Politechnika Białostocka									
Kierunek studiów	**Matematyka Stosowana**							**Poziom i forma studiów**	**drugiego stopnia stacjonarne**
Specjalność / Ścieżka dyplomowania	**Analityka Danych i Modelowanie Matematyczne**							**Profil kształcenia**	praktyczny
Nazwa przedmiotu	**Inżynieria oprogramowania**							**Kod przedmiotu**	**MAT2IOP**
								Rodzaj przedmiotu	obieralny
Forma zajęć i liczba godzin	W	Ć	L	P	Ps	T	S	**Semestr**	2/3
	30				30			**Punkty ECTS**	**3**
Przedmioty wprowadzające	Wybrane techniki programistyczne (MAT2WTP),								
Cele przedmiotu	Celem wykładu jest przedstawienie całego procesu związanego z tworzeniem i wykorzystywaniem systemów informatycznych. Powinien uświadomić słuchaczom, że programowanie jest tylko elementem składowym tego procesu oraz, że na powodzenie przedsięwzięcia mają wpływ wszystkie fazy cyklu życia oprogramowania. W skład wykładu wchodzi również mini-kurs pokazujący wykorzystanie Unified Modeling Language (UML) w modelowaniu i projektowaniu systemów. Celem pracowni specjalistycznej jest praktyczne zapoznanie się z modelowaniem i projektowaniem w UML-u przy wykorzystaniu narzędzia CASE. W pierwszej części zajęć rysowane są diagramy UML na podstawie zadanych scenariuszy, natomiast w drugiej części zdobyte umiejętności są weryfikowane podczas tworzeniu (wstępnego) projektu wybranego systemu informatycznego.								
Treści programowe	Wykład: cele inżynierii oprogramowania, przyczyny powstania IO, metodyka a metodologia, narzędzia CASE; wprowadzenie do UML, diagramy przypadków użycia systemu, diagramy czynności; UML: Diagramy klas i obiektów, pakiety; UML: Diagramy interakcji i stanów; UML: Diagramy fizyczne: komponentów i wdrożenia; cykl życia oprogramowania (modele: wodospadowy, spiralny, COTS, ...); inżynieria wymagań dla systemów informatycznych (metody zbierania informacji, wymagania funkcjonalne i niefunkcjonalne); modelowanie i projektowanie systemów; implementacja systemu; testowanie, weryfikacja i walidacja oprogramowania (testy dynamiczne i statyczne); zapewnienie jakości oprogramowania i metryki oprogramowania; dokumentowanie, instalacja, wdrażanie oraz konserwacja oprogramowania; wiarygodność systemów informatycznych; zarządzanie projektami programistycznymi; zarządzanie ryzykiem w projektach. Pracownia specjalistyczna: przykładowe narzędzia CASE; diagram przypadków użycia, opisywanie przypadków użycia; diagram klas, pakiety; diagram czynności; diagram stanów; diagramy interakcji (przebiegu); diagramy fizyczne (komponentów i wdrożenia); uzgadnianie tematyki zadania grupowego, określanie celów i zakresu projektowanego systemu oraz korzyści z jego wdrożenia; tworzenie i opisywanie diagramów przypadków użycia, projektowanie interfejsu użytkownika; tworzenie diagramu klas, identyfikowanie atrybutów i metod, opracowywanie realizacji przypadków użycia, d. interakcji - poziom pojęciowy; opracowywanie realizacji przypadków użycia, tworzenie diagramów przebiegu - poziom implementacyjny, czynności; przygotowywanie diagramów zmiany stanu; specyfikowanie wymagań niefunkcjonalnych i propozycji technologii informatycznych, przygotowanie proponowanego planu pracy i analiza ryzyka projektu; prezentacja projektu, przedstawienie podziału pracy i przekazanie sprawozdania projektowego do oceny.								
Metody dydaktyczne	wykład problemowy, wykład informacyjny, metoda projektów, programowanie z użyciem komputera,								
Forma zaliczenia	Wykład: pisemne zaliczenie (2 zadania praktyczne - diagramy UML oraz 3 pytania teoretyczne); warunkiem przystąpienia do zaliczenia jest zaliczenie pracowni specjalistycznej. Pracownia specjalistyczna: na podstawie krótkich sprawdzianów na zajęciach oraz przygotowywanego w zespołach sprawozdania projektowego.								

Symbol efektu uczenia się	**Zakładane efekty uczenia się**	**Odniesienie do kierunkowych efektów uczenia się**
EU1	zna i rozumie zasady inżynierii oprogramowania, metody i techniki wykorzystywane w projektowaniu systemów informatycznych; zna model cyklu życia oprogramowania; zna i rozumie procesy jego wytwarzania, wdrażania i utrzymania oraz powiązane metody zarządzania i organizacji pracy; zna języki modelowania i komputerowe narzędzia wspomagające projektowanie	K_W07
EU2	zna i rozumie procesy i zasady zarządzania przedsięwzięciami informatycznymi; zna zasady planowania procesu realizacji systemu informatycznego; zna techniki szacowania kosztów przedsięwzięcia i czasu potrzebnego na realizację zleconego zadania	K_W07
EU3	potrafi zaprojektować i zaplanować implementację, testowanie i wdrożenie systemu informatycznego oraz jego komponentów stosując odpowiednie metody, techniki oraz narzędzia, uwzględniając zadane kryteria użytkowe i ekonomiczne	K_U11 K_U13 K_U15 K_U18
EU4	potrafi opracować dokumentację projektową: specyfikację wymagań, architekturę systemu, opis realizacji i technologii, instrukcję użytkownika.; potrafi pracować w grupie jak i samodzielnie	K_U11 K_U13 K_U15 K_U18

Symbol efektu uczenia się	**Sposób weryfikacji efektu uczenia się**	**Forma zajęć na której zachodzi weryfikacja**
EU1	zaliczenie pisemne	W
EU2	zaliczenie pisemne	W
EU3	dokumentacja projektu, dyskusja na temat projektu, obserwacja pracy na zajęciach, krótkie sprawdziany	Ps
EU4	dokumentacja projektu, dyskusja na temat projektu, obserwacja pracy na zajęciach, krótkie sprawdziany	Ps

Bilans nakładu pracy studenta (w godzinach)	**Liczba godz.**
1 - Udział w wykładach -	30
2 - Udział w pracowni specjalistycznej -	30
3 - Opracowanie sprawozdań z pracowni i wykonanie zadań domowych (prac domowych) -	7
4 - Udział w konsultacjach -	2
5 - Realizacja zadań projektowych (w tym przygotowanie prezentacji) -	4
6 - Przygotowanie do zaliczenia wykładu -	2
RAZEM:	**75**

Wskaźniki ilościowe	**GODZINY**	**ECTS**
Nakład pracy studenta związany z zajęciami wymagającymi bezpośredniego udziału nauczyciela	62 (2)+(1)+(4)	2.5
Nakład pracy studenta związany z zajęciami o charakterze praktycznym	41 (2)+(5)+(3)	1.6

Literatura podstawowa

1. W. Dąbrowski, K. Subieta, Podstawy inżynierii oprogramowania, Wydawnictwo PJWSTK, 2005.
2. K. Sacha, Inżynieria oprogramowania, PWN, 2010.
3. A. Koszlajda, Zarządzanie projektami IT. Przewodnik po metodykach, Helion, 2010.
4. D. Pilone, N. Pitman, UML 2.0. Almanach, Helion, 2007.
5. A. Jaszkiewicz, Inżynieria oprogramowania, Helion, 1997.

Literatura uzupełniająca	1. M. E. Bays, Metodyka wprowadzania oprogramowania na rynek, WNT, 2001. 2. P. Graessle, H. Baumann, P. Baumann, UML 2.0 w akcji. Przewodnik oparty na projektach, Helion, 2006. 3. J. Schmuller, UML dla każdego, Helion, 2003. 4. M. Fowler, K. Beck, D. Roberts, E. Gamma, Refaktoryzacja. Ulepszanie struktury istniejącego kodu, WNT, 2006. 5. I. Sommerville, Software engineering, Pearson Education, 2004.	
Jednostka realizująca	Katedra Oprogramowania	**Data opracowania programu**
Program opracował(a)	dr inż. Krzysztof Jurczuk,prof. dr hab. inż. Marek Krętowski	2020.04.06