

# I. KARTA PRZEDMIOTU

1. Nazwa przedmiotu: **PROGRAMOWANIE OBIEKTOWE**
2. Kod przedmiotu: **Ovi2**
3. Jednostka prowadząca: **Wydział Mechaniczno-Elektryczny**
4. Kierunek: **Mechatronika**
5. Specjalność: **Techniki Komputerowe w Mechatronice**
6. Moduł: **treści podstawowych**
7. Poziom studiów: **I stopnia**
8. Forma studiów: **niestacjonarne**
9. Semestr studiów: **II**
10. Profil: **praktyczny**
11. Prowadzący: **dr hab. inż. Jerzy Garus**

## CEL PRZEDMIOTU

<b>C1</b>	Zapoznanie z metodami opisu algorytmu dla potrzeb programowania obiektowego.
<b>C2</b>	Zapoznanie z zasadami programowania w języku C++/C#.
<b>C3</b>	Nabywanie umiejętności programowania w języku C++/C#.
<b>C4</b>	Nabywanie umiejętności rozwiązywania podstawowych problemów inżynierskich w języku C++/C#.

## WYMAGANIA WSTĘPNE W ZAKRESIE WIEDZY, UMIEJĘTNOŚCI I INNYCH KOMPETENCJI

<b>1</b>	Wiedza z zakresu matematyki i technologii informacyjnej.
<b>2</b>	Znajomość i umiejętność programowania w języku C.

## EFEKTY KSZTAŁCENIA

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

## TREŚCI PROGRAMOWE

WYKŁADY		Liczba godzin
<b>W1</b>	Historia programowania – Od instrukcji bitowych do programowania obiektowego.	<b>1</b>
<b>W2</b>	Platforma programowa Microsoft .NET i biblioteki predefiniowanych klas.	<b>1</b>
<b>W3</b>	Zrozumieć obiekty oraz notację UML.	<b>2</b>
<b>W4</b>	Tworzenie własnych klas.	<b>2</b>
<b>W5</b>	Dziedziczenie i polimorfizm	<b>2</b>
<b>W6</b>	Zasady projektowania obiektowego	<b>2</b>
	Razem	<b>10</b>
ZAJĘCIA LABORATORYJNE		
<b>L1</b>	Zintegrowane środowisko programistyczne Visual Studio.	<b>2</b>
<b>L2</b>	Różnice pomiędzy typami bezpośrednimi (value types) a typami referencyjnymi.	<b>2</b>
<b>L3</b>	Realizacja koncepcji hermetyzacji.	<b>2</b>
<b>L4</b>	Realizacja koncepcji dziedziczenia.	<b>2</b>
<b>L5</b>	Realizacja koncepcji polimorfizmu.	<b>2</b>
<b>L6</b>	Wzorce projektowe: Singleton i Strategia.	<b>2</b>

<b>L7</b>	Projekt własny.	<b>4</b>
<b>L8</b>	Zaliczenie przedmiotu	<b>4</b>
Razem		<b>20</b>

### NARZĘDZIA DYDAKTYCZNE

<b>1</b>	Notebook z projektorem
<b>2</b>	Stanowiska komputerowe z oprogramowaniem dydaktycznym

### SPOSOBY OCENY

#### FORMUJĄCA

<b>F2</b>	Odpowiedź ustna	EK1-EK3
<b>F3</b>	Wykonanie zadanie praktycznego	EK4-EK5

### OBCIĄŻENIE PRACĄ STUDENTA

Forma aktywności	Średnia liczba godzin na zrealizowanie aktywności	
	semestr	razem
Godziny kontaktowe z nauczycielem	30	30
Samodzielne opracowanie zagadnień	20	20
Rozwiązywanie zadań domowych	20	20
Przygotowanie do wykładów i laboratoriów	20	20
<b>SUMA GODZIN W SEMESTRZE</b>	<b>90</b>	<b>90</b>
<b>PUNKTY ECTS W SEMESTRZE</b>	<b>3</b>	<b>3</b>

### LITERATURA

#### PODSTAWOWA

<b>1</b>	D. Farbaniec: Microsoft Visual Studio 2012. Programowanie w C# , Helion 2013.
<b>2</b>	E. Freeman, E. Freeman, B. Bates, K. Sierra: Rusz głową! Wzorce projektowe, Helion 2010
<b>3</b>	B. D. McLaughlin, G. Pollice, D. West: Rusz głową! Analiza i projektowanie obiektowe, Helion 2010
<b>4</b>	A. Stellman, J. Greene: Rusz głową! C#, Helion 2011
<b>5</b>	B. Stroustrup: Język C++, WNT, Warszawa 2000

### PROWADZĄCY PRZEDMIOT

<b>1</b>	dr hab. inż. Jerzy Garus, j.garus@amw.gdynia.pl
----------	---

## Formy oceny

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