Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»

Кафедра біомедичної кібернетики

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Робоча програма навчальної дисципліни (Силабус)

Реквізити навчальної дисципліни

Рівень вищої освіти	***Перший (бакалаврський)***
Галузь знань	*12 Інформаційні технології*
Спеціальність	*122 Комп'ютерні науки*
Освітня програма	*Комп'ютерні технології в біології та медицині*
Статус дисципліни	*Нормативна*
Форма навчання	*Очна(денна)*
Рік підготовки, семестр	*2 курс, весняний семестр*
Обсяг дисципліни	*4,5 кредитів ЄКТС / 135 год.* *(32 год. – лекції, 40 год. – комп'ютерні практикуми, 63 год. – СРС)*
Семестровий контроль/ контрольні заходи	*екзамен, модульна контрольна робота (МКР)*
Розклад занять	*http://rozklad.kpi.ua*
Мова викладання	*Українська*
Інформація про керівника курсу / викладачів	*к.т.н. Алхімова Світлана Миколаївна,* *контактний телефон: +380674045083, e-mail: alkhimova.svitlana@lll.kpi.ua*
Розміщення курсу	Посилання на дистанційний ресурс Google classroom: https://classroom.google.com/c/NTg4Mzk5ODAxMzE4

Програма навчальної дисципліни

1. Опис навчальної дисципліни, її мета, предмет вивчання та результати навчання

Предметом навчальної дисципліни є система здатностей та умінь з основ об'єктно-орієнтованого проєктування та програмування і створення об'єктно-орієнтованих програмних застосунків мовою програмування високого рівня C++, які необхідні під час виконання обов'язків, виробничих функцій та типових задач діяльності фахівця; а також практичні навички використання об'єктно-орієнтованого підходу для створення повноцінного програмного забезпечення.

Метою навчальної дисципліни є формування знань з основ об'єктно-орієнтованого програмування, а також формування у студентів твердих практичних навичок щодо розробки програмних застосунків з використанням об'єктно-орієнтованого підходу; вивчення та практичного застосування принципів побудови та використання сучасних технологій розробки програмного забезпечення комп'ютерних систем на різних рівнях їх функціонування.

Програмні результати навчання.

Студенти після засвоєння навчальної дисципліни мають набути наступні компетентності:

Інтегральні компетентності

ІК Здатність розв'язувати складні спеціалізовані задачі та практичні проблеми у галузі комп'ютерних наук або у процесі навчання, що передбачає застосування теорій та методів комп'ютерних наук, інформаційних технологій і характеризується комплексністю та невизначеністю умов

Загальні компетентності

ЗК 2 Здатність застосовувати знання у практичних ситуаціях

ЗК 8 Здатність застосовувати знання у практичних ситуаціях

ЗК 11 Здатність приймати обґрунтовані рішення

Фахові компетентності

ФК 3 Здатність до логічного мислення, побудови логічних висновків, використання формальних мов і моделей алгоритмічних обчислень, проектування, розроблення й аналізу алгоритмів, оцінювання їх ефективності та складності, розв'язності та нерозв'язності алгоритмічних проблем для адекватного моделювання предметних областей і створення програмних та інформаційних систем.

ФК 8 Здатність проектувати та розробляти програмне забезпечення із застосуванням різних парадигм програмування: узагальненого, об'єктно-орієнтованого, функціонального, логічного, з відповідними моделями, методами й алгоритмами обчислень, структурами даних і механізмами управління.

Студенти після засвоєння навчальної дисципліни мають продемонструвати наступні програмні результати навчання

ПР 5 Проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій.

ПР 9 Розробляти програмні моделі предметних середовищ, вибирати парадигму програмування з позицій зручності та якості застосування для реалізації методів та алгоритмів розв'язання задач в галузі комп'ютерних наук.

2. Пререквізити та постреквізити дисципліни (місце в структурно-логічній схемі навчання за відповідною освітньою програмою)

В структурно-логічній схемі програми підготовки фахівця навчальна дисципліна «Об'єктно-орієнтовне програмування» входить до переліку нормативних дисциплін, циклу професійної підготовки

Пререквізити. Навчальна дисципліна базується на раніше засвоєних навчальних дисциплінах: «Алгоритмізація та програмування. Частина 1-2».

Постреквізити. Навчальна дисципліна забезпечує наступну дисципліну «Проектування інформаційних систем», а також є основою для підготовки дипломних робіт за спеціальністю та в подальшій практичній роботі за фахом.

3. Зміст навчальної дисципліни

Освітній компонент «Об'єктно-орієнтоване програмування»

Тема 1. Об'єктно-орієнтований підхід. Мова UML.

Об'єктно-орієнтований підхід до розробки програмного забезпечення. Основні складові об'єктно-орієнтованого підходу; об'єктно-орієнтовані програмування, проєктування та аналіз; складні системи; декомпозиція, абстрагування та ієрархія; основи об'єктної моделі. Об'єктно-орієнтований підхід у розробці складних систем. Застосування об'єктно-орієнтованого підходу до розробки програмного забезпечення. Використання UML в об'єктно-орієнтованому аналізі та проектуванні. Основи мови UML; різновиди UML діаграм; діаграми класів, об'єктів, послідовності. Принципи

проєктування класів; основні компоненти діаграми класів в UML; атрибути класу; операції класу; визначення відношень між класами на діаграмі класів.

Тема 2. Абстрагування даних та інкапсуляція.

Формат опису класу мовою C++; поля та методи класу; співвідношення між класом та його об’єктом. Абстрагування даних; поняття інкапсуляції; статус доступу до членів класу; специфікатори доступу public, private, protected; операції доступу до членів класу; методи для встановлення/отримання даних полів класу (set/get). Константні методи. Конструктор класу, його призначення; список ініціалізації; конструктор за замовченням; конструктори з параметрами; конструктор копіювання; проблема поверхневого копіювання; час життя об'єктів; деструктор, його призначення; випадки неявного виклику деструкторів.

Тема 3. Особливості роботи з об’єктами.

Визначення та робота з об’єктами, приклади. Неявний покажчик this. Константні об’єкти; використання специфікатора mutable. Поняття ієрархії; агрегація як ієрархія “part of” [“частина ...”]; відношення агрегації (колекція об’єктів); відношення композиції (агрегація за значенням).

Тема 4. Успадковування та поліморфізм

Успадковування як ієрархія “is-a” [“є”]; загальна форма успадковування. Базовий клас і класи нащадки; способи спадкування базового класу; використання спадкування. Перевизначення методів базових класів у класах нащадках. Невизначеності під час використання множинного спадкування. Порядок ініціалізації об’єктів під час виклику конструкторів. Поліморфізм, способи його реалізації мовою C++; статичний та динамічний поліморфізм. Реалізація динамічного поліморфізму; віртуальні функції. Диспетчеризація динамічних викликів; таблиця віртуальних функцій. Робота із віртуальними функціями. Виклики віртуальних функцій і права доступу до членів класу. Чисті віртуальні функції та абстрактні класи. Віртуальні деструктори/конструктори.

Тема 5. Перевантаження операцій та функцій

Статичні поля та методи класу; доступ до статичних членів класу. Дружні функції. Дружні класи. Перевантаження операцій; операції, що не можуть бути перевантажені; успадкування функцій для перевантажених операцій; специфіка перевантаження деяких операцій в мові C++. Перевантаження операцій за допомогою компонентних функцій класу. Перевантаження операцій за допомогою дружніх до класу функцій. Перевантажені функції.

Тема 6. Шаблони функцій та класів

Узагальненого програмування; шаблони функцій; конкретизації шаблона функції; спеціалізація шаблона функції. Шаблони класів, формат опису. Конкретизація шаблона класів. Спеціалізація шаблонних класів. Особливості роботи з параметрами шаблона класів. Статичні члени шаблона класів. Властивості параметрів шаблона. Моделі компіляції шаблонів.

Тема 7. Стандартна бібліотека шаблонів STL

Огляд стандартної бібліотеки шаблонів; загальна структура бібліотеки STL. Контейнери бібліотеки STL; типова організація класу контейнера. Ітератори бібліотеки STL. Алгоритми бібліотеки STL.

Тема 8. Виняткові ситуації

Обробка помилкових і нетипових ситуацій. Обробка помилок під час виконання програми за допомогою макросу assert. Механізм обробки винятків try-throw-catch. Формат try-throw-catch; синтаксис та семантика генерації та обробки виключень. Перехоплення всіх типів винятків. Класи винятків. Критерії пошуку відповідного catch-блока.

Тема 9. Класи потоків уведення та виведення

Потоки введення та виведення. Файлові потоки. файлове введення-виведення даних. Вбудовані C++ потоки. Класи потоків; класи стандартних потоків, їх ієрархія; заголовні файли бібліотеки вводу/виводу C++.

Тема 10. Статичні та динамічні бібліотеки

Поняття бібліотеки. Статичне і динамічне компонування. Статичні і динамічні бібліотеки. Динамічні бібліотеки, їх призначення; спільне використання. Способи завантаження динамічних бібліотек в адресний простір процесу. Точка входу до динамічних бібліотек.

4. Навчальні матеріали та ресурси

Освітній компонент «Об’єктно-орієнтоване програмування»

Базова

1. Постіл, С. Д. UML. Уніфікована мова моделювання інформаційних систем : навч. посіб. /

С. Д. Постіл ; Ун-т держ. фіск. служби України. - Ірпінь : Ун-т держ. фіск. служби України, 2019. - 321 с.
2. Основи об’єктно-орієнтованого програмування : навч. посібник / Гришанович Т. О., Глинчук Л. Я.; ВНУ імені Лесі Українки. – Луцьк : ВНУ імені Лесі Українки, 2022. – 120 с.
3. Васильєв, О.М. Програмування на С++ в прикладах і задачах : навч. посібник / О. М. Васильєв. – Київ : Ліра-К, 2020. – 382 с.
4. Kirk, D. R. Deciphering Object-Oriented Programming with C++: A practical, in-depth guide to implementing object-oriented design principles to create robust code / Dorothy R. Kirk. – Birmingham, UK. : Packt Publishing, 2022. – 594 p.
5. Об’єктно-орієнтоване програмування. Об’єктно-орієнтоване програмування мовою високого рівня С++ : метод. вказівки до викон. лаб. робіт / Уклад. С. М. Алхімова. – [2-ге вид.] – К. : НТУУ «КПІ», 2018. – 48 с.

Допоміжна

1. Грицюк Ю.І., Рак Т.Є. Об'єктно-орієнтоване програмування мовою С++ : навчальний посібник. – Львів : Вид-во Львівського ДУ БЖД, 2011. – 404 с.
2. Booch, G. The Unified Modeling Language User Guide / G. Booch, J. Rumbaugh, I. Jacobson. – [2nd ed.]. – Boston, MA : Addison-Wesley, 2017. – 504 p.
3. Booch, G. Object-Oriented Analysis and Design with Applications / Grady Booch, Robert A. Maksimchuk, Michael W. Engle, Bobbi J. Young, Jim Conallen, Kelli A. Houston. – [3rd ed.]. – Boston, MA : Addison Wesley, 2007. – 694 p.
4. Gomaa, H. Designing Concurrent, Distributed, and Real-time Applications with UML / Hassan Gomaa ; Addison-Wesley object technology series. – Boston, MA : Addison Wesley Professional, 2000. – 816 p.
5. Кравець, П. О. Об’єктно-орієнтоване програмування : навч. посібн. / П. О. Кравець. – Львів : Вид-во Львів. політехніки, 2012. – 624 с.
6. С++. Теорія та практика : навч. посібник / [О. Г. Трофименко, Ю. В. Прокоп, І. Г. Швайко, Л. М. Буката та ін.] ; за ред. О. Г. Трофименко. – Одеса: ОНАЗ ім.. О.С. Попова, 2011. – 588 с.
7. Дудзяний, І. М. Об’єктно-орієнтоване моделювання програмних систем : навч. посібн. / І. М. Дудзяний. – Львів : Вид. Центр ЛНУ імені Івана Франка, 2007. – 108 с.
8. Бублик, В. В. Об’єктно-орієнтоване програмування : підручник / В.В. Бублик. – К.: ІТкнига, 2015. – 624 с.
9. Сопронюк, Т. М. Об’єктно-орієнтоване програмування на С++ : навч. посіб. / Т. М. Сопронюк ; Чернів. нац. ун-т ім. Юрія Федьковича. – Чернівці : Рута, 2014. – 175 с.
10. Gamma, E. Design Patterns: Elements of Reusable Object-Oriented Software / E. Gamma, R. Helm, R. Johnson, and J. Vlissides. – [1st ed.]. – Boston, MA : Addison-Wesley, 1994. – 416 p.
11. Davidson, J. Beautiful C++: 30 Core Guidelines for Writing Clean, Safe, and Fast Code / J. Davidson, K. Gregory. – Boston, MA : Addison Wesley Professional, 2021. – 352 p.
12. Stroustrup, B. C++ Programming Language / Bjarne Stroustrup. – [3rd ed.]. – Boston, MA : Addison Wesley Professional, 2000. – 1030 p.
13. Stroustrup, B. Programming : principles and practice using C++ / Bjarne Stroustrup. – [2nd ed.]. – Boston, MA : Addison Wesley Professional, 2014. – 1312 p.
14. Пелешко, Д. Д. Об’єктні технології С++11 : навч. посібн. / Д. Д. Пелешко, В. М. Теслюк. – Львів : Вид-во Львів. політехніки, 2013. – 360 с.
15. Meyers, S. Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library / Scott Meyers. – Boston, MA : Addison Wesley Professional, 2001. – 288 p.
16. Lischner, R. STL Pocket Reference/ Ray Lischner. – Sebastopol, California : O'Reilly Media, Incorporated, 2003. – 128 p.
17. Josuttis, N. C++ Standard Library, The: A Tutorial and Reference / Nicolai Josuttis. – [2nd ed.]. – Boston, MA : Addison Wesley Professional, 2012. – 1136 p.
18. Cormen, T. H. Introduction to Algorithms / Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. – Cambridge, Massachusetts : The MIT Press, 2009. – 1292 p.
19. Коротєєва, Т. О. Алгоритми та структури даних: навч. посіб. / Т.О. Коротєєва. – Львів : Видавництво Львівської політехніки, 2014. – 280 с.
20. Ткачук, В. М. Алгоритми і структура даних: навч. посіб / В.М.Ткачук. – Івано-Франківськ : Видавництво Прикарпатського національного університету імені Василя Стефаника, 2016. –

286 с.

Інформаційні ресурси

1. Інтернет-портал організаційно-методичного забезпечення дистанційних олімпіад з програмування для обдарованої молоді навчальних закладів України : [Електронний ресурс]. – Режим доступу : http://www.e-olimp.com.ua/ua
2. International Standard ISO/IEC 14882:2014(E) – Programming Language C++ : [Електронний ресурс]. – Режим доступу : https://isocpp.org/std/the-standard
3. Google C++ Style Guide [Електронний ресурс]. – Режим доступу: https://google.github.io/styleguide/cppguide.html.
4. The Stanford University C++ Style Guide [Електронний ресурс]. – Режим доступу: https://hownot2code.com/2017/01/18/the-stanford-university-c-style-guide/

Навчальний контент

5. Методика опанування навчальної дисципліни (освітнього компонента) і самостійна робота студента

Освітній компонент «Об'єктно-орієнтоване програмування»

Лекції

Лекція 1. Об'єктно-орієнтований підхід

Питання, що розглядаються:

1) об'єктно-орієнтований підхід до розробки програмного забезпечення;
2) основні складові об'єктно-орієнтованого підходу;
3) об'єктно-орієнтований підхід у розробці складних систем;
4) застосування об'єктно-орієнтованого підходу до розробки програмного забезпечення.

Література:

основна – [1, 3(С. 59-105, 106-150)];
додаткова – [1 (С. 20-50, 51-66, 100-116), 2-4, 10].

Завдання на СРС:

вивчити конспект лекції; визначити зміст проєктування та еволюцію об'єктної моделі; ознайомитися з принципами побудови об'єктної моделі предметної області; визначити основні парадигми об'єктно-орієнтованого стилю програмування; провести аналіз основних риси спадкування, інкапсуляції та поліморфізму; навчитися визначати характеристики в задачах предметної області для проведення абстрагування даних; визначити основні складові частини об'єктно-орієнтованого стилю та навчитися чітко визначати складові інтерфейсу та реалізації класу; знати специфіку застосування об'єктно-орієнтованого підходу до розробки програмного забезпечення.

Лекція 2. Універсальна мова моделювання UML. Діаграми класів

Питання, що розглядаються:

1) мова моделювання;
2) визначення класів на діаграмі класів;
3) атрибути класу;
4) операції класу;
5) визначення відношень між класами на діаграмі класів.

Література:

основна – [1 (С.223-238, 239-244, 271-273)];
додаткова – [1 (С. 195-207), 2-4, 10].

Завдання на СРС:

вивчити конспект лекції; ознайомитися із основами універсальної мови моделювання UML та основними видами діаграм (варіантів використання, класів, послідовності, об'єктів), що використовується в об'єктно-орієнтованому проектуванні; знати специфіку розробки діаграми класів, вміти проектувати відношення між класами засобами мови UML.

Лекція 3. Базові поняття класу

Питання, що розглядаються:

1) визначення класу мовою C++;
2) механізми доступу до членів класу;
3) концепція інкапсуляції;
4) *set-* і *get*-методи класу.

Література:

основна – [2 (С. 29-54), 3 (С. 217-260)];
додаткова – [1 (С. 51-66, 88-99), 5 (С. 16-44)].

Завдання на СРС:

вивчити конспект лекції; знати як визначати нові абстрактні типи даних користувача мовою С++ за допомогою створення класів, структур, об'єднань; визначити відмінності структур і об'єднань від класів; вміти абстрагувати дані та їх поведінку в предметній області для визначення списку полів та алгоритмів їх опрацювання в методах розроблюваних класів; знати механізм створення екземплярів класів в програмах, що написані мовою С++; визначити співвідношення між класом та його об'єктами (що задає клас, що реалізує об'єкт); детально розглянути принцип інкапсуляції даних в об'єктно-орієнтованому підході; знати як задавати статус доступу до членів класу та вивчити особливості використання специфікаторів доступу public, private, protected; знати як отримати доступ до членів класу через його екземпляр, використовуючи операції доступу; проаналізувати відмінності в роботі операцій доступу «.» та «–>» та вміти переходити від використання операції «->» до операції «.»; знати як створювати та використовувати методи для встановлення/отримання даних полів класу.

Лекція 4. Спеціальні методи класу

Питання, що розглядаються:

1) конструктор як спеціальні методи класу;
2) списки ініціалізації;
3) конструктор за замовчуванням;
4) конструктор з параметрами;
5) деструктори як спеціальні методи класу;
6) побітові копії і конструктор копіювання;
7) робота з полями всередині конструкторів та деструкторів.

Література:

основна – [2 (С. 59-79), 3 (С. 217-260)];
додаткова – [1 (С. 67-73), 5 (С. 45-54)].

Завдання на СРС:

вивчити конспект лекції; знати призначенням і правилами створення та використання конструкторів класу; знати принцип ініціалізації полів класу, використовуючи списки ініціалізації; знати специфіку роботи конструкторів за замовченням; вивчити особливості розробки конструкторів з параметрами та навчитися їх використовувати; проаналізувати роботу конструктора копіювання та випадки його виклику під час ініціалізації полів одного об'єкта значенням полів іншого об'єкта класу; знати причини виникнення проблеми поверхневого копіювання; проаналізувати час життя об'єкта; ознайомитися з призначенням і правилами створення та використання деструкторів класу; знати випадки неявного виклику деструкторів.

Лекція 5. Особливості роботи з об'єктами

Питання, що розглядаються:

1) неявний покажчик *this*;
2) константні об'єкти;
3) масиви об'єктів.

Література:

основна – [2 (С. 29-54), 3 (С. 217-260)];
додаткова – [1 (С. 51-66, 88-99), 5].

Завдання на СРС:

вивчити конспект лекції; знати специфіку використання покажчика this в методах класу; знати як організувати код для об'єднання викликів методів в один ланцюжок; знати як визначати константні об'єкти, специфіку роботи із ними; знати призначення використання специфікатора mutable; знати як організувати код для роботи із статичними і динамічними масивами об'єктів; знати специфіку ініціалізації елементів у масивах об'єктів; знати правила організації коду під час алокації та звільнення пам'яті під динамічні масиви об'єктів; знати специфіку агрегації як виду відносин між класами в мові програмування С++ (агрегація як ієрархія "part of" ["частина ..."]); вміти розрізняти відношення агрегації (колекція об'єктів) від відношення композиції (агрегація за значенням).

Лекція 6. Спадкування

Питання, що розглядаються:

1) концепція спадкування;
2) способи спадкування базового класу;
3) використання спадкування;
4) перевизначення методів базових класів у класах нащадках;
5) невизначеності під час використання множинного спадкування;
6) порядок ініціалізації об'єктів під час виклику конструкторів

Література:

основна – [1 (С. 116-144), 2 (С.106-150), 4 (С. 361-410)];
додаткова – [1 (С. 131-137, 141-148), 5, 7 (С. 117-130, 138-140, 180-193, 207-229, 254-261)].

Завдання на СРС:

вивчити конспект лекції; проаналізувати специфіку спадкування як виду відносин між класами в мові програмування С++ (спадкування як ієрархія "is-a" ["є"]); навчитися розрізняти класи-колекції від відносин композиції; знати загальний формат визначення успадкування класів; знати, що таке базовий клас і клас нащадок; знати особливості вживання специфікаторів доступу, що задають різні способи спадкування класів; знати принципи простого та множинного спадкування в об'єктно-орієнтованому програмуванні; знати специфіку роботи конструкторів та деструкторів під час використання принципів

успадкування класу; знати як обмежити спадкування для класу; знати специфіку перевизначення методів базових класів у класах нащадках; знати дві невизначеності під час використання множинного спадкування, як вони вирішуються; знати порядок ініціалізації об'єктів під час виклику конструкторів; навчитися будувати ієрархії класів в предметній області.

Лекція 7. Поліморфізму. Реалізація динамічного поліморфізму

Питання, що розглядаються:

1) концепція поліморфізму, способи його реалізації мовою С++;
2) реалізація динамічного поліморфізму;
3) диспетчеризація динамічних викликів.

Література:

основна – [3 (С. 145-161), 4 (С. 476-535)];
додаткова – [2 (С. 230-252)].

Завдання на СРС:

вивчити конспект лекції; знати поняттям поліморфізму; знати основні способи реалізації поліморфізму мовою програмування С++; розуміти поняттями раннього (статичного) та пізнього (динамічного) зв'язування; знати механізм реалізації динамічного поліморфізму з використанням віртуальних функцій; знати специфіку роботи таблиці віртуальних функцій та покажчика на таблицю віртуальних функцій.

Лекція 8. Віртуальні функції та абстрактні класи.

Питання, що розглядаються:

1) робота із віртуальними функціями;
2) виклики віртуальних функцій і права доступу до членів класу;
3) чисті віртуальні функції та абстрактні класи;
4) віртуальні деструктори/конструктори.

Література:

основна – [3 (С. 145-161), 4 (С. 476-535)];
додаткова – [2 (С. 230-252, 254-261), 7-9].

Завдання на СРС:

вивчити конспект лекції; вивчити правила створення та використання віртуальних функцій в мові програмування С++; ознайомитися з призначенням і вивчити правилами створення та використання абстрактних класів в мові програмування С++; розглянути механізм визначення чистих віртуальних функцій за допомогою конструкції «=0».

Лекція 9. Статичні члени класів, дружні функції та класи.

Питання, що розглядаються:

1) статичні поля класу;
2) статичні методи класу;
3) дружні функції;
4) дружні класи.

Література:

основна – [3 (С. 55-58), 4 (С. 349-352)];
додаткова – [2 (С. 55-60), 11].

Завдання на СРС:

вивчити конспект лекції; ознайомитися з механізмом розподілення пам'яті в процесі роботи із об'єктами класу; вивчити призначенням і правилами створення та використання статичних членів класу; вивчити призначення і правила створення та використання дружніх функцій та класів.

Лекція 10. Перевантаження операцій та функцій.

Питання, що розглядаються:

1) перевантаження операцій;
2) перевантаження операцій за допомогою компонентних функцій класу;
3) перевантаження операцій за допомогою дружніх до класу функцій;
4) перевантажені функції.

Література:

основна – [1 (С. 80-115), 3 (С. 312-360)];
додаткова – [2 (С. 65-90), 11].

Завдання на СРС:

вивчити конспект лекції; ознайомитися з призначенням і правилами створення та використання перевантаження методів базового класу в методах класів-нащадків; вивчити специфіку перевантаження операцій за допомогою компонентних функцій класу та за допомогою глобальних (дружніх до класу) функцій; вивчити операції, що не можуть бути перевантажені; ознайомитися і знати можливості успадкування функцій для перевантаження операцій; вивчити специфіку перевантаження операцій привласнення, індексування, інкремента і декремента префіксних і постфіксних операцій; вивчити призначення і правила створення та використання перевантаження функцій.

Лекція 11. Шаблони функцій.

Питання, що розглядаються:

1) узагальнене (шаблонне) програмування;
2) шаблони функцій, формат опису;
3) конкретизації шаблона функції;

4) спеціалізація шаблона функції.

Література:

основна – [1 (С. 162-170), 3, 4];

додаткова – [2 (С. 307-335), 11-13].

Завдання на СРС:

вивчити конспект лекції; ознайомитися з основними принципами узагальненого програмування в мові С++; ознайомитися з призначенням і вивчити правилами створення та використання шаблонів функції; проаналізувати механізм та навчитися використовувати на практиці конкретизацію та спеціалізацію шаблона функції.

Лекція 12. Шаблони класів.

Питання, що розглядаються:

1) шаблони класів, формат опису;
2) конкретизація шаблона класів;
3) спеціалізація шаблонної класів;
4) особливості роботи з параметрами шаблона класів;
5) статичні члени шаблона класів;
6) властивості параметрів шаблона;
7) моделі компіляції шаблонів.

Література:

основна – [1 (С. 171-181) , 3, 4];

додаткова – [2 (С. 278-306), 11-13].

Завдання на СРС:

вивчити конспект лекції; ознайомитися з призначенням і вивчити правилами створення та використання шаблонів класу; проаналізувати механізм та навчитися використовувати на практиці конкретизацію та спеціалізацію шаблона класу; визначити призначення і навчитися створювати та використовувати параметри за замовчуванням та статичні члени шаблона класу; проаналізувати властивості параметрів шаблона.

Лекція 13. Стандартна бібліотека шаблонів STL.

Питання, що розглядаються:

1) огляд стандартної бібліотеки шаблонів;
2) контейнери бібліотеки STL;
3) ітератори бібліотеки STL;
4) алгоритми бібліотеки STL.

Література:

основна – [1 (С. 287-332), 3 (С. 681-751)];

додаткова – [3 (С. 52-82), 15, 16, 18-20].

Завдання на СРС:

вивчити конспект лекції; ознайомитися структурою та призначенням основних сутностей бібліотеки шаблонів STL; проаналізувати організацію різних видів контейнерів STL; ознайомитися із стандартними алгоритмами бібліотеки STL та прикладами використання алгоритмів різних груп (алгоритми, що не змінюють порядок слідування елементів контейнера; алгоритми, що змінюють порядок слідування елементів контейнера; алгоритми сортування; алгоритми пошуку; алгоритми поділу послідовності; алгоритми роботи з множинами; алгоритми роботи з купою; алгоритми для виконання операцій мінімуму і максимуму; алгоритми для виконання операцій порівняння; алгоритми для виконання операцій перестановок; чисельні алгоритми); ознайомитися з призначенням і вивчити правилами створення та використання п’яти категорій ітераторів (ітератори вводу, ітератори виводу, односпрямовані ітератори, двоспрямовані ітератори та ітератори довільного доступу); ознайомитися із предикатами та функціональними об’єктами (функторами) і вивчити правила їх використання.

Лекція 14. Обробка виняткових ситуацій.

Питання, що розглядаються:

1) обробка помилкових і нетипових ситуацій;
2) обробка помилок під час виконання програми за допомогою макросу *assert*;
3) механізм обробки винятків *try-throw-catch*;
4) формат *try-throw-catch*;
5) перехоплення всіх типів винятків;
6) класи винятків;
7) критерії пошуку відповідного *catch*-блока.

Література:

основна – [2, 3];

додаткова – [1 (С. 74-77), 2 (С. 171-206)].

Завдання на СРС:

вивчити конспект лекції; ознайомитися з загальними принципами механізму обробки виняткових ситуацій; визначити основні особливостями обробки виняткових ситуацій в мові програмування С++; вивчити, як обробити помилки під час виконання програми за допомогою макросу assert; вивчити синтаксис та семантику генерації та обробки винятків в мові програмування С++; вивчити, як перехопити

всі типи винятків; вивчити особливості розробки класів винятків; знати критерії пошуку відповідного catch-блока; вивчити функції, глобальні змінні та стандартні класи підтримки механізму винятків.

Лекція 15. Директиви препроцесора і класи потоків уведення та виведення.

Питання, що розглядаються:

1) директиви препроцесора;
2) потоки введення та виведення;
3) файлові потоки;
4) вбудовані C++ потоки;
5) класи потоків.

Література:

основна – [3 (С. 203-239), 4 (С. 536-586)];
додаткова – [14 (С. 273-340),17].

Завдання на СРС:

вивчити конспект лекції; вивчити особливості використання директив препроцесора; ознайомитися з поняттям потоків в мові програмування C++; ознайомитися із ієрархією та специфікою класів потоків (ios – базовий потоковий клас, streambuf – клас буферизації потоків, istream – клас потоку введення, ostream – клас потоку виведення, iostream – клас двоспрямованого потоку, iostream_withassign – клас потоку з перевизначенням операції привласнення, istrstream – клас строкового потоку введення, ostrstream – клас строкового потоку виведення, strstream – клас двоспрямованого строкового потоку, ifstream – клас файлового потоку введення, ofstream – клас файлового потоку виведення, fstream клас двоспрямованого файлового потоку); навчитися проводити форматування даних з використанням методів класу ios; навчитися використовувати маніпулятори введення-виведення даних; визначити особливості механізму перевантаження операцій введення/виведення даних; навчитися виконувати відкриття та закриття файлу, зчитування та запис даних до файлу, перевіряти статус введення-виведення даних файлу; ознайомитися з прикладами використання довільного доступу до вмісту файлу.

Лекція 16. Статичні та динамічні бібліотеки.

Питання, що розглядаються:

1) поняття бібліотеки;
2) статичне і динамічне компонування;
3) динамічні бібліотеки, їх призначення;
4) способи завантаження динамічних бібліотек в адресний простір процесу;
5) точка входу до динамічної бібліотек.

Література:

основна – [4 (С. 365-386)];
додаткова – [11, 14].

Завдання на СРС:

вивчити конспект лекції; ознайомитися із поняттями статичної та динамічної бібліотек та встановити основні їх відмінності; визначити призначенням і особливості розробки DLL-бібліотек; вивчити способи завантаження динамічних бібліотек в адресний простір процесу, детально розглянути особливості «зв'язування» DLL-бібліотеки з виконуваним файлом: неявне (під час завантаження), явне (під час виконання); вивчити складові DLL-бібліотеки: функція входу (DllMain).

Комп’ютерні практикуми (Лабораторні роботи)

Основна ціль комп’ютерних практикумів полягає в опануванні студентами об’єктно-орієнтованого підходу у створенні програмного забезпечення, а саме – ознайомленні студентів із описом і принципами побудови складних систем; наданні уявлення про етапи створення складних систем, використовуючи об’єктно-орієнтований підхід (об’єктно-орієнтований аналіз, об’єктно-орієнтоване проєктування, об’єктно-орієнтоване програмування); формуванні у студентів навиків опису програм і постановки задач; формуванні у студентів абстрактного мислення, що повинне допомагати рішенню прикладних задач, пов’язаних з різними галузями знань.

Комп’ютерний практикум 1. Реалізація класів, конструктори та деструктори (проектування).

Питання, що розглядаються:

1) проектування класів мовою UML;
2) основні конструкції, що використовуються при проектування класів;
3) атрибути та операції діаграми класів.

Література:

основна – [1 (С. 29-54, 59-79), 3 (С. 217-260)];
додаткова – [1 (С. 51-73, 88-99), 2 (С. 16-44, 45-54)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти спроектувати діаграму класів мовою UML, які пов’язані відношенням залежності.

Комп'ютерний практикум 2. Реалізація класів, конструктори та деструктори (реалізація).

Питання, що розглядаються:

1) створення класів та об'єктів мовою програмування С++;
2) основні конструкції, що використовуються при побудові класів і об'єктів;
3) інкапсуляція;
4) конструктори і деструктори користувача;
5) відкриті й закриті члени класу;
6) реалізація методів класу.

Література:

основна – [1 (С. 29-54, 59-79), 3 (С. 217-260)];
додаткова – [1 (С. 51-73, 88-99), 2 (С. 16-44, 45-54)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: що таке клас, навіщо він потрібен; що таке об'єкт; як пов'язані між собою класи і об'єкти в програмі; що таке інкапсуляція; за рахунок чого реалізується захист від несанкціонованого доступу до даних; чим відрізняються поля від змінних; що визначають методи класу; перелічити, чим можуть бути поля класу; що таке конструктор, навіщо він потрібен; що таке деструктор, навіщо він потрібен.

Комп'ютерний практикум 3. Реалізація класів, конструктори та деструктори (тестування).

Питання, що розглядаються:

1) тестування програм, що містять в коді класи та об'єкти;
2) перевірка інкапсуляції даних;
3) перевірка конструкторів і деструкторів користувача.

Література:

основна – [1 (С. 29-54, 59-79), 3 (С. 217-260)];
додаткова – [1 (С. 51-73, 88-99), 2 (С. 16-44, 45-54)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти тестувати програмні застосунки, в коді яких є класи і об'єкти.

Комп'ютерний практикум 4. Робота з об'єктами (проектування).

Питання, що розглядаються:

1) особливості проектування роботи з об'єктами в мові UML;
2) відношення асоціації в UML;
3) відношення агрегації в UML;
4) відношення композиції в UML;
5) поняття потужності відношення асоціації та її різновидів;
6) проектування обробки масивів об'єктів.

Література:

основна – [1 (С. 59-79, 116-144), 3 (С. 106-150, 217-260, 361-410)];
додаткова – [1 (С. 67-73, 131-137, 141-148), 2 (С. 45-54,117-130, 138-140, 180-193, 207-229, 254-261)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою практичного заняття, щоб вміти спроектувати діаграму об'єктів мовою UML.

Комп'ютерний практикум 5. Робота з об'єктами (реалізація та тестування).

Питання, що розглядаються:

1) особливості роботи з об'єктами в мові програмування С++;
2) створення об'єктів та масивів об'єктів, обробки їх відповідно до запитів користувача;
3) особливостями використання покажчика this.

Література:

основна – [1 (С. 59-79, 116-144), 3 (С. 106-150, 217-260, 361-410)];
додаткова – [1 (С. 67-73, 131-137, 141-148), 2 (С. 45-54,117-130, 138-140, 180-193, 207-229, 254-261)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: що таке покажчик this; навести приклад, в якому не можна організувати код без використання покажчика this; два способи ініціалізації об'єктів у програмі; як ініціалізувати масив об'єктів; чому не можна ініціалізувати поле, що є масивом, під час визначення класу; перелічити випадки, коли один клас використовує інший; які є види відношень між класами; які види відношень між класами визначені в мові моделювання UML; що означає оператор (::).

Комп'ютерний практикум 6. Успадкування та поліморфізм (проектування).

Питання, що розглядаються:

1) відношення узагальнення в UML;
2) проектування діаграм класів, пов'язаних ієрархічно.

Література:

основна – [1 (С. 145-161), 3 (С. 476-535)];
додаткова – [2 (С. 230-252, 254-261)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти спроектувати діаграму класів із відношенням узагальнення мовою UML.

Комп'ютерний практикум 7. Успадкування та поліморфізм (реалізація).

Питання, що розглядаються:

1) успадкуванням мовою C++;
2) особливості роботи з механізмом успадкування класів в мові C++;
3) правилами написання та використання віртуальних функцій в мові C++;
4) освоїти принципи побудови ланцюжків класів, пов'язаних ієрархічно.

Література:

основна – [1 (С. 145-161), 3 (С. 476-535)];
додаткова – [2 (С. 230-252, 254-261)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: що означає поняття спадкування; який клас називається базовим, а який клас є нащадком; скільки базових класів може мати похідний клас; чи може похідний клас бути базовим; як змінюється доступ до елементів базового класу при спадкуванні з різними специфікаторами доступу: з розділів класу, із програми, з інших класів; у чому різниця між простим і множинним спадкуванням; чи можна з класу-нащадка одержати доступ до private частини базового класу, якщо специфікатор доступу при спадкуванні private; чи успадковуються конструктори, деструктори; що таке віртуальна функція; чи можна віртуальну функцію визначити як static, відповідь пояснити.

Комп'ютерний практикум 8. Успадкування та поліморфізм (тестування).

Питання, що розглядаються:

1) тестування програмних застосунків із успадкуванням;
2) тестування коду із використанням віртуальних функцій в мові C++.

Література:

основна – [1 (С. 145-161), 3 (С. 476-535)];
додаткова – [2 (С. 230-252, 254-261)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти тестувати програмні застосунки, в коді яких містяться ієрархічні структури класів і використовуються віртуальні функції.

Комп'ютерний практикум 9. Перевантаження операцій (проектування).

Питання, що розглядаються:

1) запис перевантаженого оператора на діаграмі класів UML;
2) проектування дружніх функцій та класів.

Література:

основна – [1 (С. 80-115), 3 (С. 312-360)];
додаткова – [2 (С. 65-90)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти спроектувати діаграму класів із перевантаженими операціями мовою UML.

Комп'ютерний практикум 10. Перевантаження операцій (реалізація).

Питання, що розглядаються:

1) можливості перевантаження операцій в мові C++;
2) властивості та обмеження перевантаження операцій як компонентних функцій класів;
3) властивості та обмеження перевантаження операцій та як дружніх до класів глобальних функцій;
4) перевантаження унарних та бінарних операцій.

Література:

основна – [1 (С. 80-115), 3 (С. 312-360)];
додаткова – [2 (С. 65-90)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: навіщо потрібне перевантаження операторів; якими двома різними способами визначаються перевантажені операції.

Комп'ютерний практикум 11. Перевантаження операцій (тестування).

Питання, що розглядаються:

1) тестування перевантажених операцій в мові C++;
2) особливості тестування перевантаження унарних та бінарних операцій.

Література:

основна – [1 (С. 80-115), 3 (С. 312-360)];
додаткова – [2 (С. 65-90)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: які оператори мови C++ не можуть бути перевантажені; які операції не можна перевантажити за допомогою глобальної дружньої функції; у яких випадках операцію можна перевантажити тільки за допомогою глобальної функції; у яких випадках глобальна операція-функція

повинна бути дружньою; чи обов'язковий у функції operator параметр типу "клас" або"посилання на клас"; чи можна успадкувати або повторно перевантажити в похідному класі операцію, що була перевантажена в базовому класі; чому не можна перевантажувати деструктори; скільки необхідно передати параметрів функції-методу класу, що реалізує перевантаження бінарної операції.

Комп'ютерний практикум 12. Шаблони (проектування).

Питання, що розглядаються:

1) параметризовані класи на діаграмах класів UML;
2) проектування інстанціювання класів.

Література:

основна – [1 (С. 162-181)];
додаткова – [2 (С. 278-335)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти спроектувати діаграму параметризованих класів із відношенням інстанціювання мовою UML.

Комп'ютерний практикум 13. Шаблони (реалізація).

Питання, що розглядаються:

1) механізми використання шаблонів функцій та шаблонів класів;
2) створення та використання параметризованих функцій;
3) створення та використання параметризованих класів.

Література:

основна – [1 (С. 162-181)];
додаткова – [2 (С. 278-335)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: для чого використовують шаблони; що таке шаблон класів, як його визначити в С++; що таке шаблон функцій, як його визначити в С++; що таке конкретизація шаблона; в яких випадках необхідне виконання явної конкретизації шаблона; що таке спеціалізація шаблона, коли її необхідно виконувати; для чого використовують параметри за замовченням для шаблону класів; чи можна викликати параметризовану функцію без параметрів; чи може бути порожнім список параметрів шаблона; чи можна за допомогою шаблона створити функцію з таким самим ім'ям, як і в явно визначеної функції; чи можуть шаблони класів містити віртуальні функції.

Комп'ютерний практикум 14. Шаблони (тестування).

Питання, що розглядаються:

1) тестування програмних застосунків із використанням шаблонів функцій та шаблонів класів;
2) тестування інстанціювання шаблона.

Література:

основна – [1 (С. 162-181)];
додаткова – [2 (С. 278-335)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти тестувати програмні застосунки, в коді яких містяться шаблони функцій та шаблони класів.

Комп'ютерний практикум 15. Робота з бібліотекою STL (проектування).

Питання, що розглядаються:

1) проектування із використанням класів із бібліотеки STL;
2) використання в проектуванні контейнерів STL (vector, list, stack, queue, priority queue, deque, set, multiset, map, multimap).

Література:

основна – [1 (С. 287-332), 3 (С. 681-751)];
додаткова – [3 (С. 52-82)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти спроектувати діаграму параметризованих класів із використанням стандартних класів бібліотеки STL мовою UML.

Комп'ютерний практикум 16. Робота з бібліотекою STL (реалізація).

Питання, що розглядаються:

1) основні сутності бібліотеки шаблонів STL;
2) різні види контейнерів STL (vector, list, stack, queue, priority queue, deque, set, multiset, map, multimap);
3) ітератори та стандартні алгоритми їх обробки, що надає бібліотека STL.

Література:

основна – [1 (С. 287-332), 3 (С. 681-751)];
додаткова – [3 (С. 52-82)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: опишіть організацію контейнера STL; які класи належать до категорії контейнерів; які вимоги до типів, які можна використовувати з контейнерами STL; яка різниця між контейнерами та адаптерами контейнерів; що таке ітератори, яких типів вони бувають та в чому їх

принципова різниця; які операції можна виконувати над усіма типами ітераторів; що таке ітератор довільного доступу та які операції він дозволяє виконувати; які вимоги до ітераторів з боку алгоритмів.

Комп'ютерний практикум 17. Робота з бібліотекою STL (алгоритми та тестування).

Питання, що розглядаються:

1) стандартні алгоритми бібліотеки шаблонів STL;
2) тестування різних видів контейнерів STL (vector, list, stack, queue, priority queue, deque, set, multiset, map, multimap).

Література:

основна – [1 (С. 287-332), 3 (С. 681-751)];
додаткова – [3 (С. 52-82)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: що таке стандартні алгоритми; які алгоритми вимагають впорядкованості.

Комп'ютерний практикум 18. Виняткові ситуації (проектування).

Питання, що розглядаються:

1) проектування класів для обробки виняткових ситуацій;
2) використання в проектування класів роботи з помилками std.

Література:

основна – [1 (С. 182-202)];
додаткова – [1 (С. 74-77), 2 (С. 171-206)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти спроектувати діаграму класів мовою UML для програмного додатку, в якому передбачено обробку виняткових ситуації.

Комп'ютерний практикум 19. Виняткові ситуації (реалізація та тестування).

Питання, що розглядаються:

1) обробка різні ситуації виникнення помилок;
2) помилки введення, переповнення пам'яті, обробки помилок часу виконання та інші;
3) оператори try, throw, catch;
4) засоби опрацювання виняткових ситуацій у мові програмування С++;
5) обробка різні ситуації виникнення помилок за допомогою генерації винятку.

Література:

основна – [1 (С. 182-202)];
додаткова – [1 (С. 74-77), 2 (С. 171-206)].

Завдання на СРС:

вивчити конспект лекції та ознайомитися з навчально-методичними матеріалами за темою заняття, щоб вміти відповісти на питання: що називається винятковою ситуацією; що називається обробкою виняткової ситуації; як здійснюється обробка виняткової ситуації в мові С++; чи можна помістити блок try у функцію, залишивши блок catch для обробки виняткових ситуацій у функції main(); назвіть і опишіть стандартні типи винятків з бібліотеки std; навести приклад коду, за допомогою якого можна вивести на екран текстове повідомлення з номером строчки, під час виконання якої сталася виняткова ситуація, та назвою файлу, що містить цю строчку; до чого призведе передавання винятку в блок catch за посиланням; для чого використовують вираз though, що не містить операндів; до чого призведе генерація похідної виняткової ситуації за умови наявності блоку catch, що призначений для обробки базової виняткової ситуації; для чого під час обробки виняткових ситуацій використовують призначені функції terminate(), unexpected(), abort().

Комп'ютерний практикум 20. Проведення МКР.

Питання, що виносяться на МКР, пов'язані із засвоєнням студентами всіх тем:

Тема 1. *Об'єктно-орієнтований підхід. Мова UML.*
Тема 2. *Абстрагування даних та інкапсуляція.*
Тема 3. *Особливості роботи з об'єктами.*
Тема 4. *Успадковування та поліморфізм.*
Тема 5. *Перевантаження операцій та функцій.*
Тема 6. *Шаблони функцій і класів. STL.*
Тема 7. *Виняткові ситуації.*
Тема 8. *Класи потоків уведення та виведення.*
Тема 9. *Статичні та динамічні бібліотеки.*

Література:

основна – [1-5];
додаткова – [1-20].

Завдання на СРС:

вивчити конспекти лекції; вивчити теоретичні відомості щодо об'єктно-орієнтованого аналізу та проектування, технології об'єктно-орієнтованого програмування.

Політика та контроль

6. Політика навчальної дисципліни (освітнього компонента)

Форми організації освітнього процесу, види навчальних занять і оцінювання результатів навчання регламентуються Положенням про організацію освітнього процесу в Національному технічному університеті України «Київському політехнічному інституті імені Ігоря Сікорського».

Правила відвідування занять. Відвідування є обов'язковим (за винятком випадків, коли існує поважна причина, наприклад, хвороба чи дозвіл працівників деканату). Якщо студент не може бути присутніми на заняттях, він все одно несете відповідальність за виконання завдань, що проводились в комп'ютерному класі.

Правила поведінки на заняттях. Під час виконання комп'ютерних практикумів (лабораторних робіт) студент може користуватися ноутбуком, мобільним телефоном або іншими пристроями для пошуку інформації на гугл-диску викладача чи в інтернеті. Проте під час лекційних занять та обговорення завдань комп'ютерних практикумів зазначеними раніше пристроями користуватися не можна. Це відволікає викладача і студентів групи та перешкоджає навчальному процесу. Якщо мається намір використовувати ноутбук або інший пристрій для аудіо- чи відеозапису, необхідно заздалегідь отримати дозвіл викладача. Під час лекційних занять заборонено відволікати викладача від викладання матеріалу, усі питання, уточнення та ін. студенти задають в кінці лекції або відведений для цього час.

Виконання завдань контрольних заходів

Плагіат та інші форми нечесної роботи неприпустимі. Всі комп'ютерні практикуми студенти мають виконувати самостійно із використанням рекомендованої літератури й отриманих знань та навичок. Цитування в письмових роботах допускається тільки із відповідним посиланням на авторський текст. Недопустимі підказки і списування у ході захисту комп'ютерних практикумів, на модульній контрольній роботі, на екзамені.

Комп'ютерні практикуми захищаються особисто з попередньою перевіркою теоретичних знань, які необхідні для виконання завдань за темою комп'ютерного практикуму. Перевірка практичних результатів включає перевірку коду та виконання програмного застосунку.

Модульна контрольна робота проводиться письмово без застосування допоміжних засобів (мобільні телефони, планшети, література та ін.) за принципом хронометражу часу виконання: доступ до контрольного завдання МКР відкривається викладачем у заздалегідь оголошений момент на визначений період часу. Результати модульної контрольної роботи оголошуються студентам на наступному занятті.

Екзамен проводиться письмово. На екзамені студенту не дозволяється користуватись будь-якими матеріалами.

Порядок зарахування пропущених занять. Відпрацювання пропущеного заняття з лекційного курсу здійснюється шляхом підготовки і захисту реферату за відповідною темою. Захист реферату відбувається відповідно до графіку консультацій викладача, з яким можна ознайомитись на кафедрі. Відпрацювання пропущеного комп'ютерного практикуму здійснюється шляхом самостійного виконання завдання і його захисту відповідно до графіку консультацій викладача.

Політика щодо академічної доброчесності. Політика та принципи академічної доброчесності визначені у розділі 3 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: https://kpi.ua/code.

Норми етичної поведінки. Норми етичної поведінки студентів і працівників визначені у розділі 2 Кодексу честі Національного технічного університету України «Київський політехнічний інститут імені Ігоря Сікорського». Детальніше: https://kpi.ua/code.

7. Види контролю та рейтингова система оцінювання результатів навчання (РСО)

Освітній компонент «Об'єктно-орієнтоване програмування»

Рейтинг студента розраховується виходячи із 100-бальної шкали,
з них 50 балів складає стартова шкала.
Стартовий рейтинг студента (протягом семестру) складається з балів, що студент отримує за:

- виконання та захист комп'ютерних практикумів із 7 тем:
  1. Реалізація класів, конструктори та деструктори (Тема 2. Абстрагування даних та інкапсуляція);
  2. Робота з об'єктами (Тема 3. Особливості роботи з об'єктами);
  3. Успадкування та поліморфізм (Тема 4. Успадковування та поліморфізм);
  4. Перевантаження операцій (Тема 5. Перевантаження операцій та функцій);
  5. Шаблони (Тема 6. Шаблони функцій і класів);
  6. Робота з бібліотекою STL(Тема 7. Стандартна бібліотека шаблонів);
  7. Виняткові ситуації (Тема 8. Виняткові ситуації);
- одну модульну контрольну роботу.

Максимальна сума вагових балів за всі контрольні заходи протягом семестру R_C складає:

R_C = 7 комп'ютерних практикумів · 6 балів + 1 МКР · 8 запитань · 1 бал = 50 балів.

Поточний контроль

Комп'ютерні практикуми

Ваговий бал – 6.

Максимальна кількість балів за всі комп'ютерні практикуми дорівнює 7 робіт · 6 балів = 42 бали.

Критерії оцінювання:

6 балів	–	вірно виконані проектування класів, реалізація алгоритмів, розрахункова частина та досконало проведений аналіз отриманих результатів;
4.9…5.9 балів	–	вірно виконані реалізація алгоритмів та розрахункова частина, але з 7-6 недостатньо проведеним аналізом отриманих результатів та обґрунтуванням, помилки в оформленні UML діаграми;
3.6..4.8 балів	–	виконано з грубими помилками;
0 балів	–	невірно виконано.

Модульна контрольна робота

Ваговий бал запитання – 1.

Максимальна кількість балів за контрольну роботу дорівнює 1 МКР · 8 запитань · 1 бал = 8 балів.

Критерії оцінювання:

0.6…1 бали	–	вірна відповідь на питання (відповідь містить не менше ніж 60 % необхідної інформації);
0 балів	–	відсутність відповіді або невірна відповідь на питання (менше 60% необхідної інформації).

Календарний контроль

Проміжна атестація студентів (далі – атестація) є календарним контролем. Метою проведення атестації є підвищення якості навчання студентів та моніторинг виконання графіка освітнього процесу студентами.

Умовою першої атестації є отримання не менше 12 балів та захист всіх комп'ютерних практикумів на час атестації. Умовою другої атестації є отримання не менше 24 балів та захист всіх комп'ютерних практикумів на час атестації.

Семестровий контроль

Умовами допуску до екзамену є:

- зарахування всіх комп'ютерних практикумів;
- написана МКР;
- стартовий рейтинг не менше 30 балів.

Екзаменаційну роботу всі студенти пишуть обов'язково.
Кількість запитань у кожному білеті – 3. Ваговий бал першого запитання – 10, другого – 20, третього – 20. Максимальна кількість балів за всі питання екзаменаційного білету дорівнює:

10+20+20 = 50 балів.

Критерії оцінювання першого запитань на екзамені:

9…10 балів – змістовна відповідь на теоретичне питання білетабілету;
7…8 балів – добра відповідь на питання, але з невеликими зауваженнями;
6 балів – задовільна відповідь на питання (є декілька грубих помилок у відповіді);
0 бали – більше двох грубих помилок/незнання питання

Критерії оцінювання другого та третього запитання на екзамені:

18…20 балів – правильна програмно-алгоритмічна реалізація розв'язку задачі білета;
15…17 балів – допущено несуттєві неточності в програмній реалізації;
12…14 балів – помилки в рахунках, не викладено основні етапи алгоритмічної частини;
0 балів – незнання розв'язку задачі, грубі помилки.

Таблиця відповідності рейтингових балів оцінкам за університетською шкалою

Сума стартових балів та балів за екзаменаційну роботу переводиться до оцінки згідно з таблицею:

Бали	**Оцінка**
100…95	Відмінно
94…85	Дуже добре
84…75	Добре
74…65	Задовільно
64…60	Достатньо
Менше 60	Незадовільно
Є не зараховані комп'ютерні практикуми або не написана МКР або стартовий рейтинг менше 30 балів	Не допущено

8. Додаткова інформація з дисципліни (освітнього компонента)

Дистанційне навчання. В умовах дистанційного режиму організація освітнього процесу здійснюється з використанням технологій дистанційного навчання: платформи дистанційного навчання «Сікорський» та «Електронний кампус». Навчальний процес у дистанційному режимі здійснюється відповідно до затвердженого розкладу навчальних занять. Заняття проходять з використанням сучасних ресурсів проведення онлайн-зустрічей (організація відео-конференцій).

Робочу програму навчальної дисципліни (силабус):

Складено доцентом кафедри БМК, к.т.н., Алхімовою Світланою Миколаївною

Ухвалено кафедрою біомедичної кібернетики (протокол №1 від 31 серпня 2023 року)

Погоджено Методичною комісією факультету (протокол №1 від 1 вересня 2023 року)