1.	**Nazwa kierunku**	**informatyka**
2.	Wydział	Wydział Nauk Ścisłych i Technicznych
3.	Cykl rozpoczęcia	2019/2020 (semestr letni)
4.	Poziom kształcenia	studia drugiego stopnia
5.	Profil kształcenia	ogólnoakademicki
6.	Forma prowadzenia studiów	stacjonarna

Moduł kształcenia: Paradygmaty programowania

Kod modułu: 08-IN-S2-PP

1. Liczba punktów ECTS: 5

2. Zakładane efekty uczenia się modułu

kod	opis	efekty uczenia się kierunku	stopień realizacji (skala 1-5)
PP _K_7	Potrafi pracować w zespole projektowo-programistycznym	K_K02 K_U02	1 1
PP _U_4	Potrafi skonstruować rozwiązanie podanego problemu zgodnie z określonym paradygmatem programowania i zapisać go w wybranym języku programowania	K_U01 K_U05 K_U12 K_U13 K_U15	1 1 1 1 1
PP _U_5	Potrafi stosować podejście obiektowe, strukturalne, funkcyjne i deklaratywne w wybranych językach programowania	K_U15	1
PP _U_6	Potrafi sprawdzić niezawodność programu komputerowego za pomocą testowania w wybranym środowisku programistycznym i udokumentować program	K_U01 K_U03 K_U05 K_U12 K_U13 K_U15	1 1 1 1 1 1
PP _W_1	Zna paradygmaty programowania: programowanie proceduralne, programowanie obiektowe, programowanie strukturalne, współbieżne, programowanie imperatywne, funkcyjne i deklaracyjne oraz ich powiązanie z architekturą komputerów (w tym równoległych i wieloprocesorowych)	K_W04 K_W06 K_W09 K_W10	1 1 1 1

		K_W12 K_W14	1 1
PP _W_2	Rozumie podstawowe konstrukcje programistyczne oraz zna typy danych języków imperatywnych oraz konstrukcje programistyczne charakterystyczne dla podejścia deklaratywnego i funkcyjnego	K_W06 K_W09 K_W10 K_W12 K_W14	1 1 1 1 1
PP _W_3	Ma wiedzę dotyczącą implementacji mechanizmów charakterystycznych dla konkretnego paradygmatu programowania w wybranych językach programowania	K_W06 K_W09 K_W10 K_W12 K_W14	1 1 1 1 1

3. Opis modułu	
Opis	Celem zajęć jest uzupełnienie wiedzy studentów dotyczącej zasad projektowania i implementowania programów komputerowych oraz rozszerzenie umiejętności pisania czytelnych i sprawnych programów w wybranych językach reprezentujących podejście imperatywne, funkcyjne i deklaratywne. Studenci rozwijają swoją wiedzę i umiejętności stosowania różnych paradygmatów programowania.
Wymagania wstępne	

4. Sposoby weryfikacji efektów uczenia się modułu

kod	nazwa (typ)	opis	efekty uczenia się modułu
PP _w_1	ocena projektu	Studenci wykonują samodzielnie oprogramowanie, którego specyfikacja jest podawana przez prowadzącego	PP _K_7, PP _U_4, PP _U_5, PP _U_6, PP _W_1, PP _W_2, PP _W_3
PP _w_2	prace kontrolne	Kolokwia pisemne (w tym wykonane na komputerze w czasie zajęć)	PP _U_4, PP _U_5, PP _U_6, PP _W_1, PP _W_2, PP _W_3
PP _w_3	egzamin	Studenci projektują i implementują klasy/funkcje/aplikacje, zgodnie z podaną specyfikacją	PP _U_4, PP _U_5, PP _U_6, PP _W_1, PP _W_2, PP _W_3

5. Rodzaje prowadzonych zajęć

kod	rodzaj prowadzonych zajęć			praca własna studenta		sposoby weryfikacji efektów uczenia się
	nazwa	opis (z uwzględnieniem metod dydaktycznych)	liczba godzin	opis	liczba godzin	
PP _fs_1	wykład	Podanie treści kształcenia w formie werbalnej z wykorzystaniem wizualizacji treści. Skupienie się na materiale trudnym	30	Zapoznanie się z tematyką wykładu z wykorzystaniem istniejących pakietów metod: podręczników, skryptów, stron	15	PP _w_1, PP _w_2, PP _w_3

		pojęciowo i wskazanie źródeł. Ilustracja treści za pomocą przykładów.		internetowych itp.		
PP _fs_2	laboratorium	Szczegółowe przygotowanie studentów do rozwiązywania zadań ze wskazaniem na metodologię postępowania, wskazaniem kolejności wykonywanych czynności. Projektowanie rozwiązań i ich implementacja komputerowa.	30	Rozwiązywanie zadań z poszczególnych tematów wraz z analizą rozwiązań już istniejących – w skrypcie i na stronach internetowych. Przygotowanie zagadnień do przedyskutowania lub przygotowanie się do nadrobienia zaległości Samodzielne wykonanie oprogramowania, którego specyfikacja została podana przez prowadzącego, oraz wykonanie dokumentacji Powtórzenie wiadomości podanych na wykładach oraz przećwiczonych w czasie ćwiczeń laboratoryjnych	75	PP _w_1, PP _w_2, PP _w_3