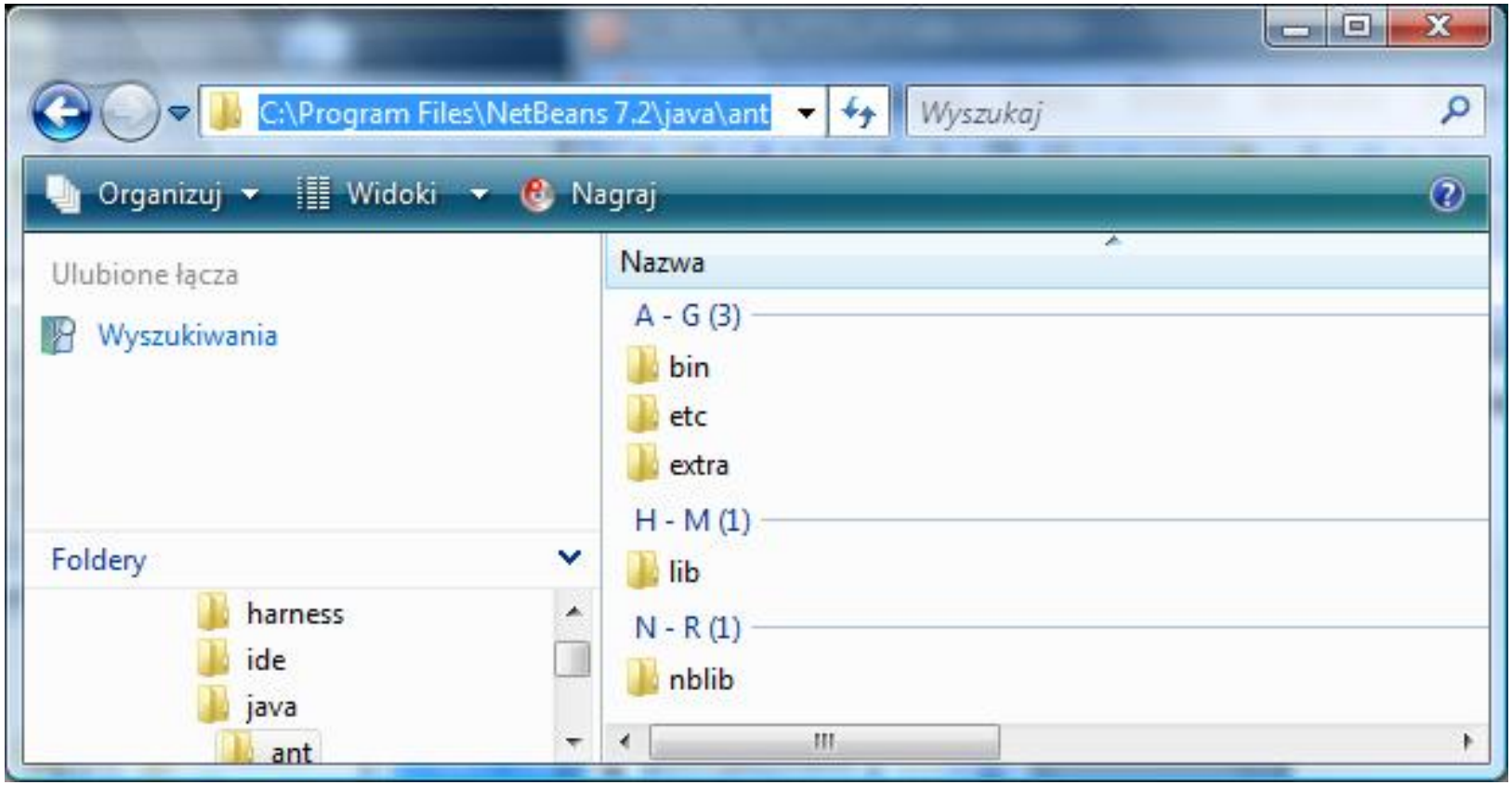
Metryki

Narzędzia do pomiaru złożoności
modułowej i międzymodułowej
oprogramowania

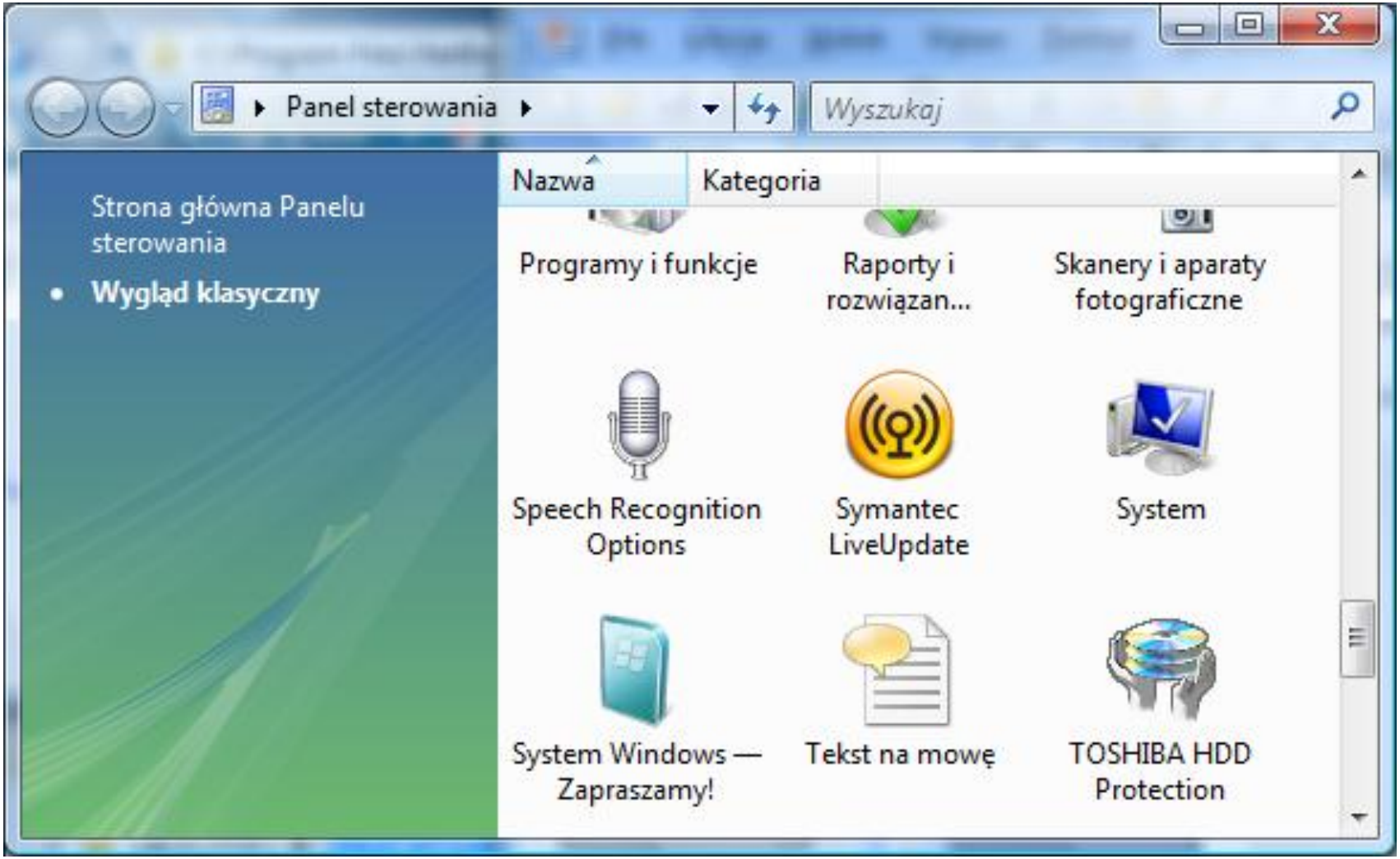
autor: Zofia Kruczkiewicz

Zastosowanie narzędzi **ant i ckjm** do pomiaru złożoności oprogramowania

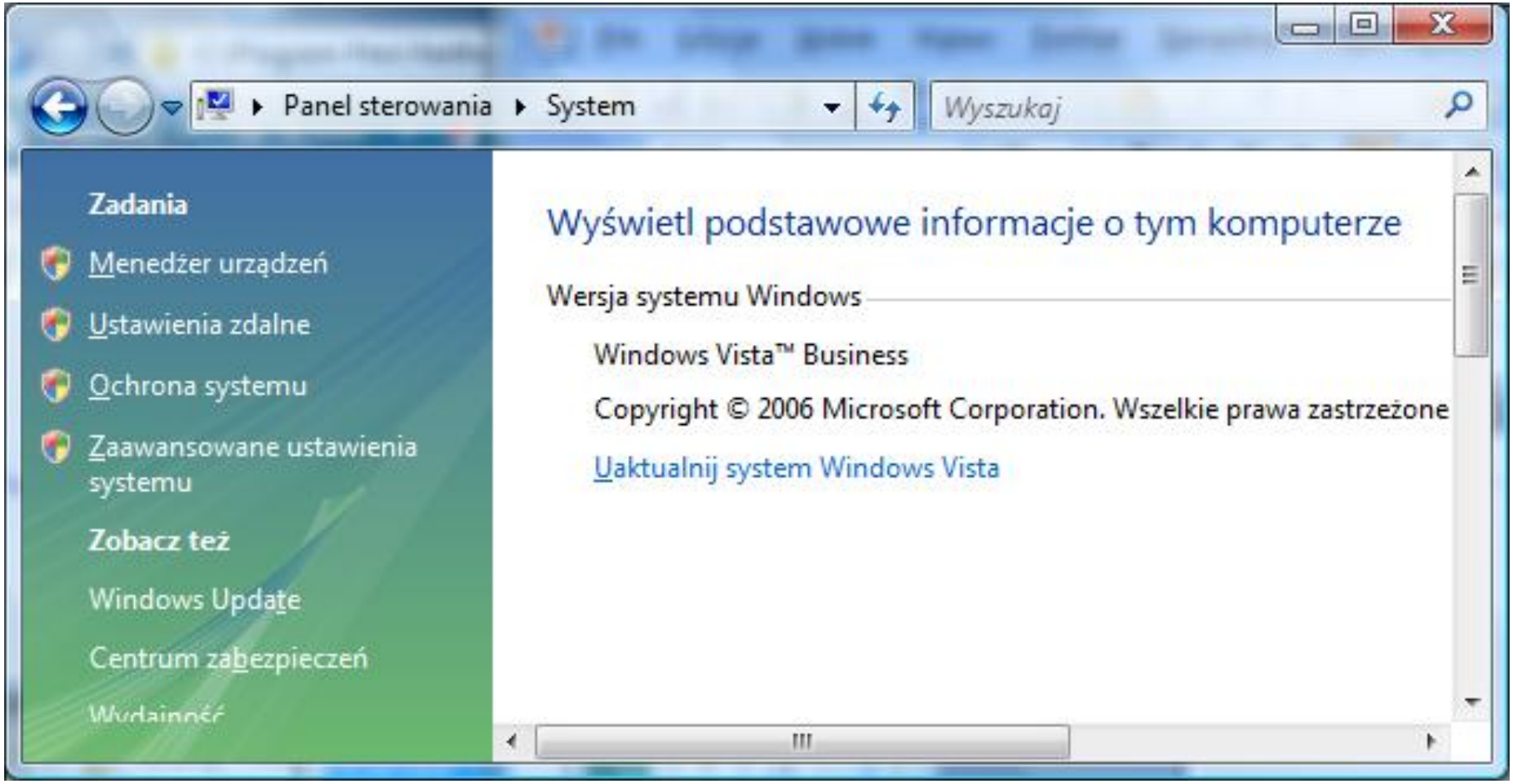
1. Wskazanie ścieżki dostępu do programu ant, umieszczonego w podkatalogu C:\Program Files\NetBeans 7.2\java\ant (instalowanego podczas instalacji NetBeans)



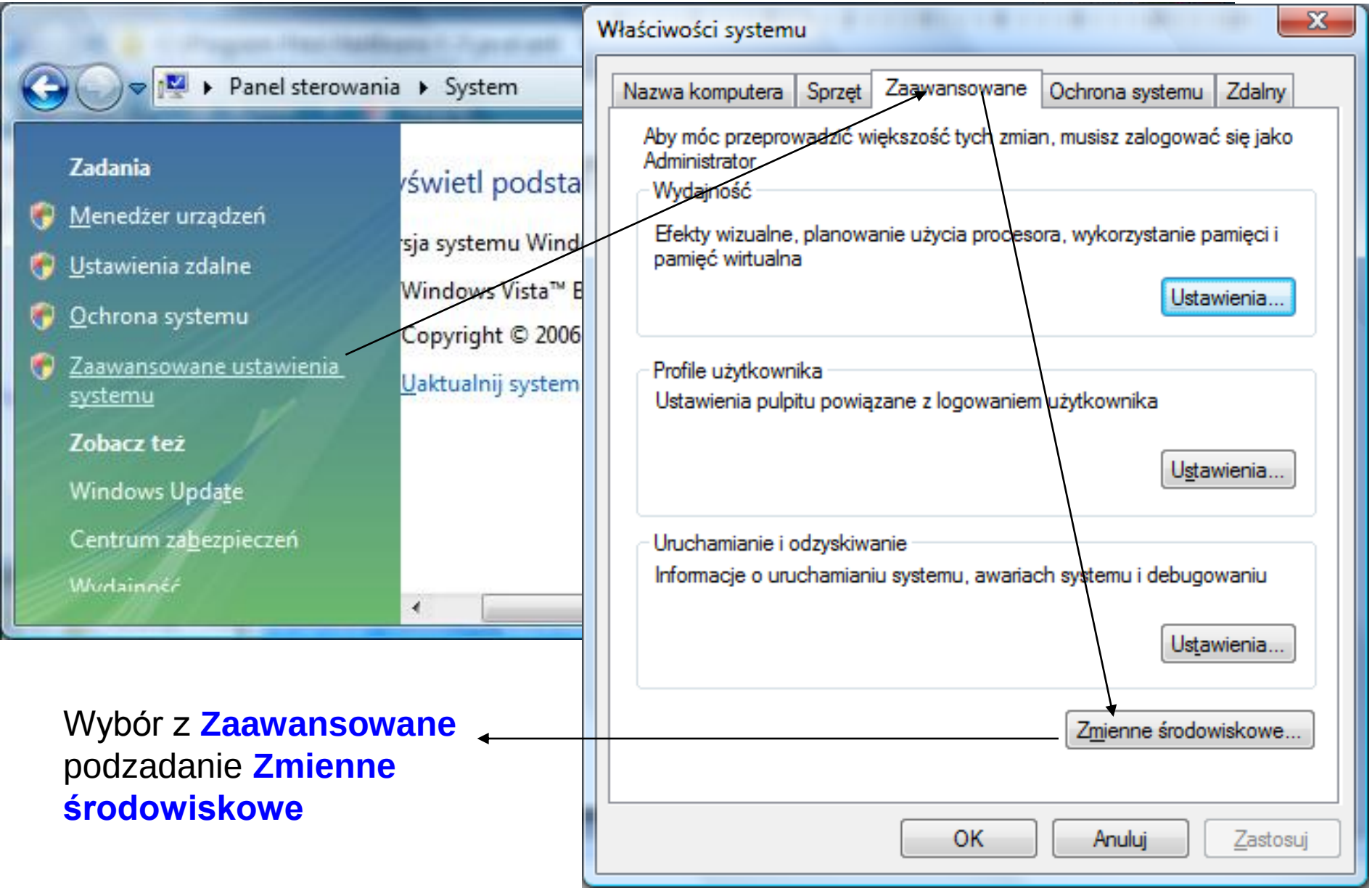
Dostęp do definiowania zmiennych środowiskowych SO Windows w Panelu Sterowania



Przejście w Panelu Sterowania do zadań typu System

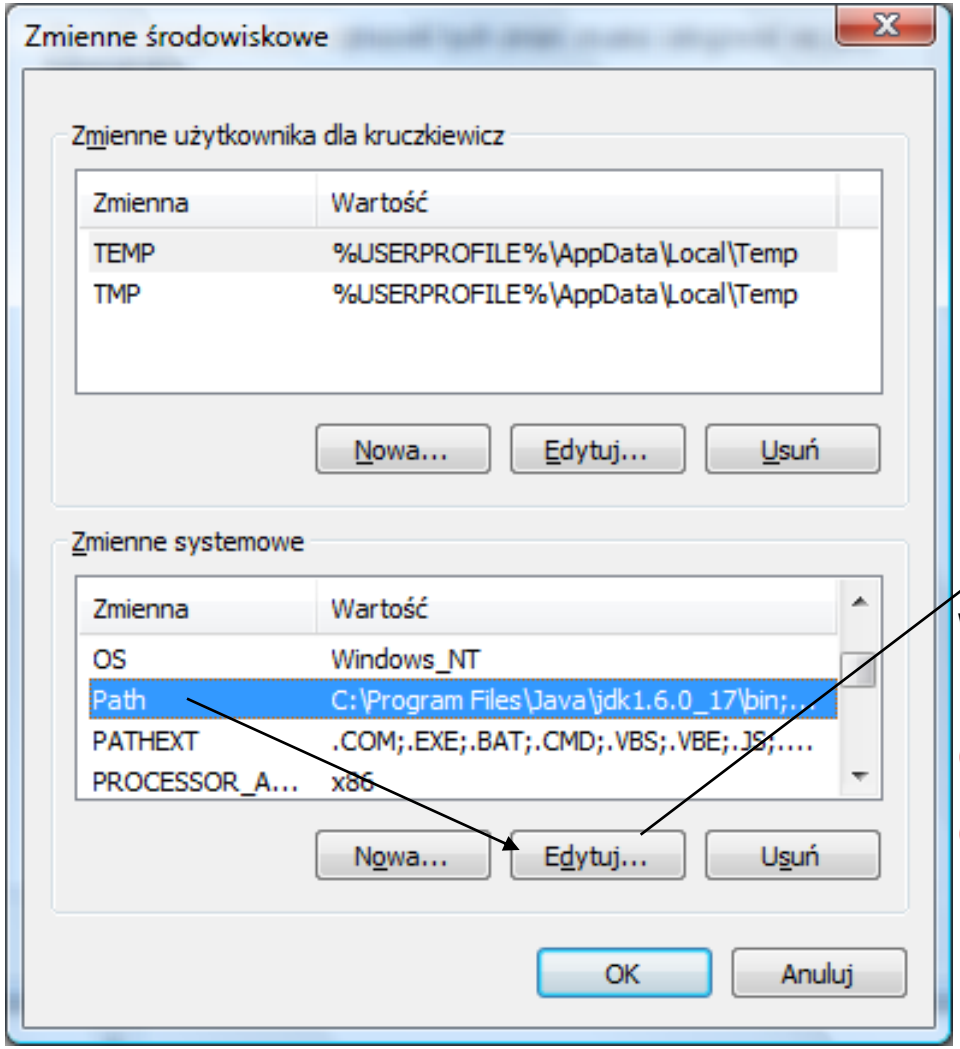


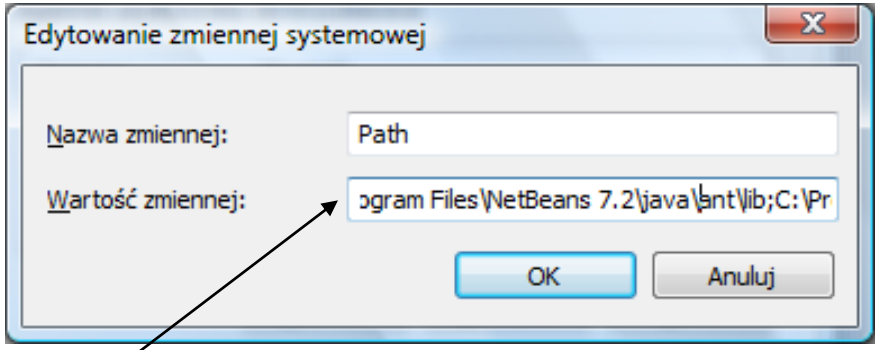
Wybór zadania Zaawansowane ustawienia systemu



Wybór z **Zaawansowane** podzadanie **Zmienne środowiskowe**

Wybór z listy Zmienne systemowe zmiennej Path i kliknięcie na przycisk Edytuj





Wpisanie ścieżek dostępu do katalogów bin i lib w katalogu ant:

C:\Program Files\NetBeans 7.2\java\ant \bin;

C:\Program Files\NetBeans 7.2\java\ant\lib ;

Ustawienie w systemowej zmiennej środowiskowej Path ścieżek do podkatalogów bin i lib podkatalogu ant (poprzedni slajd).

C:\PROGRA~1\Borland\CBUILD~1\Bin;C:\PROGRA~1\Borland\CBUILD~1\Projects\Bpl;%SystemRoot%\system32;%SystemRoot%;%SystemRoot%\System32\Wbem;c:\Program Files\Microsoft SQL Server\90\Tools\binn\; **C:\Program Files\NetBeans 7.2\java\ant \bin; C:\Program Files\NetBeans 7.2\java\ant\lib ;**

2. Instalacja narzędzia ckjm

Linki do strony z programem CKJM

Strona główna:

http://www.spinellis.gr/sw/ckjm

Link do pobrania kodu narzędzia:

http://www.spinellis.gr/sw/ckjm/ckjm-1.9.zip

Opis narzędzia:

http://www.spinellis.gr/sw/ckjm/doc/indexw.html

Budowa skryptu build.xml dla programu ant, zawierającego informacje, gdzie znajduje się narzędzie ckjm, jaki program należy zmierzyć oraz sposób prezentacji wyników pomiaru

Utworzenie i wywołanie skryptu build.xml

- Należy napisać skrypt build.xml wg wzoru podanego na następnym slajdzie
- Należy umieścić skrypt build.xml w wybranym katalogu,
- Należy wywołać program **ant** w wybranym katalogu, gdzie znajduje się plik **build.xml**
- Wynik pomiaru w postaci pliku typu html warto umieszczać w wybranym katalogu, gdzie znajduje się skrypt build.xml. W skrypcie podanym na następnej stronie w znaczniku <**ckjm outputfile="ckjm.xml" format="xml"** podano, że plik **ckjm.xml** zawiera wyniki pomiaru. W znaczniku **<xslt>** ten plik jest wskazany w atrybucie **in**, jako dane wejściowe przekształcone na podstawie wskazanego arkusza stylu **ckjm_extra.xsl** w atrybucie **style** na plik **ckjm.html** podany przez atrtybut **out**. Brak ścieżki oznacza generowanie wyniku w katalogu, gdzie znajduje się skrypt build.xml

Przykład skryptu build.xml wywołanego domyślnie przez program ant dla programu typu Java Application

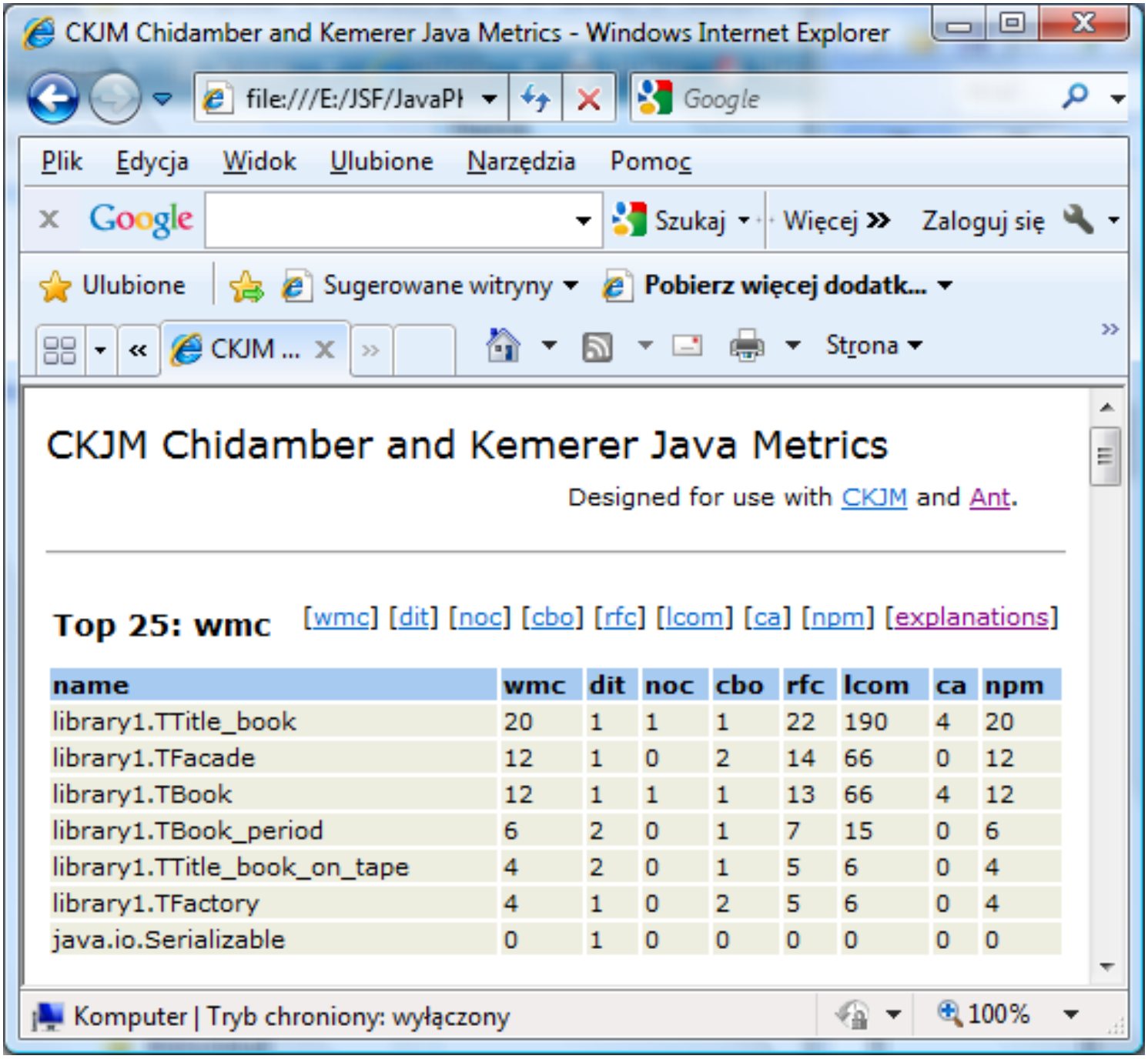
```
<?xml version="1.0" encoding="UTF-8"?>
<project name="myproject" default="ckjm">

<target name="compile">
  <!-- your compile instructions -->
</target>

  <target name="ckjm" depends="compile">
    <taskdef name="ckjm" classname="gr.spinellis.ckjm.ant.CkjmTask">
      <classpath>
        <pathelement location="C:/downloads/ckjm-1.9/ckjm-1.9/build/ckjm-1.9.jar"/>
      </classpath>
    </taskdef>

    <ckjm outputfile="ckjm.xml" format="xml"
      classdir="E:\EnglishLecture\Laboratory\BT_Library1-Model2\BT_Library1\build\classes">
      <include name="**/*.class" />
      <exclude name="**/*Test.class" />

      <extdirs path="E:\EnglishLecture\Laboratory\BT_Library1-Model2\BT_Library1\dist"/>
    </ckjm>

    <xslt in="ckjm.xml" style="C:/downloads/ckjm-1.9/ckjm-1.9/xsl/ckjm_extra.xsl" out="ckjm.html" />
  </target>
</project>
```

<!– classname zawiera nazwę klasy głównej narzędzia ckjm, która jest wskazana za pomocą podanej ścieżki w znaczniku pathelement-->

<!-- classdir lokalizuje pakiety z bajkodami uzywanymi przez projekt za pomocą podanej ścieżki – w przykładzie classes jest katalogiem, w ktorym sa umieszczane pakiety z plikami (*.class)-->

Mierzone są pliki .class, wyłączone z pomiarów są pliki *Test.class

<!-- extdirs lokalizuje biblioteki uzywane przez klasy projektu za pomocą podanej ścieżki - w przykładzie dist jest katalogiem, w ktorym są umieszczone spakowane pakiety bibliotek (*.jar)-
Można dodać kolejne znaczniki extdirs zawierajace różne ścieżki do pozostałych bibliotek używanych przez klasy aplikacji->



Przykład pomiaru metryk wg skryptu z poprzedniej strony

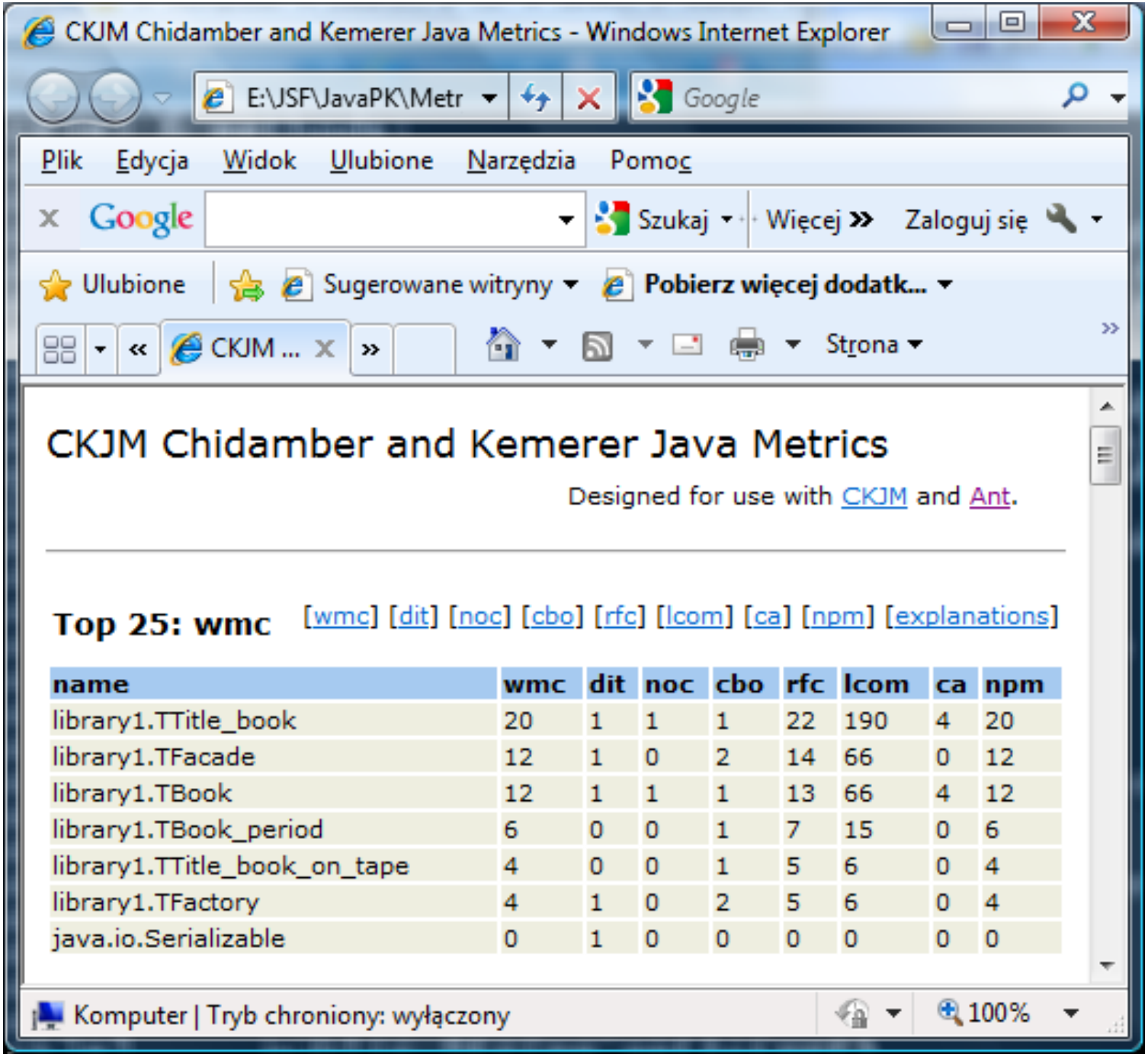
Przykład skryptu build.xml wywołanego domyślnie przez program ant dla programu typu Java Application

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="myproject" default="ckjm">

<target name="compile">
  <!-- your compile instructions -->
</target>

  <target name="ckjm" depends="compile">
    <taskdef name="ckjm" classname="gr.spinellis.ckjm.ant.CkjmTask">
      <classpath>
        <pathelement location="C:/downloads/ckjm-1.9/ckjm-1.9/build/ckjm-1.9.jar"/>
      </classpath>
    </taskdef>

   <ckjm outputfile="ckjm.xml" format="xml"
     classdir="E:\EnglishLecture\Laboratory\BT_Library1-Model2\BT_Library1\build\classes">
     <include name="**/*.class" />
     <exclude name="**/*Test.class" />

    <!--<extdirs path="E:\EnglishLecture\Laboratory\BT_Library1-Model2\BT_Library1\dist"/>→
   </ckjm>

   <xslt in="ckjm.xml" style="C:/downloads/ckjm-1.9/ckjm-1.9/xsl/ckjm_extra.xsl" out="ckjm.html" />
  </target>
</project>
```

<!– classname zawiera nazwę klasy głównej narzędzia ckjm, która jest wskazana za pomocą podanej ścieżki w znaczniku pathelement-->

<!-- classdir lokalizuje pakiety z bajkodami uzywanymi przez projekt za pomocą podanej ścieżki – w przykładzie classes jest katalogiem, w ktorym sa umieszczane pakiety z plikami (*.class)-->

<!-- extdirs lokalizuje biblioteki uzywane przez klasy projektu za pomocą podanej ścieżki - w przykładzie dist jest katalogiem, w ktorym są umieszczone spakowane pakiety bibliotek (*.jar)-
Można dodać kolejne znaczniki extdirs zawierajace różne ścieżki do pozostałych bibliotek używanych przez klasy aplikacji->



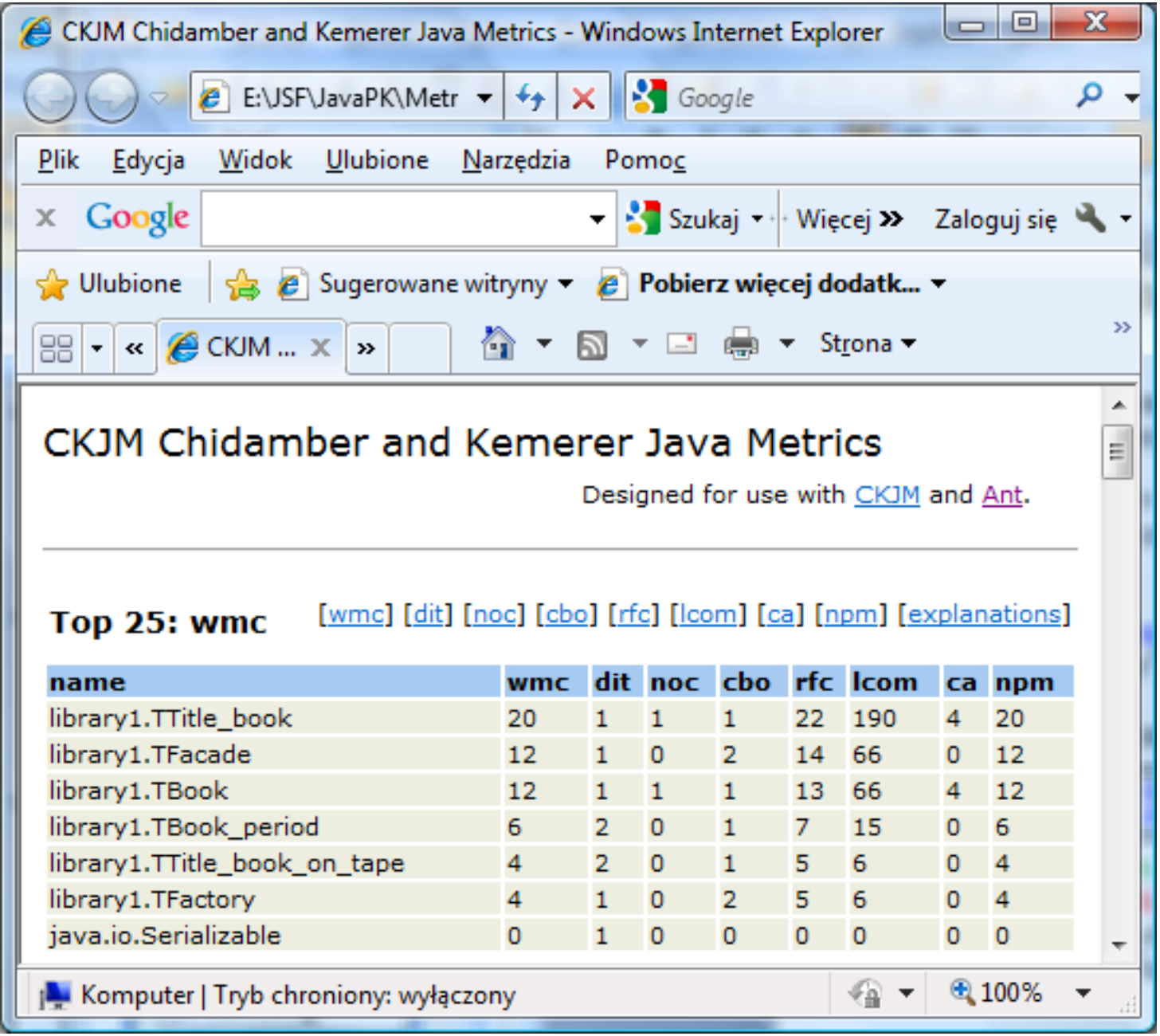
Przykład pomiaru metryk wg skryptu z poprzedniego slajdu, przy usuniętym znaczniku **extdirs**, która wskazywała klasy bazowe – wynik pomiaru głębokości dziedziczenia **dit** zawiera błędy (wartość 0), gdyż najmniejsza głębokość dziedziczenia w Javie to 1 (dziedziczenie po klasie Object)

Przykład skryptu build.xml wywołanego domyślnie przez program ant dla programu typu Java Application- zalecane dla pozostałych typów aplikacji

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="myproject" default="ckjm">

<target name="compile">
  <!-- your compile instructions -->
</target>

  <target name="ckjm" depends="compile">
     <taskdef name="ckjm" classname="gr.spinellis.ckjm.ant.CkjmTask">
        <classpath>
          <pathelement location="C:/downloads/ckjm-1.9/ckjm-1.9/build/ckjm-1.9.jar"/>
        </classpath>
     </taskdef>

    <ckjm outputfile="ckjm.xml" format="xml"
          classdir="E:\EnglishLecture\Laboratory\BT_Library1-Model2\BT_Library1\dist">
      <include name="**/*.class" />
      <exclude name="**/*Test.class" />
    </ckjm>

    <xslt in="ckjm.xml" style="C:/downloads/ckjm-1.9/ckjm-1.9/xsl/ckjm_extra.xsl"
      out="ckjm.html" />
  </target>
</project>
```

<!– classname zawiera nazwę klasy głównej narzędzia ckjm, która jest wskazana za pomocą podanej ścieżki w znaczniku pathelement-->

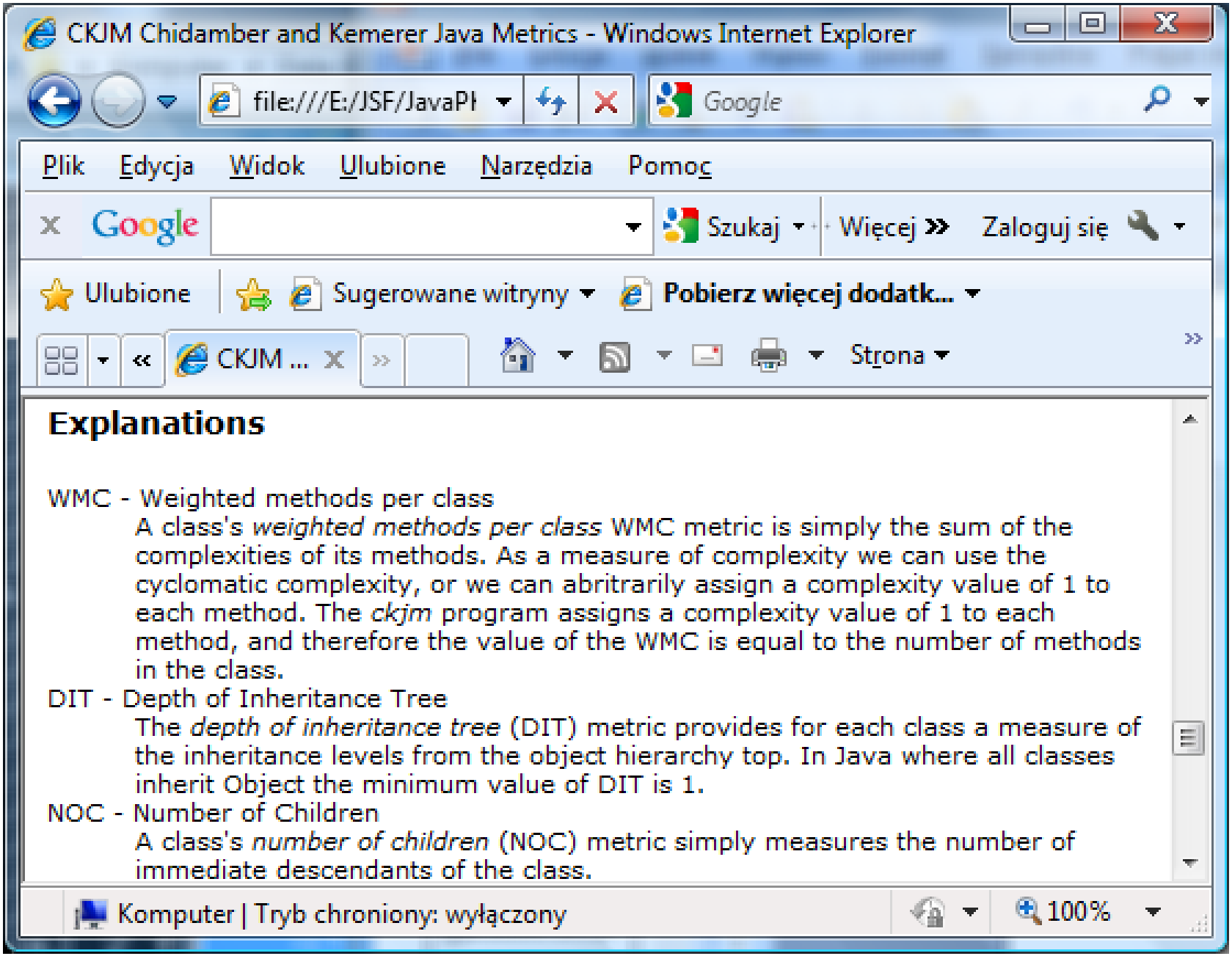
<!-- classdir lokalizuje wszystkie pakiety z bajkodami uzywanymi przez projekt za pomocą podanej ścieżki – w przykładzie dist jest katalogiem, w ktorym są umieszczane spakowane pakiety z plikami (*.lar)-->



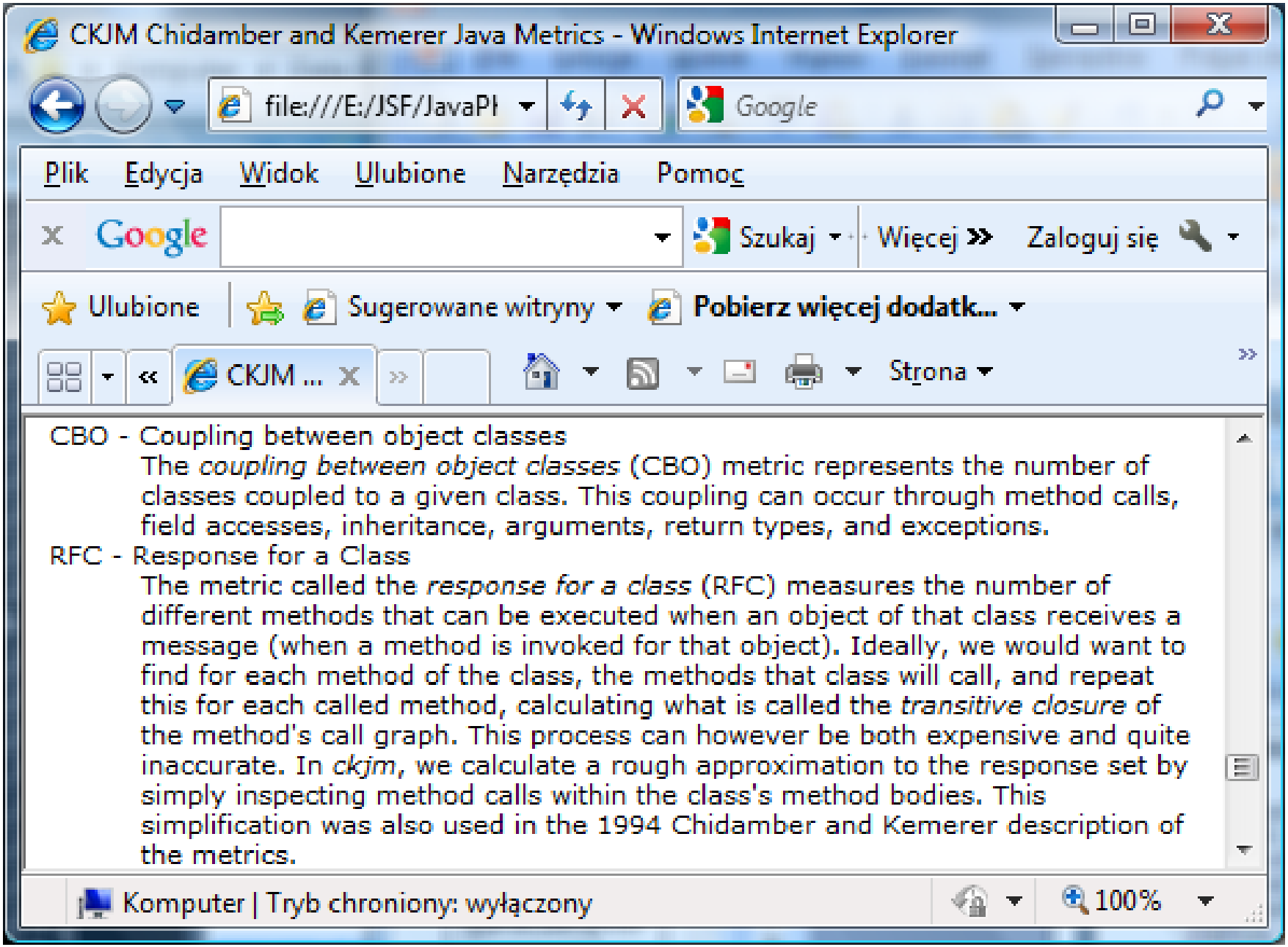
name	wmc	dit	noc	cbo	rfc	lcom	ca	npm
library1.TTitle_book	20	1	1	1	22	190	4	20
library1.TFacade	12	1	0	2	14	66	0	12
library1.TBook	12	1	1	1	13	66	4	12
library1.TBook_period	6	2	0	1	7	15	0	6
library1.TTitle_book_on_tape	4	2	0	1	5	6	0	4
library1.TFactory	4	1	0	2	5	6	0	4
java.io.Serializable	0	1	0	0	0	0	0	0

Poprawne wyniki pomiarów, gdy w ścieżce zdefiniowanej w atrybucie **classdir** wskazano wszystkie możliwe biblioteki wykorzystywane w aplikacji (skrypt **build.xml** z poprzedniego slajdu)

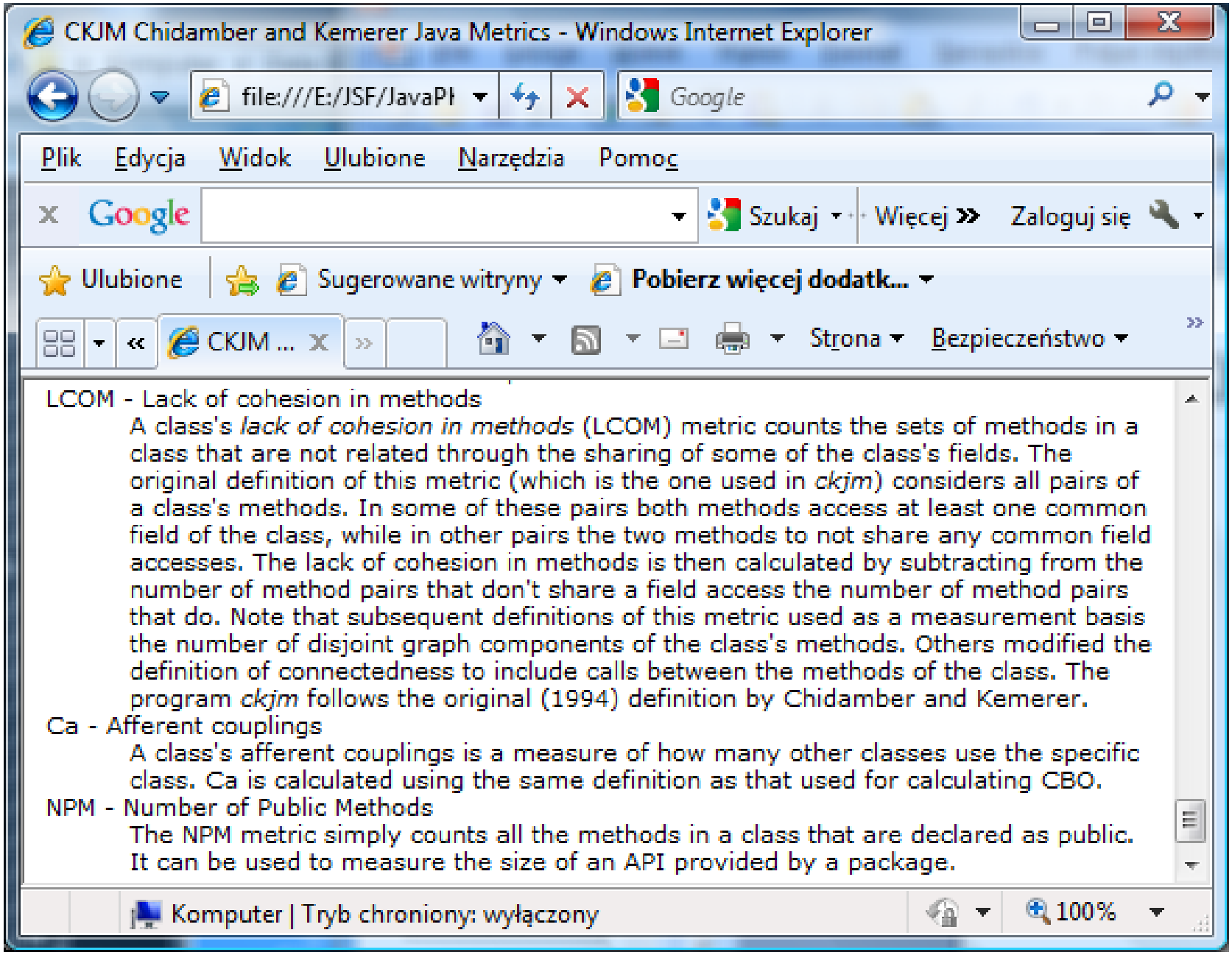
Wybór zakładki Explanations zawierającej definicje metryk



Wybór zakładki Explanations zawierającej definicje metryk (cd)



Wybór zakładki Explanations zawierającej definicje metryk (cd)



Przykład metryk CK trzech systemów

System analyzed	Java	Java	C++
Classes	46	1000	1617
Lines	50,000	300,000	500,000
Quality	"Low"	"High"	"Medium"
CBO	2.48	1.25	2.09
LCOM1	447.65	78.34	113.94
RFC	80.39	43.84	28.60
NOC	0.07	0.35	0.39
DIT	0.37	0.97	1.02
WMC	45.7	11.10	23.97