

KARTA KURSU

| | |
|-----------------|----------------------------------|
| Nazwa | Inżynieria oprogramowania |
| Nazwa w j. ang. | Software engineering |

| | | |
|---------------|---|----------------------|
| Koordynator | mgr inż. Witold Wilk | Zespół dydaktyczny |
| | | mgr inż. Witold Wilk |
| Punkcja ECTS* | st. stacjonarne: 3 st. niestacjonarne: 3 | |

Opis kursu (cele kształcenia)

Celem kursu jest zapoznanie studentów z procesem produkcji oprogramowania, wszystkimi jego fazami poczynając od fazy analizy wymagań, a kończąc na testowaniu gotowego produktu. Kurs prowadzony jest w języku polskim.

Warunki wstępne

| | |
|--------------|---|
| Wiedza | Podstawowe wiadomości z zakresu: algorytmiki, systemów operacyjnych i systemów wbudowanych. Znajomość języków C++ lub JAVA. |
| Umiejętności | Umiejętność konstruowania prostych algorytmów. Umiejętność programowania obiektowego w języku C++, JAVA lub C#. |
| Kursy | Algorytmy i struktury danych. Systemy operacyjne. Systemy wbudowane. Programowanie obiektowe |

Efekty uczenia się

| | Efekt uczenia się dla kursu | Odniesienie do efektów kierunkowych |
|--------|---|-------------------------------------|
| Wiedza | Po zakończeniu kursu student ma wiedzę na temat: W01: powstania inżynierii oprogramowania, jej definicji i celów. | K_W11 |
| | W02: procesów wytwarzania oprogramowania i ich modeli, modeli (paradygmaty) tworzenia oprogramowania (modele cyklu życia oprogramowania). | K_W11 K_W11 |
| | W03: narzędzi i środowiska wspomagającego tworzenie oprogramowania, wymagań, procesu inżynierii wymagań i ewolucji wymagań. | K_W13 |
| | W04: metod i model analizy(modelowania) oprogramowania: analizy strukturalnej i obiektowej (m.in. język UML). | K_W11 |
| | W05: wzorców projektowych | K_W11 K_W13 |
| | W06: wiadomości o walidacji i testowaniu oprogramowania oraz inspekcji kodu. | K_W11 |

| | Efekt uczenia się dla kursu | Odniesienie do efektów kierunkowych |
|--------------|---|-------------------------------------|
| Umiejętności | Po zakończeniu kursu student: U01: umie specyfikować wymagania stawiane oprogramowaniu oraz przeprowadzać ich przegląd. U02: szacuje nakład potrzebny na wytworzenie oprogramowania. U03: umie dokonać wyboru narzędzi wspomagających tworzenie oprogramowania (m.in. narzędzia CASE). Umie projektować oprogramowanie zgodnie z podejściem obiektowym lub strukturalnym. Potrafi implementować w np. języku C++, JAVA wybrane diagramy utworzone w notacji języka UML. U04: stosuje wzorce projektowe | K_U01 K_U01 K_U01 |
| | U05: dokonuje inspekcji kodu. Potrafi stosować podstawowe metody i strategię testowania oprogramowania, a także umie sporządzić plan i ewaluację testowania. | K_U01 K_U01 |

| | Efekt uczenia się dla kursu | Odniesienie do efektów kierunkowych |
|-----------------------|---|---|
| Kompetencje społeczne | Po zakończeniu kursu student: K01: potrafi określić możliwości wykorzystywania swojej wiedzy dotyczącej projektowania oprogramowania w pracy zawodowej. K02: rozumie konieczność kształcenia ustawicznego w szczególności w związku z dynamicznym rozwojem informatyki i nowych technologii. K03: potrafi współdziałać i pracować w grupie (zespole projektowym, programistycznym), przy jednoczesnym przestrzeganiu zasad etykiety zawodowej i netykiety. K04: rozumie społeczne aspekty stosowania zdobytej wiedzy i umiejętności, a także zna i stosuje kodeks etyczny zawodu informatyka. | K_K01 K_K01 K_K03 K_K05 K_K06 |
| | | |

Studia stacjonarne

| | | Organizacja | | | | | | | | | |
|---------------|------------|---------------------|--|---|--|----|--|---|--|---|---|
| Forma zajęć | Wykład (W) | Ćwiczenia w grupach | | | | | | | | | |
| | | A | | K | | L | | S | | P | E |
| Liczba godzin | 5 | | | | | 15 | | | | | |

Studia niestacjonarne

| | | Organizacja | | | | | | | | | |
|---------------|------------|---------------------|--|---|--|----|--|---|--|---|---|
| Forma zajęć | Wykład (W) | Ćwiczenia w grupach | | | | | | | | | |
| | | A | | K | | L | | S | | P | E |
| Liczba godzin | 10 | | | | | 15 | | | | | |

Opis metod prowadzenia zajęć

Podczas pracy laboratoryjnej studenci będą rozwiązywać problemy zadane przez prowadzącego zajęcia oraz opracowywać własne projekty.

Na ćwiczeniach na bieżąco weryfikowana będzie wiedza przekazywana podczas wykładów.

Formy sprawdzania efektów uczenia się

| | E – learning | Gry dydaktyczne | Ćwiczenia w szkole | Zajęcia terenowe | Praca laboratoryjna | Projekt indywidualny | Projekt grupowy | Udział w dyskusji | Referat | Praca pisemna (esej) | Egzamin ustny | Egzamin pisemny | Inne |
|-----|--------------|-----------------|--------------------|------------------|---------------------|----------------------|-----------------|-------------------|---------|----------------------|---------------|-----------------|------|
| W01 | | | | | | | | x | | | | x | |
| W02 | | | | | x | x | | | | | | x | |
| W03 | | | | | x | | | | | | | x | |
| W04 | | | | | x | x | | | | | | x | |
| W05 | | | | | x | | | | | | | x | |
| W06 | | | | | x | | | | | | | x | |
| U01 | | | | | x | x | | | | | | x | |
| U02 | | | | | x | x | | | | | | | |
| U03 | | | | | x | x | | | | | | x | |
| U04 | | | | | x | x | | | | | | x | |
| U05 | | | | | x | | x | | | | | | |
| K01 | | | | | x | | x | | | | | | |
| K02 | | | | | x | | x | | | | | | |
| K04 | | | | | x | | | x | | | | x | |

Kryteria oceny

Ocena końcowa zależna od ocen cząstkowych oraz systematyczności realizowanych zadań.

Kryterium uzyskania oceny dobrej lub bardzo dobrej (w zakresie zrealizowanym na zajęciach):

- 1) student zna i umie stosować podstawowe miary złożoności oprogramowania.
- 2) student umie wskazać elementy ewolucji oprogramowania
- 3) metody projektowania oprogramowania.
- 4) korzystanie z Application Programming Interface (API).
- 5) potrafi opracować plan przedsięwzięcia programistycznego
- 6) potrafi przeprowadzać przegląd projektu oprogramowania.
- 7) umie zarządzać konfiguracją oprogramowania

Uwagi

Treści merytoryczne (wykaz tematów)

1. Powstanie inżynierii oprogramowania, jej definicja i cele.
2. Procesy wytwarzania oprogramowania i ich modele, modele tworzenia oprogramowania (modele cyklu życia oprogramowania).
3. Narzędzia i środowiska wspomagające tworzenie oprogramowania
4. Metody i modele analizy(modelowania) oprogramowania: analiza strukturalna i obiektowa (m.in. język UML).
5. Stosowanie wzorców projektowych.
6. Wiadomości o walidacji i testowaniu oprogramowania oraz inspekcji kodu.

Wykaz literatury podstawowej

1. Ian Sommerville Inżynieria Oprogramowania, Wydawnictwo Naukowe PWN, Wydanie 10 (1 w WN PWN) Warszawa, 2020 - wybrane fragmenty.
2. K.Beck, A.Cynthia, Wydajne programowanie – Extreme Programming, Mikom, 2005 - wybrane fragmenty.
3. A. Cockburn, Jak pisać efektywne przypadki użycia, WNT, Warszawa 2004.
4. S.Covey, 7 nawyków skutecznego działania REBIS, 2002 - wybrane fragmenty
5. M.Fowler, K.Scott, UML w kropelce, LTP, 2002.
6. R. Pressman, Software Engineering, McGraw-Hill, New York 1997 - wybrane fragmenty
7. Kim Hamilton , Russell Miles, Learning UML 2.0, O'Reilly, April 2006.

Wykaz literatury uzupełniającej

1. Grady Booch, Rumbaugh James, Jacobson Ivar, UML przewodnik użytkownika, WNT, Warszawa 2002.
2. Wzorce projektowe: elementy oprogramowania obiektowego wielokrotnego użytku, Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides; z angielskiego przełożył Tomasz Walczak, słowo wstępne Grady'ego Boocha, Helion 2021.
3. Wzorce projektowe: leksykon kieszonkowy Daniel Krasnokucki, Helion Gliwice 2017.
4. Zwinne wytwarzanie oprogramowania : najlepsze zasady, wzorce i praktyki Robert C. Martin; [tłumaczenie: Radosław Meryk] Wydawnictwo Helion Gliwice 2017.
5. Jarosław Żeliński, Analiza biznesowa. Praktyczne modelowanie organizacji., Helion Gliwice 2017.

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) **studia stacjonarne**

| | | |
|--|--|----|
| Ilość godzin w kontakcie z prowadzącymi | Wykład | 5 |
| | Konwersatorium (ćwiczenia, laboratorium itd.) | 15 |
| | Pozostałe godziny kontaktu studenta z prowadzącym | 10 |
| Ilość godzin pracy studenta bez kontaktu z prowadzącymi | Lektura w ramach przygotowania do zajęć | 10 |
| | Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu | 10 |
| | Przygotowanie projektu lub prezentacji na podany temat (praca w grupie) | 10 |
| | Przygotowanie do egzaminu/zaliczenia | 15 |
| Ogółem bilans czasu pracy | | 75 |
| Ilość punktów ECTS w zależności od przyjętego przelicznika | | 3 |

Bilans godzinowy zgodny z CNPS (Całkowity Nakład Pracy Studenta) **studia niestacjonarne**

| | | |
|---|--|----|
| Ilość godzin w kontakcie z prowadzącymi | Wykład | 10 |
| | Konwersatorium (ćwiczenia, laboratorium itd.) | 15 |
| | Pozostałe godziny kontaktu studenta z prowadzącym | 5 |
| Ilość godzin pracy studenta bez kontaktu z prowadzącymi | Lektura w ramach przygotowania do zajęć | 15 |
| | Przygotowanie krótkiej pracy pisemnej lub referatu po zapoznaniu się z niezbędną literaturą przedmiotu | 5 |

| | | |
|--|--|----|
| | Przygotowanie projektu lub prezentacji na podany temat (praca w grupie) | 10 |
| | Przygotowanie do egzaminu/zaliczenia | 15 |
| | Ogółem bilans czasu pracy | 75 |
| | Ilość punktów ECTS w zależności od przyjętego przelicznika | 3 |