	Wydział Politechniczny **Katedra Informatyki**
Prowadzący	Tomasz Gądek
Kurs	Narzędzia i środowiska programistyczne
Rok / Semestr	1 / Letni
Temat	Lab01: Wprowadzenie. Podstawy programowania w języku Kotlin.

Data ostatniej modyfikacji: **01-10-2024**

Warunki zaliczenia przedmiotu

- Kolokwium zaliczeniowe z przedmiotu.
- Realizacja zadań na laboratoriach.
- Obecność.

Narzędzia i technologie wykorzystywane na zajęciach

Język Kotlin

Statycznie typowany język programowania działający na maszynie wirtualnej Javy, który jest głównie rozwijany przez programistów **JetBrains**. Nazwa języka pochodzi od wyspy **Kotlin** niedaleko Petersburga.

Jednym z zastosowań języka **Kotlin** jest platforma **Android**. Został on ogłoszony oficjalnym językiem programowania na konferencji **Google I/O 2017**.

IntelliJ IDEA

Komercyjne zintegrowane środowisko programistyczne (IDE) firmy JetBrains.

Bitbucket

Internetowa usługa hostingowa należąca do Atlassian, używana do tworzenia kodu źródłowego i projektów programistycznych wykorzystujących system kontroli wersji **Git**.

Git

Jest jednym ze znanych i szeroko stosowanych systemów kontroli wersji do tworzenia oprogramowania. Jest to darmowy, na licencji open source, rozproszony system przeznaczony do szybkiej i wydajnej obsługi dowolnych projektów, zarówno tych najmniejszych, jak i tych bardzo dużych i skomplikowanych. Do przechowywania kodów źródłowych projektów i kontroli wersji wykorzystywane jest repozytorium **Git**.

Narzędzia

Instalacje:

- Instalacja Git.
- Instalacja Java.
- Instalacja IntelliJ IDEA.
- Załóż konto w serwisie Bitbucket.

Dokumentacje:

- Dokumentacja języka Kotlin.
- Dokumentacja systemu kontroli wersji Git.

YouTube:

- Kotlin in 100 Seconds.

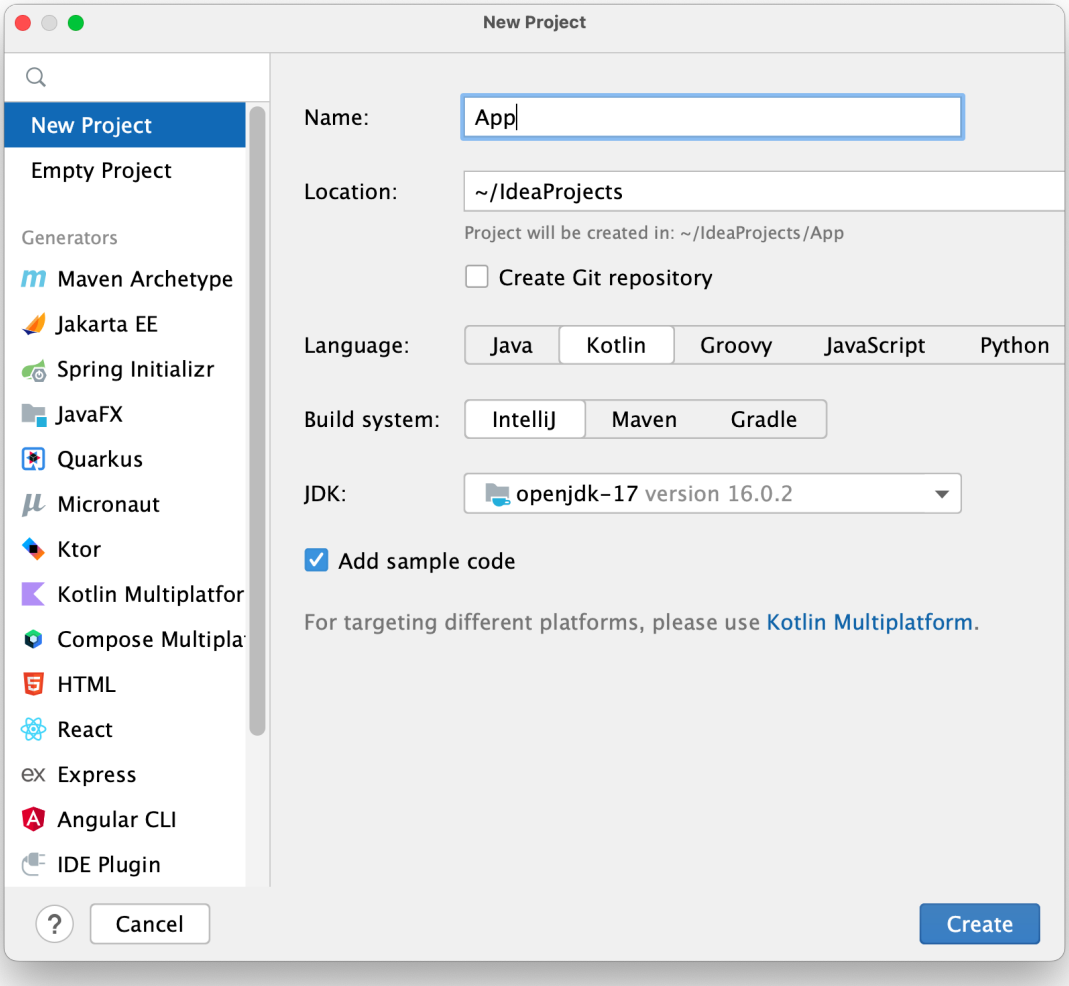
Tworzenie nowego projektu w Kotlinie

Otwórz IntelliJ IDEA

Kliknij w przycisk New Project

Następnie:

- Nadaj nazwę (np. Name: **App**),
- Wybierz lokalizację projektu (np. Location: **~/IdeaProjects**),
- Wybierz język programowania (Language: **Kotlin**),
- Wybierz system budowania (**IntelliJ**),
- Wybierz wersję Java (**1.8** lub **nowsza**),
- Zaznacz **Add sample code**.

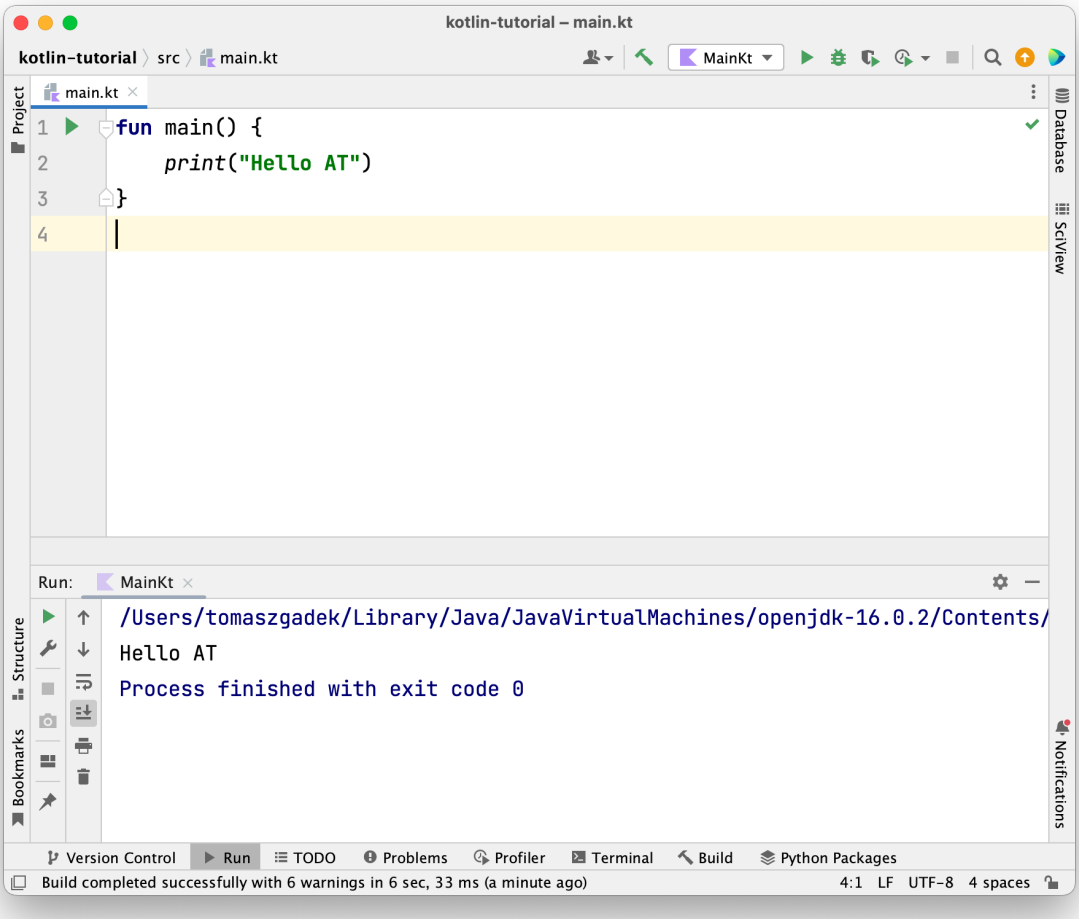


Utwórz projekt klikając **Create**.

Rozwiń drzewo katalogów z lewej strony i w pliku main.kt umieść poniższy fragment kodu:

```
fun main() {
    print("Hello AT")
}
```

Uruchomienie: **Run / Run 'Main.kt'**.

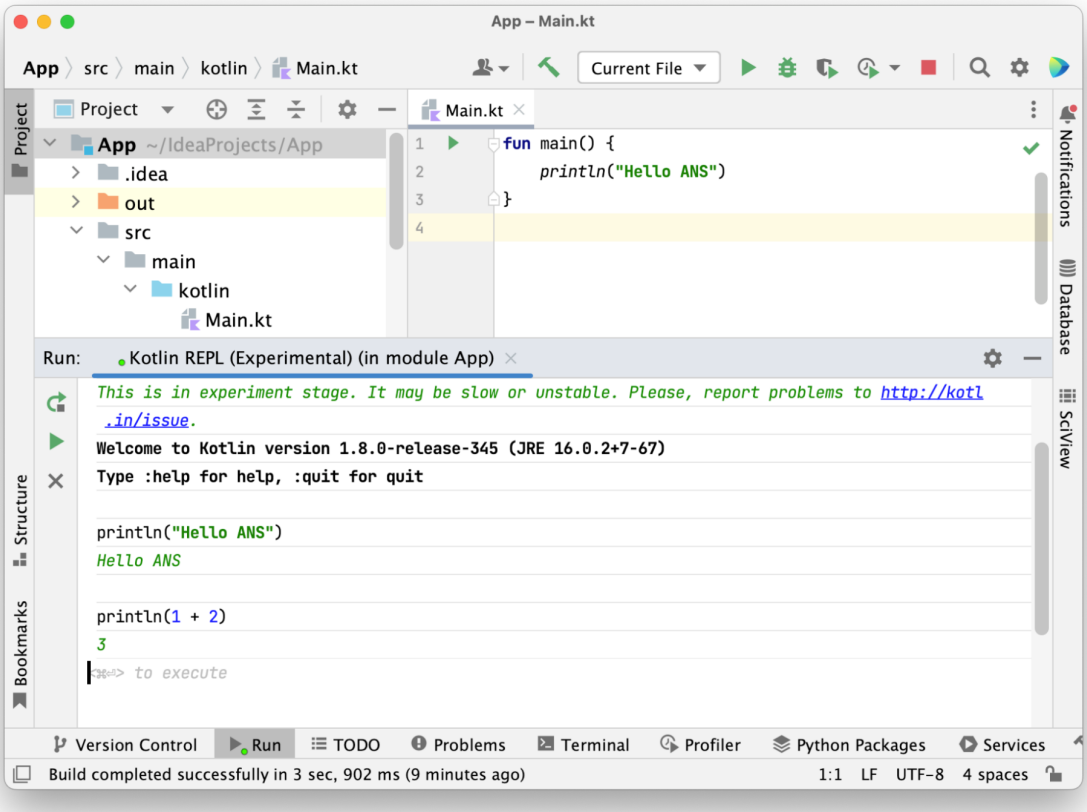


REPL

REPL (ang. *read-eval-print loop*) - proste, interaktywne środowisko programowania.

Wybierz opcje: **Tools / Kotlin / Kotlin REPL (Experimental)**

Teraz możesz eksperymentować z językiem Kotlin. Możesz testować i próbować różne funkcje / funkcjonalności. Poniżej zaprezentowano przykład użycia:



Podstawy programowania w języku Kotlin

Szczypta historii i szybki start

Język **Kotlin** powstał w 2011 roku jako alternatywa dla języków uruchamianych na JVM (wirtualna maszyna Java). Jego nazwa wywodzi się od wyspy Kotlin, która usytuowana jest niedaleko **Petersburga**, siedziby firmy **JetBrains**. Cechuje go niezwykła zwięzłość w stosunku do języka **Java**.

Spójrz na pierwszy program w języku Kotlin. Kod języka Kotlin umieszczamy w plikach z rozszerzeniem **kt**.

```
fun main() {
   print("Hello AT")
}
```

Rezultat:

```
Hello AT
```

Spójrz, jak niewiele trzeba, aby uruchomić program w Kotlin. Wystarczy wewnątrz nawiasów klamrowych **{** i **}** umieścić następującą konstrukcję **println("Hello AT")**. To co znajduje się wewnątrz znaków cudzysłowu zostanie wyprowadzone jako odpowiedź naszego programu.

W programie znajduje się sporo elementów, których nie znasz. Słowo kluczowe **fun** świadczy o występowaniu funkcji (o funkcjach opowiemy sobie w kolejnych etapach), następnie **main()** to nazwa funkcji, która musi wystąpić w naszym skrypcie. Jest to główna funkcja, od której nasz program rozpoczyna swoje działanie. Klamry określają ciało funkcji, to w jej wnętrzu możemy umieszczać różne instrukcje, czy wyrażenia. My umieściliśmy specjalną funkcję o nazwie **print()**. Jej zadaniem jest drukowanie danych na wyjście. W naszym konkretnym przypadku wydrukowaliśmy ciąg tekstowy **Hello AT**.

Komentarze

Komentarze służą do dokumentowania (wyjaśniania) fragmentów kodu. Kompilator języka Kotlin pomija komentarze podczas procesu kompilacji. Komentarz jednoliniowy rozpoczynamy od znaków //, a komentarz wieloliniowy rozpoczynamy od **/***, a kończymy ***/**.

```
// funkcja main() – komentarz jednoliniowy
fun main() {
   /*
       funkcja println()
       komentarz wieloliniowy
   */
   println("Hello AT") // Hello AT
}
```

Rezultat:

```
Hello AT
```

Zmienne

Zmienne to specjalne opakowania, które mogą przechowywać liczby, znaki, czy ciągi tekstowe. Nasze zmienne (opakowania) posiadają odpowiednie nazwy. W kodzie możemy posługiwać się nazwą takiej zmiennej i wykonywać odpowiednie operacje. Spójrz na przykład użycia zmiennych w języku Kotlin.

```
fun main() {
   var age = 18
   var name: String = "Monica"

   println(name)
   println(age)
}
```

Jeżeli inicjalizuje zmienną to nie musimy podawać typu. Kompilator Kotlina na podstawie r-wartości "stwierdzi" z jakim typem ma do czynienia.

W powyższym kodzie źródłowym znajduje się użycie dwóch zmiennych. Pierwsza zmienna o nazwie **age** przechowuje liczby całkowite (18), druga zmienna to **name**, przechowuje ciąg tekstowy **Monica**. Spójrz, w jaki sposób możemy tworzyć zmienne. Na samym początku podajemy słowo kluczowe var, następnie nazwę zmiennej. Po nazwie zmiennej (po dwukropku) może wystąpić typ danych (o typach danych opowiemy sobie w kolejnych etapach). Następnie używamy operatora “=” do przypisania wartości do zmiennej. Operator “=” nie służy do porównywania danych, służy do przypisania wartości po prawej stronie do zmiennej występującej po jego lewej stronie.

Zwróć uwagę, że w języku Kotlin średniki na końcu instrukcji nie są wymagane. Po zainicjowaniu naszych zmiennych możemy wyprowadzić ich wartości za pomocą funkcji **println()**. Ta metoda nieco różni się od metody, która została zaprezentowana w pierwszym etapie. Funkcja **println()** oprócz tego, że wyświetla dane na wyjście dodatkowo dodaje na końcu znak nowej linii, więc dane możemy wyprowadzić jedna pod drugą. Oto rezultat działania naszego kodu.

```
Monica
18
```

Zadanie

Zmodyfikuj kod i postaraj się w pierwszej kolejności zadeklarować zmienne, a następnie przypisać do nich wartości. Jaki będzie wynik działania programu? Jakie nasuwają się wnioski?

Stałe

Stałe definiuje się przy pomocy słowa kluczowego **val**. Wartości stałych nie mogą ulec zmianie po zainicjowaniu.

```
fun main() {
    val pi: Double = 3.14
    val movie = "Matrix"

    println(pi)
    println(movie)
}
```

Rezultat:

```
3.14
Matrix
```

Zadania

- Zmodyfikuj kod i postaraj się w pierwszej kolejności zadeklarować stałe, a następnie przypisać do nich wartości.
- W drugim kroku zainicjuj stałą, a następnie spróbuj przypisać do niej nową wartość.

Jakie będą rezultaty wprowadzonych zmian? Jakie nasuwają się wnioski?

Raz zainicjowanej stałej nie będziesz w stanie zmodyfikować.

Typy danych

Typ danych określa, jakie dane mogą być przechowywane w zmiennych. Oto podstawowe typy danych:

- **Int** - typ danych reprezentujący liczby całkowite (np. **1**, **6**, **10**).
- **Double** - typ danych reprezentujący liczby zmiennoprzecinkowe (np. **3.14**).
- **Boolean** - typ danych reprezentujący wartości logiczne (np. **true** lub **false**).
- **Char** - typ danych reprezentujący pojedynczy znak (np. **'x'**, **'H'**).
- **String** - typ danych reprezentujący ciągi tekstowe (np. **"Hello"**).

Spójrz na przykładowe zastosowanie typów danych w praktyce.

```
fun main() {
    var name: String = "Monica"
    var gender: Char = 'W'
    var isAdult: Boolean = true
    var age: Int = 35
    var salary: Double = 3.500

    println(name)
    println(gender)
    println(isAdult)
    println(age)
    println(salary)
}
```

Rezultat:

```
Monica
W
true
35
3.5
```

Zadanie

Zmodyfikuj kod w taki sposób, aby wyprowadzić na wyjście własne atrybuty (Twoje imię, płeć, wiek, etc.).

Ciągi tekstowe

Ciągi tekstowe w języku Kotlin reprezentowane są przez typ **String**. Już mieliśmy okazję je poznać. Są to wszelkie znaki (litery, cyfry, znaki specjalne) objęte cudzysłowem. W ciągu tekstowym każdy znak posiada swój własny unikalny identyfikator (indeks). Możemy uzyskać dostęp do konkretnego znaku. Pierwszy znak posiada indeks = 0. Spójrz, jak uzyskać dostęp do drugiego i ostatniego znaku.

```
fun main() {
   var first = "ab"
   var second = "cd"
   var result = first + "," + second

   println(result[1])
   println(result[4])
}
```

Rezultat:

```
b
d
```

W języku Kotlin mamy możliwość łączenia ciągów tekstowych. Taką operację nazywamy **konkatenacją**. Łączenie ciągów tekstowych odbywa się przy pomocy operatora **+**.

Dodatkowo istnieje możliwość uzyskania informacji o liczbie znaków, które zawiera ciąg. Wystarczy po nazwie zmiennej, reprezentującej ciąg tekstowy, po kropce odnieść się od **length**.

```
fun main() {
   var numbers = "1234"
   var size = numbers.length

   println(size)
}
```

Rezultat:

```
4
```

Dodatkowo, mamy możliwość "wplatania" zmiennych w ciąg tekstowy przy pomocy specjalnej składni **${}**.

```
fun main() {
    var name: String = "Joe"
    var age: Int = 45
    var message = "Hello! I am ${name}, ${age} years old."
    println(message)
}
```

Rezultat:

```
Hello! I am Joe, 45 years old.
```

Z jaką łatwością możemy wplatać wartości naszych zmiennych w ciąg tekstowy. Jest to bardzo przydatna własność.

Zadanie

Utwórz zmienną **name** i przypisz do niej własne imię. Program powinien wyprowadzić tekst, który zawiera Twoje imię oraz liczbę znaków, z których się składa. Program powinien być uniwersalny, jeżeli przypisze inne imię do zmiennej **name** to ilość znaków w wyprowadzonym tekście powinna się zmieniać dynamicznie.

Operatory arytmetyczne

Operatory arytmetyczne w języku Kotlin służą do wykonania podstawowych operacji matematycznych. Oto lista podstawowych operacji matematycznych: dodawanie, odejmowanie, mnożenie, dzielenie, dzielenie z resztą, inkrementacja (zwiększenie wartości zmiennej o 1) oraz dekrementacja (zmniejszenie wartości zmiennej o 1).

```
fun main() {
    var sum = 2 + 4
    var sub = 2 - 4
    var mul = 2 * 4
    var div = 2.0 / 4.0
    var mod = 2 % 4

    var digit = 2

    println("sum = ${sum}, sub = ${sub}")
    println("mul = ${mul}, div = ${div}")
    println("inc = ${++digit}, dec = ${--digit}")
    println("mod = ${mod}")
}
```

Rezultat:

```
sum = 6, sub = -2
mul = 8, div = 0.5
inc = 3, dec = 2
mod = 2
```

Zaprezentowane operacją są to podstawowe operacje znane z matematyki klasycznej. Uważaj na priorytety operatorów. W pierwszej kolejności wykonujemy mnożenie i dzielenia, a następnie dodawanie i odejmowanie. Jeżeli chcesz kontrolować priorytet operacji to zastosuj nawiasy.

```
fun main() {

   var result1 = 2 + 2 * 2
   var result2 = (2 + 2) * 2

   println(result1)
   println(result2)
}
```

Rezultat:

```
6
8
```

Działania nie są wykonywane od lewej do prawej, ale zgodnie z priorytetem operatorów. Jeżeli nie jesteśmy pewni priorytetu zaleca się stosowanie nawiasów.

Zadania

- Twoje zadanie będzie polegało na zaimplementowaniu programu, który wyznaczy pole trapezu.
- Twoje zadanie będzie polegało na zaimplementowaniu programu, który wyznaczy pole trójkąta.
- Twoje zadanie będzie polegało na zaimplementowaniu programu, który wyznaczy pole koła.
- Utwórz 3 zmienne typu **Double**, przypisz do nich wartości, a następnie wyznacz ich średnią arytmetyczną.

Czy wiesz, że...

ChatGPT to zaawansowane narzędzie oparte na sztucznej inteligencji, które potrafi generować tekst na podstawie swoich treningowych danych. Może odpowiadać na pytania, prowadzić rozmowy na różne tematy i dostarczać ciekawe informacje.