



semestr zimowy 2016/2017

Michał Cieśla ([michal.ciesla@uj.edu.pl](mailto:michal.ciesla@uj.edu.pl))

<http://users.uj.edu.pl/~ciesla/>

Konsultacje: środa 10-12, pokój D-2-47

# JĘZYK JAVA

## ZAGADNIENIA:

- podstawy;
- przegląd biblioteki standardowej;
- Java w zastosowaniach.

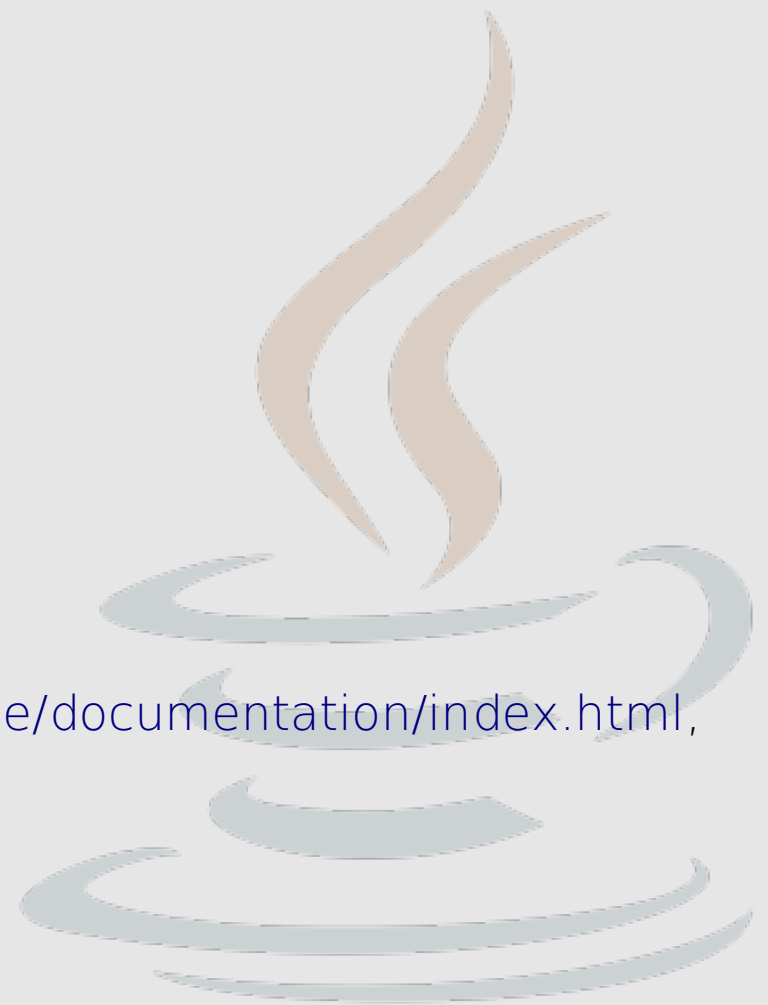
## LITERATURA:

<http://www.oracle.com/technetwork/java/javase/documentation/index.html>,

Coursera, iTunesU, itp.

<http://www.google.com/>,

Bruce Eckel, *Thinking in Java*.



# JAVA



*Java is one of several Indonesian islands that grow coffee. Originally, the term "Java coffee" or "Kopi jaw" identified the dark Arabica coffee specific to Java. In American slang, "Java" came to mean coffee in general .*

**By Tamasin Wedgwood, eHow Contributor**

Read more: [Why Is Coffee Called Java? | eHow.com](http://www.ehow.com/why-is-coffee-called-java/)

# PREREKWIZYTY

JDK – Java Development Kit

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

JRE – Java Runtime Environment

<http://www.java.com>

lub

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>



# HELLO WORLD

HelloWorld.java

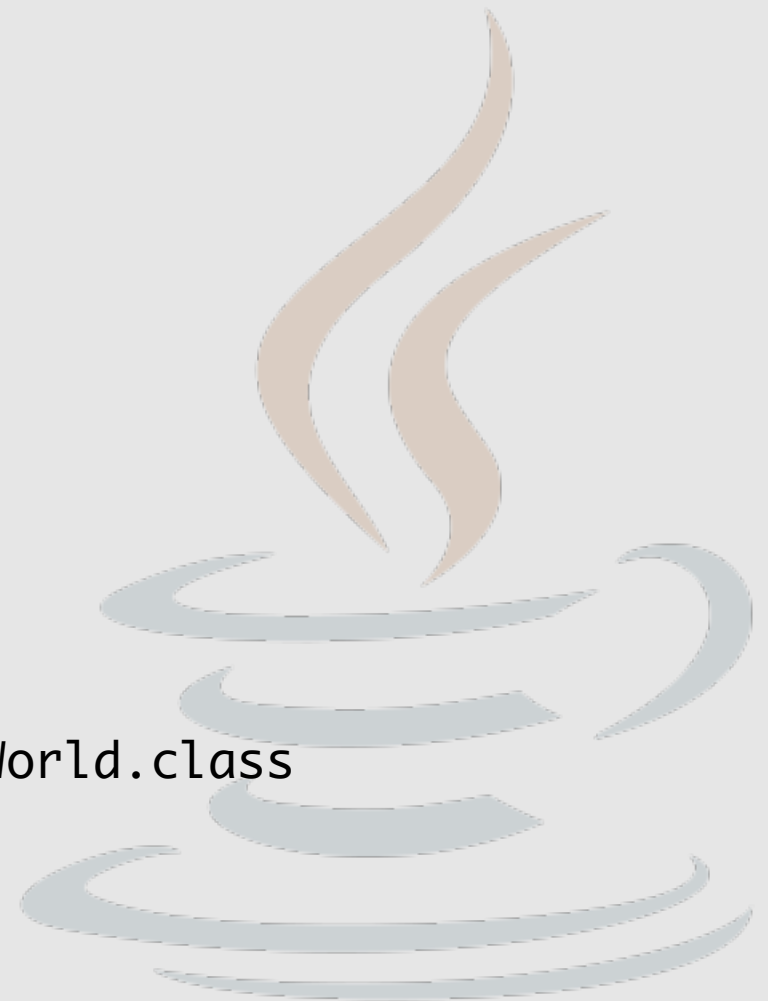
```
public class HelloWorld {  
    public static void main(String[] args){  
        System.out.println("Hello World!");  
    }  
}
```

KOMPILACJA:

`javac HelloWorld.java` → `HelloWorld.class`

URUCHOMIENIE:

`java HelloWorldConsole`



# PODSTAWY: TYPY DANYCH

PODSTAWY JĘZYKA JAVA:

<http://docs.oracle.com/javase/tutorial/java/nutsandbolts/index.html>

PRYMITYWNE TYPY DANYCH:

- **byte** (8-bit), **short** (16-bit), **int** (32-bit), **long** (64-bit)
- **float** (32-bit), **double** (64-bit),
- **boolean** (1-bit) - flaga
- **char** (16-bit) – znak w unikodzie, np. `\u015b`

OBIEKTOWE TYPY DANYCH:

- **String**, **PrintStream**, ... (wszystko inne).

# PRZEPIŁYW STEROWANIA

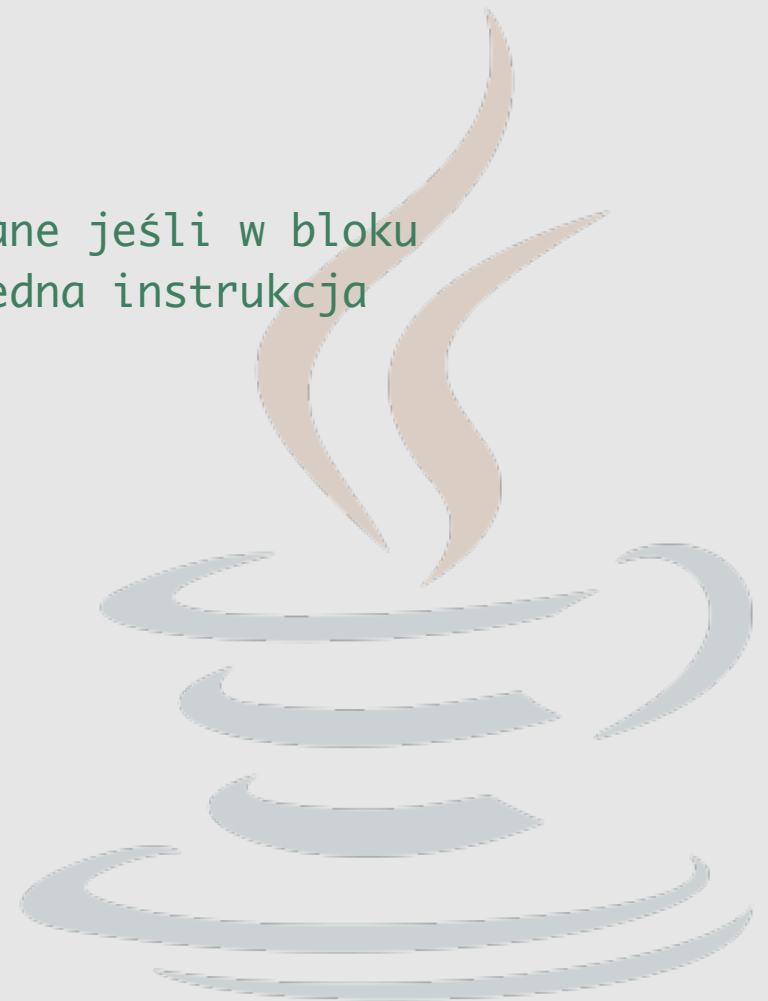
## INSTRUKCJE WARUNKOWE

- if...then...else...

```
if(a>0){           // nawiasy klamrowe są wymagane jeśli w bloku
    return 1;      // znajduje się więcej niż jedna instrukcja
}else{             // tak samo jak w C/C++
    return -1;
}
```

- switch

```
switch (a){
    case 1: makeSomething(a);
            break;
    case 2: makeSomethingElse(a);
    default: a++;
}
```



# PRZEPLÝW STEROWANIA

PĘTLE:

- for.

```
for(i=0; i<args.length; i++)  
    System.out.printf(Locale.US, "%.2f\n", args[i]);
```

- while

```
String s="Ala";  
while(s.length()<20)  
    s = " " + s;
```

- do...while

```
do{  
    String s = getValue();  
}while(s!=null);
```





# PRZEPEŁYW STEROWANIA

ZABURZENIA PRZEPEŁYWU:

- break, continue, return

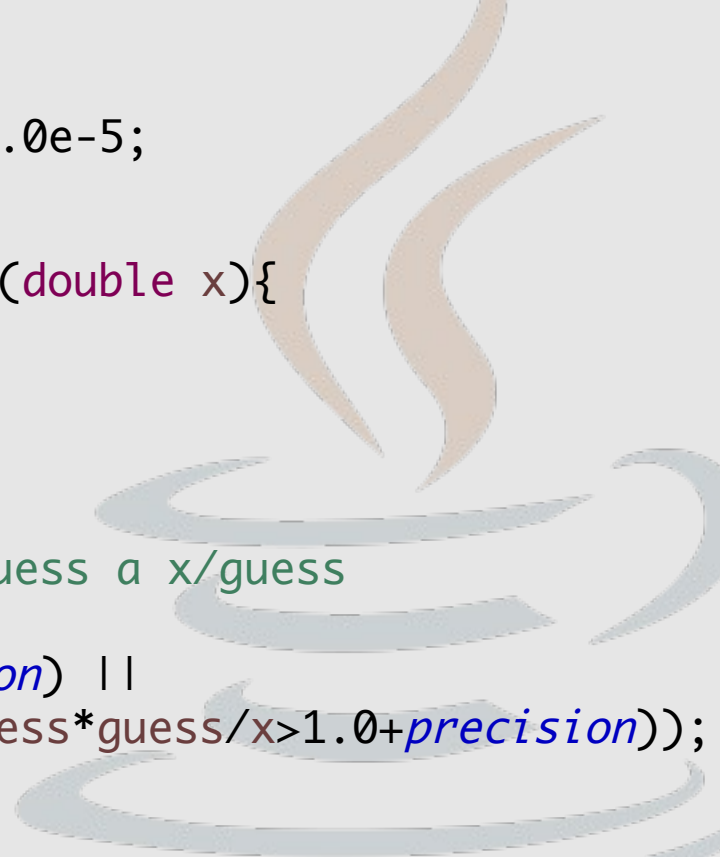
```
String[] names = getNames();
for(int i=0; i<names.length; i++){
    if (names[i].equals("JAVA")){
        found = true;           // znalezlismy i nie musimy dalej szukać
        break;
    }
}
```

```
File[] f = dir.listFiles();
for(int i=0; i<f.length; i++){
    if (f[i].isDirectory())    // chemy wypisac tylko nazwy plikow
        continue;
    System.out.println(f[i].getName());
}
```

# PRZYKŁADY

SquareRoot.java

```
public class SquareRoot {  
  
    public static final double precision = 1.0e-5;  
  
    public static double calculateSquareRoot(double x){  
  
        double guess = 1.0;  
  
        do{ // pierwiastek jest pomiędzy guess a x/guess  
            guess = (guess + x/guess)/2.0;  
        }while( (guess*guess/x < 1.0-precision) ||  
                (guess*guess/x>1.0+precision));  
  
        return guess;  
    }  
}
```



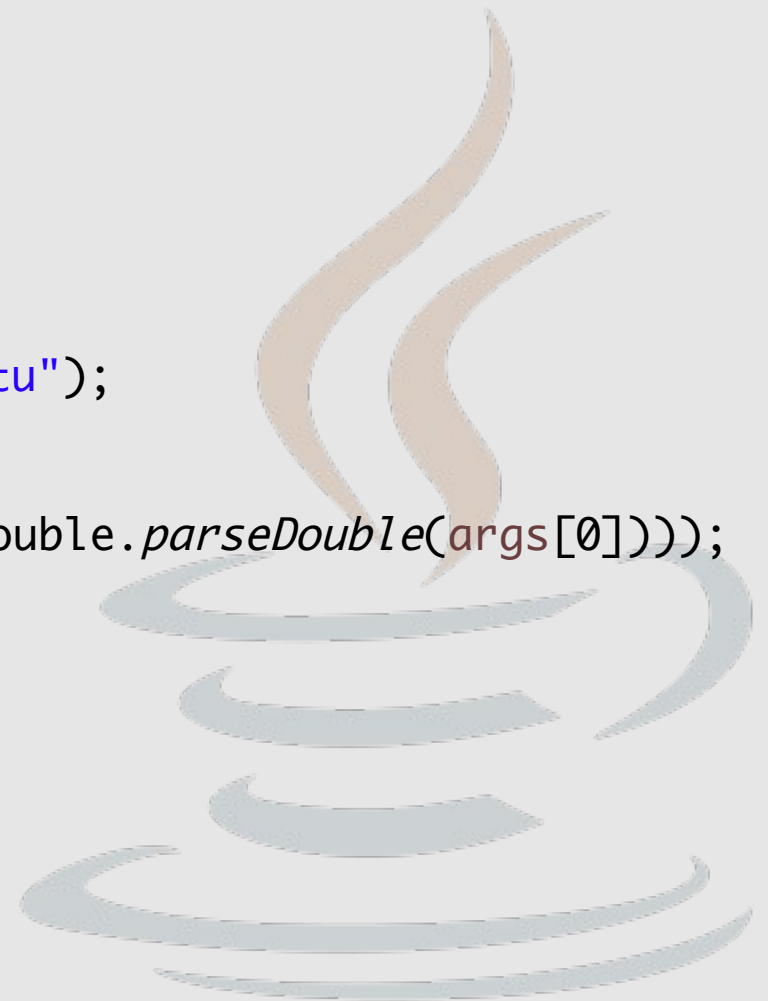
# PRZYKŁADY

SquareRoot.java (c.d)

```
public static void main(String[] args){
    if (args.length<1)
        System.out.println("Brak argumentu");
    else
        System.out.println(
            calculateSquareRoot(Double.parseDouble(args[0])));
}
}
```

URUCHOMIENIE (po skompilowaniu):

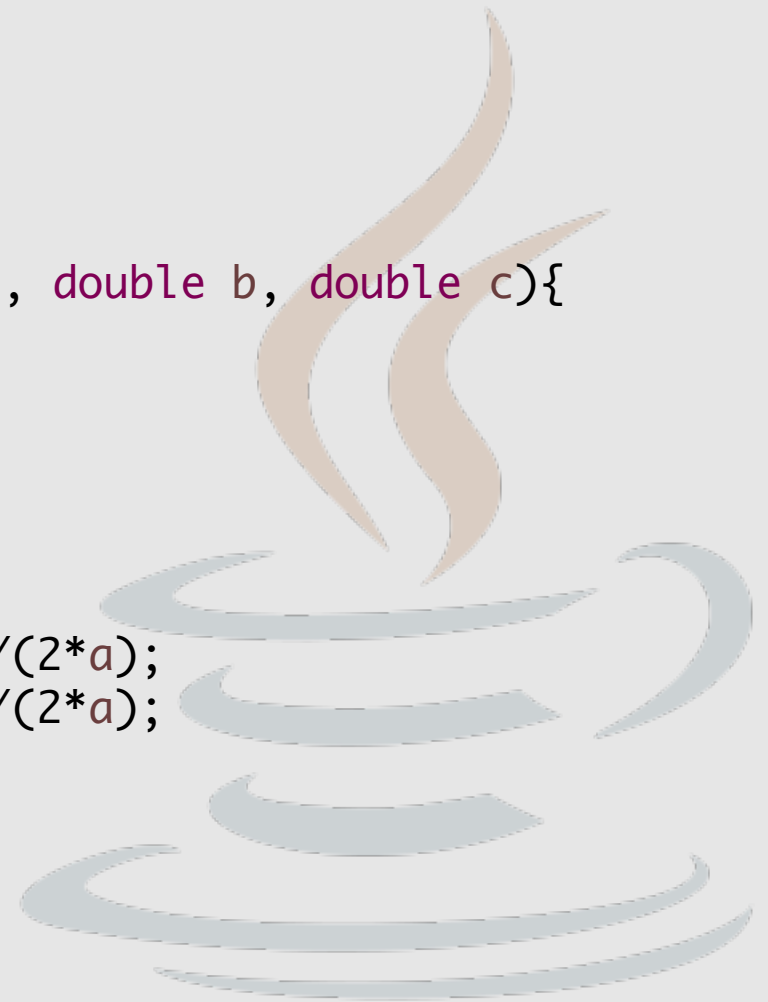
```
java SquareRoot 2
```



# PRZYKŁADY

## ParabolaRoots.java

```
public class ParabolaRoots {
    public static double[] getRoots(double a, double b, double c){
        double[] roots = new double[3];
        double delta = b*b-4*a*c;
        if (delta<0){
            roots[0] = 0;
        }else{
            roots[0] = (delta==0)?1:2;
            roots[1] = (-b+Math.sqrt(delta))/(2*a);
            roots[2] = (-b-Math.sqrt(delta))/(2*a);
        }
        return roots;
    }
}
```



# PRZYKŁADY

ParabolaRoots.java (c.d)

```
public static void main(String[] args){
    double a=Double.parseDouble(args[0]);
    if(a==0)
        System.out.println("Nieprawid\u0142owe dane");
    double b=Double.parseDouble(args[1]);
    double c=Double.parseDouble(args[2]);
    double[] results = getRoots(a, b, c);
    String[] sa = {"Liczba rzeczywistych pierwiastk\u00f3w: ",
                  "x1 = ", "x2 = "};
    for(int i=0; i<results[0]+1; i++)
        System.out.println(sa[i] + results[i]);

    } // koniec metody
} // koniec klasy
```

URUCHOMIENIE (po skompilowaniu):

```
java ParabolaRoots 1 2 -2
```

# ŚRODOWISKA DEWELOPERSKIE

- NETBEANS

<http://netbeans.org/>

- ECLIPSE

<http://www.eclipse.org/>

- INTELLIJ IDEA

<http://www.jetbrains.com/idea/>



# ĆWICZENIA

- Proszę napisać program obliczający pierwiastek  $n$ -tego stopnia z zadanej liczby ( $n$  i liczba są argumentami wywołania programu);
- proszę napisać program obliczający średnią geometryczną liczb podanych jako argumenty wywołania programu;
- proszę napisać program znajdujący największy wspólny dzielnik dwóch liczb podanych jako argumenty wywołania programu.

DZIĘKUJĘ ZA UWAGĘ