

Wyższa Szkoła Informatyki Stosowanej i Zarządzania

pod auspicjami Polskiej Akademii Nauk

WYDZIAŁ INFORMATYKI

Kierunek INFORMATYKA

Studia I stopnia (dyplom inżyniera)



Język Java – laboratorium 5

dr inż. Łukasz Sosnowski
lukasz.sosnowski@wit.edu.pl
sosnowsl@ibspan.waw.pl
l.sosnowski@dituel.pl

www.lsosnowski.pl



Część 1 – Rozliczenie pracy domowej



Prezentacja 1

java.util.Formatter – przegląd możliwości, przykłady użycia



Prezentacja 2

java.util.Calendar – przykłady użycia, konwersje, formatowanie dat



Omówienie rozwiązania Zadania 1:

Wykonaj treść przewodnika udostępnionego w pliku
Przewodnik6.pdf

- AbstractDemo.java
- StudentDemo.java
- TeacherDemo.java



Omówienie rozwiązania Zadania 2:

Zadanie 2: Wykonaj treść przewodnika udostępnionego w pliku Przewodnik7.pdf

- AbstractDemo.java
- StudentDemo.java
- TeacherDemo.java
- IDemoLogger



Omówienie rozwiązania Zadania 3:

Do poniższych klas wykonaj ćwiczenie opisane w pliku

Przewodnik8.pdf

- Person4.java
- Employee4.java
- Secretary4.java
- Programmer4.java
- DbPersons.java



Pakiet dla dzisiejszego laboratorium to:

```
package pl.wit.lab4;  
package pl.wit.lab5.p1;
```

**dla klas biznesowych oraz testów jednostkowych
jednak w odpowiednich podkatalogach katalogu *src*:
main i *test* odpowiednio,**



Część 2 – Operacje wejścia i wyjścia na plikach



Pakiet: `package pl.wit.lab4;`, Katalog UBI: 1
Czytanie zawartości pliku, analiza oraz zapis do pliku

Zadanie 1 domowe dla wszystkich:

Wykonaj treść przewodnika udostępnionego w pliku
Przewodnik9.pdf

- tutorial9.txt

(Do omówienia w trakcie laboratorium a zrobienia w domu)



Część 3 – Interfejsy funkcyjne i wyrażenia lambda



Interfejsy funkcyjne– przypomnienie z wykładu

- Interfejs funkcyjny to taki który zawiera dokładnie jedną zdefiniowaną metodę abstrakcyjną. Zazwyczaj metoda ta określa przeznaczenie interfejsu. Określa również typ docelowy.
- Operator lambda (->) dzieli wyrażenie lambda na dwie części: lewą określającą parametry i prawą określającą ciało wyrażenia.
- Wyrażenie lambda jest metodą anonimową (metodą bez nazwy). Dostarcza implementację dla metody zdefiniowanej w danym interfejsie funkcyjnym.
- (lista parametrów) -> ciało wyrażenia lambda.
- Ciało wyrażenia może mieć dwie postacie: pojedynczej wyrażenia lub bloku kodu.



Pakiet: `package pl.wit.lab4;`, Katalog UBI: 2

Przykłady z wykładu – interfejs funkcyjny i proste wyrażenia lambda:

- `ICalculator.java`
- `IComputer.java`
- `Lambda1Test.java`



Interfejsy funkcyjne parametryzowane, przechwytywanie zmiennych, obsługa wyjątków w wyrażeniach

- Używanie zmiennych lokalnych pochodzących z zasięgu w jakim zostały zdefiniowane wewnątrz wyrażenia lambda nazywamy **przechwyceniem zmiennej (ang. capture variable)**.
- Takie zmienne otrzymują status praktycznie sfinalizowanych (ang. *effectively final*), co powoduje iż po pierwszym przypisaniu nie można zmienić ich wartości.
- Wyrażenia lambda mogą zgłaszać wyjątki. W celu umożliwienia generowania wyjątku odpowiednia deklaracja wyjątków, musi się znaleźć w interfejsie funkcyjnym przy metodzie abstrakcyjnej. Do tego celu używamy słowa kluczowego *throws*.



Pakiet: `package pl.wit.lab4;`, Katalog UBI: 3

Przykłady z wykładu – parametryzowane interfejsy funkcyjne, przechwytywanie zmiennych, obsługa wyjątków w wyrażeniach lambda

- ICalculator2.java
- IComputer2.java
- Lambda2Test.java



Referencje do metod, konstruktorów

- W zmiennej interfejsu funkcyjnego można zapisać referencję do metody, co pozwala odwoływać się do metody bez jej wykonywania.
- Referencję metody statycznej tworzymy poprzez podanie nazwy klasy oraz nazwy metody statycznej rozdzielonych podwójnym znakiem „:”
- Referencję do metody instancyjnej danego obiektu można utworzyć: *referencjaObiektu::nazwaMetody* lub dla dowolnego obiektu danej klasy: *NazwaKlasy::nazwaMetodyInstancyjnej*
Wtedy pierwszy parametr interfejsu odpowiada obiektowi, a pozostałe wywołania met.
- Java umożliwia również utworzenie referencji do konstruktorów.
Ogólna postać wyrażenia: *NazwaKlasy::new*



Pakiet: `package pl.wit.lab5.p1;`, **Katalog UBI: 4**

Przykłady z wykładu – referencje do metod statycznych i instancyjnych, referencje do konstruktorów

- `IntTester.java`
- `IntTester2.java`
- `IntTester3.java`
- `IntTester4.java`
- `ExampleInstanceInt.java`
- `ExampleStaticInt.java`
- `MStaticReferenceDemo.java`
- `Lambda3Test.java`



Predefiniowane interfejsy funkcyjne - przypomnienie

Interfejs	Opis
UnaryOperator<T>	Przyjmuje jeden argument typu T oraz zwraca wartość typu T. Udostępnia metodę <i>apply()</i> .
BinaryOperator<T>	Przyjmuje dwa argumenty typu T oraz zwraca wartość typu T. Udostępnia metodę <i>apply</i> .
Consumer<T>	Przyjmuje jeden argument typu T, nie zwraca wartości. Udostępnia metodę <i>accept</i> .
Supplier<T>	Zwraca wartość typu T, nie przyjmuje argumentu. Udostępnia metodę <i>get</i> .
Function<T,R>	Przyjmuje jeden argument typu T, zwraca wartość typu R. Udostępnia metodę <i>apply</i> .
Predicate<T>	Przyjmuje jeden argument typu T, zwraca wartość typu <i>boolean</i> . Udostępnia metodę <i>test</i> .



Pakiet: `package pl.wit.lab5.p1;` **Katalog UBI: 5**

Przykłady z wykładu – predefiniowane interfejsy funkcyjne

- Lambda4Test.java



Pakiet: `package pl.wit.lab5;`, **Katalog UBI: 6**

Zadanie 2 domowe dla wszystkich:

Przewodnik 10 – proste wyrażenia lambda

Pliki ilustracyjne:

- LambdaTutorial.java
- LambdaTutorialTest.java



Część 4 – Praca domowa



Praca domowa:

- **Przewodnik nr 9**
- **Przewodnik nr 10**

Prezentacja na temat:

- `org.apache.commons.lang3.StringUtils` – przegląd metod, przykłady użycia
- `org.apache.commons.lang3.math.NumberUtils` – przegląd metod, przykłady użycia



Część 5 – Omówienie kolokwium

Wyższa Szkoła Informatyki Stosowanej i Zarządzania

pod auspicjami Polskiej Akademii Nauk

WYDZIAŁ INFORMATYKI

Kierunek INFORMATYKA

Studia I stopnia (dyplom inżyniera)



Dziękuję za uwagę!