I. KARTA PRZEDMIOTU

1. Nazwa przedmiotu: **PROGRAMOWANIE OBIEKTOWE**
2. Kod przedmiotu: **Opo**
3. Jednostka prowadząca: **Wydział Mechaniczno-Elektryczny**
4. Kierunek: **Automatyka i Robotyka**
5. Specjalność: **Komputerowe wspomaganie automatyki i robotyki**
6. Moduł: **Moduł kierunkowy**
7. Poziom studiów: **I stopnia**
8. Forma studiów: **niestacjonarne**
9. Semestr studiów: **III**
10. Profil: **ogólnoakademicki**
11. Prowadzący: **dr hab. inż. Jerzy Garus**

CEL PRZEDMIOTU

C1	Zapoznanie z metodami opisu algorytmu dla potrzeb programowania obiektowego.
C2	Zapoznanie z zasadami programowania w języku C++.
C3	Nabycie umiejętności programowania w języku C++
C4	Nabycie umiejętności rozwiązywania podstawowych problemów inżynierskich w języku C++

WYMAGANIA WSTĘPNE W ZAKRESIE WIEDZY, UMIEJĘTNOŚCI I INNYCH KOMPETENCJI

1	Wiedza z zakresu matematyki i technologii informacyjnej.
2	Znajomość i umiejetność programowania w języku C.

EFEKTY KSZTAŁCENIA

EK1	Student zna środowisko programistyczne do tworzenia oprogramowania w języku C++.
EK2	Student zna zasady programowania obiektowego. Rozumie notację UML i potrafi ją zastosować praktycznie.
EK3	Posiada znajomość języka programowania C++ w zakresie podstawowym.
EK4	Potrafi samodzielnie opracowywać algorytmy.
EK5	Potrafi napisać i uruchomić złożony program w języku C++.

TREŚCI PROGRAMOWE

	WYKŁADY	Liczba godzin
W1	Historia programowania – Od instrukcji bitowych do programowania obiektowego.	**1**
W2	Platforma programowa Microsoft .NET i biblioteki predefiniowanych klas.	**1**
W3	Zrozumieć obiekty oraz notację UML.	**1**
W4	Tworzenie własnych klas.	**1**
W5	Dziedziczenie i polimorfizm	**1**
W6	Zasady projektowania obiektowego	**1**
	Razem	**6**

	ZAJĘCIA LABORATORYJNE	
L1	Zintegrowane środowisko programistyczne Visual Studio.	**2**
L2	Realizacja koncepcji hermetyzacji.	**2**
L3	Realizacja koncepcji dziedziczenia.	**2**
L4	Realizacja koncepcji polimorfizmu.	**2**
L5	Projekt własny.	**4**
	Razem	**12**

NARZĘDZIA DYDAKTYCZNE

1	Notebook z projektorem
2	Stanowiska komputerowe z oprogramowaniem dydaktycznym

SPOSOBY OCENY

FORMUJĄCA

F2	Odpowiedź ustna	EK1-EK3
F3	Wykonanie zadanie praktycznego	EK4-EK5

OBCIĄŻENIE PRACĄ STUDENTA

Forma aktywności	**Średnia liczba godzin na zrealizowanie aktywności**	
semestr	**III**	**razem**
Godziny kontaktowe z nauczycielem	18	18
Rozwiązywanie zadań domowych	17	17
Przygotowanie do wykładów i laboratoriów	20	20
Opracowanie sprawozdań z ćwiczeń laboratoryjnych	20	20
SUMA GODZIN W SEMESTRZE	**75**	**75**
PUNKTY ECTS W SEMESTRZE	**3**	**3**

LITERATURA

PODSTAWOWA

1	D. Farbaniec: Microsoft Visual Studio 2012. Programowanie w C# , Helion 2013.
2	B. Stroustrup: Język C++, WNT 2002
3	B. D. McLaughin, G. Pollice, D. West: Rusz głową! Analiza i projektowanie obiektowe, Helion 2010
4	A. Stellman, J. Greene: Rusz głową! C#, Helion 2011

PROWADZĄCY PRZEDMIOT

1	dr hab. inż. Jerzy Garus, j.garus@amw.gdynia.pl

Formy oceny

Efekt	Na ocenę 2	Na ocenę 3	Na ocenę 4	Na ocenę 5
EK1	*Student zna środowisko programistyczne do tworzenia oprogramowania w języku C++.*			
	Student nie zna środowiska programistycznego do tworzenia oprogramowania w języku C++. Nie potrafi napisać i uruchomić prostego programu w języku C++.	Student ałabo zna środowisko programistyczne do tworzenia oprogramowania w języku C++. Potrafi napisać i uruchomić prosty program w języku C++.	Student zna środowisko programistyczne do tworzenia oprogramowania w języku C++. Potrafi napisać i uruchomić prosty program w języku C++.	Student doskonale zna środowisko programistyczne do tworzenia oprogramowania w języku C++. Potrafi biegle napisać i uruchomić prosty program w języku C++.
EK2	*Student zna zasady programowania obiektowego. Rozumie notację UML i potrafi ją zastosować praktycznie.*			
	Student nie zna zasad programowania obiektowego. Nie rozumie notacji UML.	Student zna zasady programowania obiektowego. Rozumie notację UML, ale nie potrafi jej zastosować praktycznie.	Student zna zasady programowania obiektowego. Rozumie notację UML i potrafi ją zastosować praktycznie.	Student zna doskonale zasady programowania obiektowego. Rozumie notację UML i potrafi ją zastosować praktycznie.
EK3	*Posiada znajomość języka programowania C++ w zakresie podstawowym.*			
	Zna przeznaczenie i sposób deklaracji klasy. Nie rozumie pojęcia: składowe klasy, widoczność składowych, klasa i instancja klasy. konstruktor, destruktor; czas życia obiektu. Nie potrafi dokonać ich implementacji w środowisku programistycznym.	Zna przeznaczenie i sposób deklaracji klasy. Rozumie pojęcia: składowe klasy, widoczność składowych, klasa i instancja klasy. konstruktor, destruktor; czas życia obiektu. Słabo potrafi dokonać ich implementacji w środowisku programistycznym.	Zna przeznaczenie i sposób deklaracji klasy. Rozumie pojęcia: składowe klasy, widoczność składowych, klasa i instancja klasy. konstruktor, destruktor; czas życia obiektu. Rozumie pojęcia dziedziczenie i polimorfizm. Potrafi dokonać ich implementacji w środowisku programistycznym.	Zna przeznaczenie i sposób deklaracji klasy. Doskonale rozumie pojęcia: składowe klasy, widoczność składowych, klasa i instancja klasy. konstruktor, destruktor; czas życia obiektu. Rozumie pojęcia dziedziczenie i polimorfizm.Potrafi biegle dokonać ich implementacji w środowisku programistycznym.
EK4	*Potrafi samodzielnie opracowywać algorytmy.*			
	Student nie potrafi samodzielnie opracowywać algorytmy dla wybranych problemów inzynierskich.	Student potrafi małosamodzielnie opracowywać algorytmy dla wybranych problemów inżynierskich.	Student potrafi samodzielnie opracowywać algorytmy dla wybranych problemów inzynierskich.	Student potrafi doskonale opracowywać algorytmy dla wybranych problemów inzynierskich.
EK5	*Potrafi napisać i uruchomić złożony program w języku C++.*			
	Nie potrafi napisać i uruchomić złożony program w języku C++.	Słabo potrafi napisać i uruchomić złożony program w języku C++.	Potrafi napisać i uruchomić złożony program w języku C++.	Potrafi biegle napisać i uruchomić złożony program w języku C++.