Programowanie obiektowe

Karta opisu przedmiotu

Informacje podstawowe

Kierunek studiów Informatyka (kierunek wspólny - WI)	**Cykl dydaktyczny** 2020/2021
Specjalność -	**Kod przedmiotu** WIINFS.li4.00248.20
Jednostka organizacyjna Wydział Informatyki	**Języki wykładowe** polski
Poziom kształcenia Studia inżynierskie I stopnia	**Obligatoryjność** Obowiązkowy
Forma studiów Stacjonarne	**Blok zajęciowy** Przedmioty ogólne
Profil studiów Ogólnoakademicki	**Przedmiot powiązany z badaniami naukowymi** Tak

Koordynator przedmiotu	Aleksander Smywiński-Pohl
Prowadzący zajęcia	Aleksander Smywiński-Pohl, Zbigniew Kaleta, Stanisław Polak, Radosław Łazarz, Monika Dekster

Okres Semestr 3	**Forma zaliczenia** Zaliczenie **Forma prowadzenia i godziny zajęć** Wykład: 30 Ćwiczenia laboratoryjne: 30	**Liczba punktów ECTS** 3

Cele kształcenia dla przedmiotu

C1	Zapoznanie studentów z terminologią programowania obiektowego
C2	Zapoznanie studentów z modelami obiektowości
C3	Uświadomienie studentom istoty ukrywania implementacji
C4	Uświadomienie różnicy pomiędzy typami konkretnymi i abstrakcyjnymi
C5	Zapoznanie studentów z mechanizmem wyjątków
C6	Zapoznanie studentów z wybraną biblioteką realizującą paradygmat obiektowy
C7	Zaznajomienie studentów z mechanizmem refleksji
C8	Przekazanie podstawowych informacji na temat programowania współbieżnego i równoległego
C9	Zapoznanie studentów z typami ogólnymi (parametryzowanymi)

Efekty uczenia się dla przedmiotu

Kod	Efekty w zakresie	Kierunkowe efekty uczenia się	Metody weryfikacji
Wiedzy – Student zna i rozumie:			
W1	Zna charakterystykę (w tym złożoność obliczeniową najważniejszych metod) następujących klasach i interfejsów Javy: * ArrayList * LinkedList * HashSet * TreeSet * HashMap * TreeMap * Stream * Thread * Runnable	INF1A_W03, INF1A_W04	Egzamin
W2	Student zna i rozumie znaczenie pojęć: * abstrakcja * hermetyzacja * dziedziczenie * kompozycja * klasa * interfejs	INF1A_W03, INF1A_W04	Egzamin
W3	Ma wiedzę na temat następujących mechanizmów programowania: * poziomy dostępu do atrybutów i metod * zależności pomiędzy dziedziczeniem a dostępnością atrybutów i metod * cykl życia obiektu * obsługa wyjątków * hazard (programowanie współbieżne) * sekcja krytyczna * refleksja * typy ogólne (parametryczne)	INF1A_W03, INF1A_W04	Egzamin
W4	Ma wiedzę na temat właściwego stosowania mechanizmów programowania obiektowego, w szczególności: * dziedziczenia * metod i atrybutów statycznych * klas wspierających programowanie współbieżne * abstrakcji	INF1A_W03, INF1A_W04	Egzamin
Umiejętności – Student potrafi:			
U1	Posługuje się językiem angielskim w stopniu wystarczającym do czytania ze zrozumieniem dokumentacji klas dostępnych w standardowej bibliotece Javy.	INF1A_U05	Wykonanie ćwiczeń laboratoryjnych
U2	Potrafi napisać program w języku Java lub innym języku obiektowym, który nie zawiera błędów składniowych oraz realizuje nieskomplikowane wymagania funkcjonalne.	INF1A_U05	Wykonanie ćwiczeń laboratoryjnych

Kod	Efekty w zakresie	Kierunkowe efekty uczenia się	Metody weryfikacji
U3	Potrafi praktycznie wykorzystać wiedzę dotyczącą zagadnień: * abstrakcji * dziedziczenia * ukrywania implementacji * klas * interfejsów * kompozycji	INF1A_U05	Projekt
U4	Potrafi projektować i implementować oprogramowanie korzystając z gotowych bibliotek dostarczanych przez język programowania	INF1A_U05	Wykonanie ćwiczeń laboratoryjnych
Kompetencji społecznych – Student jest gotów do:			
K1	Potrafi pozyskiwać informacje z literatury, baz danych i innych źródeł w celu określenia podziału pracy na zadania oraz określenia ich priorytetów.	INF1A_K02, INF1A_K03, INF1A_K04	Wykonanie ćwiczeń laboratoryjnych

Treści programowe zapewniające uzyskanie efektów uczenia się dla modułu zajęć

Celem zajęć jest zaznajomienie studentów z paradygmatem programowania obiektowym. Studenci poznają techniki takie jak: dziedziczenie, delegowanie, ukrywanie implementacji.

Nakład pracy studenta

Rodzaje zajęć studenta	Średnia liczba godzin* przeznaczonych na zrealizowane aktywności
Wykład	30
Ćwiczenia laboratoryjne	30
Przygotowanie do zajęć	15
Samodzielne studiowanie tematyki zajęć	15
Łączny nakład pracy studenta	**Liczba godzin** 90
Liczba godzin kontaktowych	**Liczba godzin** 60

* godzina (lekcyjna) oznacza 45 minut

Treści programowe

Lp.	Treści programowe	Efekty uczenia się dla przedmiotu	Formy prowadzenia zajęć
1.	Wprowadzenie do programowania obiektowego: Przedstawienie paradygmatu programowania obiektowego na tle innych paradygmatów programowania. Wprowadzenie podstawowych pojęć związanych z programowaniem obiektowym	W1, W2, U1, K1	Wykład, Ćwiczenia laboratoryjne
2.	Język Java i wirtualna maszyna Javy: Wskazanie podstawowych własności języka Java. Omówienie wirtualnej maszyny języka Java.	W1, W2, U1	Wykład, Ćwiczenia laboratoryjne

Lp.	Treści programowe	Efekty uczenia się dla przedmiotu	Formy prowadzenia zajęć
3.	Podstawowe typy danych w Javie i Rubim: Omówienie podstawowych typów danych w języku Java. Odróżnienie typów obiektowych i prymitywnych. Omówienie typów otokowych.	W1, W2, U1, U2	Wykład, Ćwiczenia laboratoryjne
4.	Stan w programowaniu obiektowy (Java i Ruby): Sposoby definiowania stanu w programie obiektowym. Zmienne instancyjne i zmienne klasowe. Dostęp publiczny, prywatny, chroniony i pakietowy.	W1, W4, U1, U2, U4	Wykład, Ćwiczenia laboratoryjne
5.	Definiowanie zachowania obiektów: Sposoby definiowania zachowania w programowaniu obiektowym. Metody instancyjne i klasowe. Dostęp publiczny, prywatny, chroniony i pakietowy. Dziedziczenie a definiowanie zachowania. Zalecane praktyki związane z dostępem do metod.	W1, W4, U1, U2, U3	Wykład, Ćwiczenia laboratoryjne
6.	Wyjątki: Mechanizm obsługi błędów w oparciu o wyjątki. Rodzaje błędów w systemach informatycznych. Zalecane praktyki związane z obsługą wyjątków.	W1, W2, W3, W4, U1, U2, U3	Wykład, Ćwiczenia laboratoryjne
7.	Kolekcje w Javie: Najważniejsze klasy implementujące kolekcje obiektów. Omówienie interfejsu Map, Set, Collection, List. Omówienie najważniejszych implementacji interfejsów.	W1, W2, W3, W4, U1	Wykład, Ćwiczenia laboratoryjne
8.	Typy zagnieżdżone oraz domknięcia: Wprowadzenie pojęcia typów zagnieżdżonych. Omówienie zasad dostępu dla typów zagnieżdżonych. Domknięcia oraz klasy anonimowe w Javie.	W1, W2, W3, W4	Wykład
9.	Refleksja w Rubim i w Javie: Omówienie mechanizmu refleksji w języku Ruby. Omówienie tradycyjnego mechanizmu refleksji w Javie. Omówienie ograniczeń związanych z zastosowaniem refleksji.	W1, W2, W3, W4	Wykład
10.	Typy ogólne: Omówienie koncepcji typów ogólnych. Omówienie uzasadnienie dla stosowania typów ogólnych. Ograniczenia związane ze stosowaniem typów ogólnych.	W3, W4	Wykład
11.	Wątki: Omówienie implementacji wątków w Javie (Runnable oraz Thread). Omówienie ryzyk związanych z programowaniem współbieżnym. Omówienie klas ułatwiających programowanie współbieżne.	W1, W4, U3, U4	Wykład, Ćwiczenia laboratoryjne
12.	Wejście i wyjście: Omówienie mechanizmów obsługi wejścia i wyjścia w języku Java.	W4, U1, U4	Wykład, Ćwiczenia laboratoryjne
13.	Cechy dobrego kodu: Nazewnictwo Testy Law of Demeter Zasady SOLID	W1, W2, W3, W4, U2, U4, K1	Wykład, Ćwiczenia laboratoryjne

Informacje rozszerzone

Metody i techniki kształcenia :

Wykłady będą odbywać się w sposób zdalny z użyciem platformy Webex/Teams, Kształcenie zdalne, Praca grupowa, Grywalizacja, gamifikacja

Rodzaj zajęć	Metody zaliczenia	Warunki zaliczenia przedmiotu
Wykład	Egzamin	wyniki kolokwium mają wpływ na ocenę końcową
Ćwiczenia laboratoryjne	Wykonanie ćwiczeń laboratoryjnych, Projekt	aktywność na zajęciach realizacja projektu oraz zaliczenie laboratoriów maja wpływ na ocenę końcową

Dodatkowy opis

Wykłady z przedmiotu będą prowadzone w sposob zdalny z wykorzystaniem platformy Webex lub Teams.
Pozostałe zajęcia będą odbywać się w salach. Dotyczy to także zaliczeń i egzaminów odbywajacych się w sesjach egzaminacyjnych. Oczywiście będzie to zależeć od aktualnej sytuacji epidemicznej.
Zajęcia są prowadzone z wykorzystaniem innowacyjnych metod dydaktycznych opracowanych w projekcie POWR.03.04.00-00-D002/16, realizowanym w latach 2017-2019 na Wydziale Informatyki, Elektroniki i Telekomunikacji w ramach Programu Operacyjnego Wiedza Edukacja Rozwój 2014-2020.

Warunki i sposób zaliczenia poszczególnych form zajęć, w tym zasady zaliczeń poprawkowych, a także warunki dopuszczenia do egzaminu

Warunkiem uzyskania zaliczenia jest uzyskanie min. 15 punktów. Szczegółowy opis podano w sekcji "Dodatkowy opis"

Sposób obliczania oceny końcowej

W trakcie laboratoriów realizowane są następujące działania weryfikujące postępy w nauce:
- sprawdziany w formie testów - łącznie 8 punktów
- zaliczenie zadania laboratoryjnego - łącznie 10 punktów (8 zadań, 2 punkty za aktywność)
- wybór tematu metodą kuli śniegowej - max 2 punkty
- realizacja 1 projektu - 8 punktów
- realizacja 2 projektu - 4 punktów

Warunkiem uzyskania zaliczenia jest uzyskanie min. 15 punktów (w sumie) oraz uzyskanie min. 8 punktów ze sprawdzianów i zadań laboratoryjnych. Warunkiem realizacji 2 projektu jest uzyskanie min. 5 punktów za pierwszy projekt.
Studencie w ramach ćwiczeń zobowiązani są do korzystania z systemu śledzenia zmian w kodzie (Git) oraz wysyłania rozwiązań zadań na serwer, na który dostęp ma prowadzący. Ma to m.in. zapobiec powstawaniu prac niesamodzielnych oraz – w przypadku wykrycia prac tego rodzaju – ustalenia faktycznego autorstwa rozwiązania.
Stosowana jest skala AGH, przy czym przyjmuje się, że suma punktów wynosi 30 (możliwe jest uzyskanie 2 dodatkowych punktów, które nie liczą się do tej sumy).

Sposób i tryb wyrównywania zaległości powstałych wskutek nieobecności studenta na zajęciach

W przypadku wystąpienia zaległości w realizacji zadań w ramach ćwiczeń student: 1. może zaprezentować rozwiązanie zadania na kolejnych zajęciach 2. może zaprezentować rozwiązanie zadania oraz skonsultować występujące problemy w ramach konsultacji

Wymagania wstępne i dodatkowe

Znajomość języka C/C++ w podstawowym zakresie.

Zasady udziału w poszczególnych zajęciach, ze wskazaniem, czy obecność studenta na zajęciach jest obowiązkowa

Wykład: Studenci uczestniczą w zajęciach, poznając kolejne treści nauczania zgodnie z syllabusem przedmiotu. Studenci

winni na bieżąco zadawać pytania i wyjaśniać wątpliwości. Rejestracja audiowizualna wykładu wymaga zgody prowadzącego. Ćwiczenia laboratoryjne: Studenci wykonują ćwiczenia laboratoryjne zgodnie z materiałami udostępnionymi przez prowadzącego. Student jest zobowiązany do przygotowania się w przedmiocie wykonywanego ćwiczenia, co może zostać zweryfikowane kolokwium w formie ustnej lub pisemnej. Zaliczenie zajęć odbywa się na podstawie zaprezentowania rozwiązania postawionego problemu. Zaliczenie modułu jest możliwe po zaliczeniu wszystkich zajęć laboratoryjnych.

Literatura

Obowiązkowa

1. „The Java Tutorial” - zasoby online, https://docs.oracle.com/javase/tutorial/
2. Bruce Eckel: Thinking in Java 4th Edition, 1998

Dodatkowa

1. B.Evans, D. Flanagan, Java w Pigułce 2015
2. E.Gamma et al “Design Patterns – : Elements of Reusable Object-Oriented Software”, 1994
3. John Ousterhout, A Philosophy of Software Design
4. Sandi Metz, Practical Object-Oriented Design in Ruby

Badania i publikacje

Badania

1. Lemkin - inteligentny system informacji prawnej

Publikacje

1. Artykuł: A. Pohl ROD – Ruby Object Database, Studia Informatica, Volume 33, Number 2A (105), s. 281-298, ISSN: 0208-7286.

Kierunkowe efekty uczenia się

Kod	Treść
INF1A_K02	Ma świadomość ważności i rozumie pozatechniczne aspekty i skutki działalności inżyniera-informatyka
INF1A_K03	Potrafi współdziałać i pracować w grupie, przyjmując w niej różne role, ma świadomość odpowiedzialności za pracę własną i za wspólnie realizowane zadania.
INF1A_K04	Potrafi odpowiednio określić priorytety służące realizacji określonego przez siebie lub innych zadania oraz adekwatnie zaplanować pracę.
INF1A_U05	Potrafi wykorzystać poznane metody i modele do tworzenia programów o charakterze użytkowym, a także potrafi adekwatnie wykorzystać znane algorytmy i struktury danych w budowie systemu komputerowego.
INF1A_W03	Ma szczegółową wiedzę w zakresie wybranych języków i technik programowania oraz uporządkowaną wiedzę w zakresie inżynierii oprogramowania, w tym projektowania i testowania systemów.
INF1A_W04	Ma szczegółową wiedzę w zakresie technik i narzędzi implementacyjnych uwzględniających wybrane aspekty budowy oprogramowania, w szczególności systemów baz danych, aplikacji działających w środowiskach sieciowych i budowy interfejsu użytkownika.