**МІНІСТЕРСТВО ОСВІТИ І НАУКИ,
МОЛОДІ ТА СПОРТУ УКРАЇНИ**

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Робоча програма
навчальної дисципліни
"ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ"

**для студентів напряму підготовки 6.050101
"Комп'ютерні науки"
всіх форм навчання**

Харків. Вид. ХНЕУ, 2012

Затверджено на засіданні кафедри інформаційних систем.
Протокол № 2 від 14.09.2011 р.

Укладачі: Парфьонов Ю. Е.
Щербаков О. В.
Федорченко В. М.

Р58 Робоча програма навчальної дисципліни "Об’єктно-орієнтоване програмування" для студентів напряму підготовки 6.050101 "Комп'ютерні науки" всіх форм навчання / укл. Парфьонов Ю. Е., Щербаков О. В., Федорченко В. М. – Х. : Вид. ХНЕУ, 2012. – 48 с. (Укр. мов.)

Наведено тематичний план навчальної дисципліни та її зміст за модулями й темами, вміщено плани лекцій і лабораторних занять, матеріал щодо закріплення знань (самостійна робота, контрольні запитання), методичні рекомендації та оцінювання знань студентів.

Рекомендовано для студентів напряму підготовки 6.050101 "Комп’ютерні науки".

Вступ

Сьогоднішні умови господарювання вимагають від фахівців з економічного управління всебічного використання новітніх інформаційних технологій. Широкі можливості комп'ютеризованих засобів у питаннях збору, обробки та видачі необхідної інформації здатні значно підвищити якість економічних розрахунків, зробити більш ефективним процес обґрунтування економічних рішень.

Але використання потужних комп'ютеризованих засобів неможливо без програмного забезпечення. Важливість галузі розроблення програмного забезпечення збільшується, оскільки тенденції розвитку комп'ютерної техніки свідчать про те, що з одного боку складність та функціональні можливості комп'ютерної техніки постійно і швидко зростають, а з іншого боку, це потрібує більш досконалих програмних засобів для задоволення потреб користувачів.

Істотною рисою таких програмних систем є рівень складності: для одного розробника практично неможливо охопити усі її аспекти. Причому ця складність є неминучою: з нею можливо справитися, але позбавитися від неї неможливо.

У теперішній час найбільш розповсюдженим методом боротьби зі складністю є об'єктно-орієнтований підхід до розробки програмного забезпечення. З використанням цього підходу розробляється більша частина програм у всьому світі. Це потрібує від відповідних фахівців чіткого уявлення концепцій об'єктно-орієнтованого програмування, що дає можливість їх практичного використання при розробці додатків на будь-якої мові програмування.

Метою навчальної дисципліни "Об'єктно-орієнтоване програмування" є засвоєння необхідних знань з основ об'єктно-орієнтованого програмування, а також формування твердих практичних навичок щодо розроблення додатків з використанням об'єктно-орієнтованого підходу.

Предметом вивчення дисципліни є принципи об'єктно-орієнтованого програмування, а також методи їх використання при розробленні додатків.

Необхідним елементом успішного засвоєння навчального матеріалу дисципліни є самостійна робота студентів з технічною літературою, та сучасними програмними засобами розроблення програм.

Структура робочої програми навчальної дисципліни "Об'єктно-орієнтоване програмування" наведена в табл.1.

Таблиця 1

Структура навчальної дисципліни

Характеристика дисципліни: підготовка бакалаврів	Напрям підготовки, галузь знань освітньо-кваліфікаційний рівень	Характеристика навчальної дисципліни
Кількість кредитів, відповідних до ECTS – 9; у тому числі: змістовних модулів – 4; самостійна робота; ІНДЗ; курсове проектування	Галузь знань: 0501 "Інформатика та обчислювальна техніка"	Обов'язкова. Рік підготовки: 2. Семестр: 3, 4
Кількість годин: усього – 324; за змістовними модулями: модуль 1 – 36 год.; модуль 2 – 96 год.; модуль 3 – 58 год.; модуль 4 – 134 год.;	Напрям підготовки "Комп'ютерні науки"	Лекції (теоретична підготовка) – 68 год. (51 + 17). Лабораторні заняття – 68 год. (34 + 34). Самостійна робота – 188 год. у т. ч. : поточні консультації – 16 год. Іспит + передекзаменаційна консультація – 4 год. Виконання курсового проекту – 36 год.
Кількість тижнів викладання дисципліни: 34 год. Кількість год. за тиждень: 2; 4; 6	Освітньо-кваліфікаційний рівень: бакалавр	Вид контролю: залік, іспит

У процесі навчання студенти отримують необхідні знання під час проведення аудиторних занять: лекційних та лабораторних. Велике значення в процесі вивчення та закріплення знань має самостійна робота студентів, у тому числі робота над курсовим проектом.

Зміст усіх видів занять розроблено відповідно до положень Болонської декларації, враховано рекомендації щодо кредитно-модульної системи організації навчального процесу.

1. Кваліфікаційні вимоги до студентів

Дисципліна "Об‘єктно-орієнтоване програмування" є базовою для підготовки бакалаврів за напрямом підготовки "Комп’ютерні науки".

Необхідна навчальна база перед початком вивчення дисципліни: з метою найкращого засвоєння матеріалу студенти повинні до початку вивчення дисципліни засвоїти теоретичні знання та опанувати практичні вміння з дисципліни "Алгоритмізація та програмування", а також мати навички роботи з персональним комп’ютером.

У результаті вивчення даної дисципліни студенти повинні

знати:

- поняття об'єктно-орієнтованого аналізу, проектування та програмування;
- принципи об‘єктно-орієнтованого програмування;
- поняття класу та об‘єкта, співвідношення між ними;
- типи відношень між класами;
- порядок проектування класів;
- життєвий цикл об‘єктів;
- реалізацію основних концепцій об‘єктно-орієнтованого програмування у мові C#;
- способи розроблення графічного інтерфейсу користувача за допомогою технології Windows Forms.

Отримати такі компетенції:

- здатність до об'єктно-орієнтованого мислення, знання об'єктно-орієнтованих мов програмування та здатність застосовувати такий підхід під час проектування складних програмних систем (КСП. 10);
- ґрунтовна підготовка в області програмування, володіння алгоритмічним мисленням, методами програмної інженерії для реалізації програмного забезпечення з урахуванням вимог до його якості, надійності, виробничих характеристик (КЗП.02);
- професійне володіння комп'ютером та інформаційними технологіями (КІ.04).

Робоча програма навчальної дисципліни розроблена відповідно до вимог галузевого стандарту вищої освіти МОН України на базі освітньо-професійної програми підготовки бакалавра за напрямом підготовки "Комп'ютерні науки".

2. Тематичний план навчальної дисципліни

При вивченні дисципліни "Об‘єктно-орієнтоване програмування" студент має ознайомитися з програмою дисципліни, її структурою, формами та методами навчання, видами та методами контролю знань.

Тематичний план дисципліни "Об‘єктно-орієнтоване програмування" складається з чотирьох модулів, кожний з яких об'єднує у собі відносно окремий самостійний блок дисципліни, який логічно пов'язує кілька навчальних елементів дисципліни за змістом та взаємозв'язками (табл. 2).

Таблиця 2

Структура тематичного плану навчальної дисципліни

Тема	Кількість годин		
	лекцій	лабораторних занять	самостійної роботи
1	2	3	4
Модуль 1. Об'єктно-орієнтований аналіз та проектування			
Тема 1. Основні положення об‘єктно-орієнтованого підходу	6	4	10
Тема 2. Основи об'єктно-орієнтованого проектування мовою UML	2		6
Разом годин за модулем	8	4	16
Модуль 2. Технологія ООП			
Тема 3. Введення до платформи Microsoft .NET та мови C#.	4	6	10
Тема 4. Реалізація головних концепцій об‘єктно-орієнтованого програмування у мові C#.	12	16	66
Разом годин за модулем	16	22	76

Закінчення табл. 2

1	2	3	4
Модуль 3. Обробка виключень і бібліотеки класів			
Тема 5. Обробка виняткових ситуацій	2	-	6
Тема 6. Стандартні бібліотеки класів Microsoft .NET Framework	12	14	16
Разом годин за модулем	14	14	22
Модуль 4. Об'єктно-орієнтоване програмування Windows-додатків на платформі Microsoft .NET			
Тема 7. Розроблення DLL-бібліотек	2	6	6
Тема 8. Основи програмування, керованого подіями	2	6	10
Тема 9. Основи використання технології Windows Forms	2	4	10
Тема 10. Розроблення графічних інтерфейсів користувача за допомогою технології Windows Forms	12	4	14
Тема 11. Використання графічних можливостей платформи Microsoft .NET	4	4	8
Тема 12. Розгортання програмного продукту	4	4	6
Тема 13. Огляд інших технологій розроблення Windows-додатків на платформі Microsoft .NET	4	-	20
Разом годин за модулем	30	28	74
Всього годин	68	68	188

3. Зміст навчальної дисципліни за модулями та темами

Модуль 1. Об'єктно-орієнтований аналіз та проектування

Тема 1. Основні положення об‘єктно-орієнтованого підходу

Прості та складні програмні системи. Декомпозиція програмних систем. Способи декомпозиції. Об’єктно-орієнтована декомпозиція.

Поняття об’єкта. Характеристики об’єкта. Поняття класу. Співвідношення між класом та його об‘єктом. Об’єктно-орієнтований аналіз та його мета. Поняття предметної області. Головні види вимог до програм-

ної системи. Об’єктно-орієнтоване проектування. Елементи архітектури додатку. Визначення класів предметної області. Принципи проектування класів. Об’єктно-орієнтоване програмування. Принципи об’єктно-орієнтованого підходу: абстракція, інкапсуляція, ієрархія, поліморфізм.

Тема 2. Основи об'єктно-орієнтованого проектування мовою UML

Моделювання програмного забезпечення. Методи об’єктно-орієнтованого аналізу і проектування. Мова UML. UML-діаграми. Діаграми класів. Генерування програмного коду.

Модуль 2. Технологія ООП

Тема 3. Введення до платформи Microsoft .NET та мови C#

Платформа Microsoft .NET: історія розвитку, архітектура, засоби розроблення додатків, компіляція та виконання програм, бібліотека базових класів, система типізації.

Загальні відомості про мову C#: особливості використання, алфавіт, типи даних, порівняння типів-значень та типів-посилань, вбудовані типи-значення, вбудовані типі-посилання, одновимірні та багатовимірні масиви, операції, оператори, структура програми, коментарі, особливості використання функцій, механізми передачі параметрів, простори імен, основи використання бібліотеки базових класів .NET

Культура програмування: вимоги до інтерфейсу користувача та вихідного тексту програми.

Тема 4. Реалізація головних концепцій об‘єктно-орієнтованого програмування у мові C#

Абстрактні типі даних. Проектування Абстрактного типу даних. Синтаксис структур та класів у мові C#. Елементи класу. Доступ до елементів класу. Посилання this. Перевантаження методів класу.

Об‘єкти в програмі. Послідовність створення об‘єкта. Конструктори. Основні властивості конструкторів. Звільнення пам‘яті. Система "збору сміття". Статичні дані та методи: призначення, властивості, особливості використання.

Відношення агрегації. Реалізація агрегації у мові C#. Відношення спадкування. Синтаксис спадкування у мові C#. Ініціалізація об‘єкта базового класу. Варіанти використання спадкування. Перевизначення методів. Заборона спадкування. Рядкове представлення об‘єкта.

Реалізація принципу поліморфізму у мові C#. Раннє та пізнє зв‘язування. Віртуальні методи. Абстрактні класи та методи. Реалізація поліморфної поведінки на базі абстрактного класу. Правила застосування абстрактних класів. Інтерфейси. Реалізація поліморфної поведінки на базі інтерфейсу. Правила застосування інтерфейсів.

Принципи перевантаження операцій. Особливості використання функції operator. Індексатори. Властивості.

Модуль 3. Обробка виключень і бібліотеки класів

Тема 5. Обробка виняткових ситуацій

Види помилок у програмах. Проблеми традиційного підходу до обробки помилок. Механізм обробки виключень. Класи виключень стандартної бібліотеки .NET. Синтаксис обробки виключень. Перевірка на арифметичне переповнення.

Тема 6. Стандартні бібліотеки класів Microsoft .NET Framework

Джерела та споживачі даних. Загальні відомості про потоки введення-виведення даних. Алгоритми роботи потоків введення-виведення даних. Основні класі стандартної бібліотеки .NET для підтримки введення-виведення даних.

Загальні відомості про контейнери. Основні елементи та структури даних стандартної бібліотеки контейнерів .NET Типізовані контейнери.

Особливості реалізації рядкового типу даних у платформі .Net Класи стандартної бібліотеки .NET для подання рядків та особливості їхнього використання. Форматування рядків. Призначення та застосування регулярних виразів. Підтримка регулярних виразів у стандартній бібліотеці .NET Спеціальні символи, які використовуються у регулярних виразах.

Введення до атрибутів. Елементи програми до яких можливо застосування атрибутів. Визначені атрибути. Використання атрибутів умовної компіляції. Атрибути рівня модулю компіляції.

Збереження та відновлення стану об‘єктів у .NET Серіалізація та десеріалізація. "Граф" об‘єктів при серіалізації. Створення класів, об‘єкти яких можливо серіалізувати. Процеси серіалізації та десеріалізації. Формати серіалізації. Серіалізація та десеріалізація об‘єктів у двійковому та XML-форматах.

Використання вбудованих запитів LINQ.

Модуль 4. Об'єктно-орієнтоване програмування Windows-додатків на платформі Microsoft .NET

Тема 7. Розроблення DLL-бібліотек

Бібліотеки та їх використання. Статичні та динамічні бібліотеки. DLL-бібліотеки. Розроблення DLL-бібліотеки на платформі Microsoft .NET.

Тема 8. Основи програмування, керованого подіями

Загальні відомості про делегати. Оголошення та використання делегатів у мові C#. Анонімні методі. Групові делегати. Загальні відомості про події. Генерування подій.

Тема 9. Основи використання технології Windows Forms

"Традиційна" модель програмування на платформі .NET Модель "Windows-програмування" на платформі .NET Технологія Windows Forms. Форми. Загальна структура додатку з графічним інтерфейсом користувача на платформі .NET Розроблення додатків Windows Forms за допомогою інтегрованого середовища. Події рівня форми.

"Колекція" візуальних елементів управління форми. Використання базових візуальних елементів управління.

Тема 10. Розроблення графічних інтерфейсів користувача за допомогою технології Windows Forms

Основи архітектури додатків Windows Forms. Модель подій у Windows Forms. Діалогові вікна. Основні візуальні елементи управління:

властивості та використання. Компоненти форми для виключення помилкового введення даних користувачем.

Використання візуальних елементів управління "дерево" та "таблиця". Робота з даними у XML-форматі.

Тема 11. Використання графічних можливостей платформи Microsoft .NET

Особливості графічного виведення даних. Логічна система координат. Простори імен GDI+. Обробка повідомлення перемальовування. Програмне генерування повідомлення перемальовування. Графічні об‘єкти GDI+. Використання пензлів, пер та шрифтів. Робота з графічними зображеннями.

Тема 12. Розгортання програмного продукту

Структура модулю компіляції. Приватні та спільні модулі компіляції. Глобальний кеш модулів компіляції. Створення спільних модулів компіляції.

Загальні відомості про розгортання додатків. Види розгортання. Проекти установки та розгортання.

Тема 13. Огляд інших технологій розроблення Windows-додатків на платформі Microsoft .NET

Огляд технології Windows Presentation Foundation. Огляд технології Microsoft Silverlight.

4. Плани лекцій

Модуль 1. Об'єктно-орієнтований аналіз та проектування

Тема 1. Основні положення об‘єктно-орієнтованого підходу

1.1. Основи об‘єктно-орієнтованого підходу.
1.2. Елементи об‘єктно-орієнтованої технології.
1.3. Об'єктна модель предметного середовища.

Література: основна [1; 2].

Тема 2. Основи об'єктно-орієнтованого проектування мовою UML

2.1. Основи використання мови UML.

Література: додаткова [7].

Модуль 2. Технологія ООП

Тема 3. Введення до платформи Microsoft .NET та мови C#

3.1. Основні поняття платформи Microsoft .NET.

3.2. Основні елементи мови C#.

Література: основна [2; 3]; додаткова [4; 6].

Тема 4. Реалізація головних концепцій об‘єктно-орієнтованого програмування у мові C#

4.1. Основи об’єктно-орієнтованого програмування мовою C#.

4.2. Створення та руйнування об‘єктів.

4.3. Повторне використання класів.

4.4. Реалізація поліморфізму в C#.

4.5. Абстрактні класи та інтерфейси.

4.6. Принципи перевантаження операцій.

Література: основна [2; 3]; додаткова [4; 6].

Модуль 3. Обробка виключень і бібліотеки класів

Тема 5. Обробка виняткових ситуацій

5.1. Обробка виключень.

Література: основна [2; 3]; додаткова [4; 6].

Тема 6. Стандартні бібліотеки класів Microsoft .NET Framework

6.1. Введення-виведення даних.

6.2. Контейнери.

6.3. Рядки.

6.4. Регулярні вирази.

6.5. Збереження та відновлення стану об‘єктів у .NET.

6.6. Вбудовані запроси LINQ.

Література: основна [2; 3]; додаткова [4 – 6].

Модуль 4. Об'єктно-орієнтоване програмування Windows-додатків на платформі Microsoft .NET

Тема 7. Розроблення DLL-бібліотек

7.1. DLL-бібліотеки та їх використання.

Література: основна [3]; додаткова [4].

Тема 8. Основи програмування, керованого подіями

8.1. Делегати та події.

Література: основна [2; 3]; додаткова [4].

Тема 9. Основи використання технології Windows Forms.

9.1. Введення до Windows Forms.

Література: основна [2; 3]; додаткова [4; 6].

Тема 10. Розроблення графічних інтерфейсів користувача за допомогою технології Windows Forms

10.1. Основи використання елементів управління.

10.2. Особливості використання форм.

10.3. Використання основних елементів управління.

10.4. Використання "дерев" у графічному інтерфейсі користувача.

10.5. Використання таблиць у графічному інтерфейсі користувача.

10.6. Робота з даними у XML-форматі.

Література: основна [2; 3]; додаткова [6].

Тема 11. Використання графічних можливостей платформи Microsoft .NET

11.1. Основи графічної підсистеми платформи Microsoft .NET.

11.2. Використання головних елементів графічної підсистеми платформи Microsoft .NET.

Література: основна [2; 3]; додаткова [6].

Тема 12. Розгортання програмного продукту

12.1. Модулі компіляції на платформі Microsoft .NET.

12.1. Розгортання додатків.

Література: основна [2; 3].

Тема 13. Огляд інших технологій розроблення Windows-додатків на платформі Microsoft .NET

13.1. Основи технології Windows Presentation Foundation.

13.2. Основи технології Silverlight.

Література: додаткова [8].

5. Плани лабораторних занять

Лабораторне заняття – це організаційна форма навчального заняття, на якому студенти під керівництвом викладача формують уміння й навички з практичного застосування основних теоретичних положень навчальної дисципліни шляхом виконання завдань до лабораторних робіт.

Лабораторні заняття з дисципліни "Об‘єктно-орієнтоване програмування" проводяться в спеціально обладнаному навчальному класі з використанням комп’ютерного устаткування пристосованого до умов навчального процесу.

З метою підвищення якості навчального процесу, під час проведення лабораторного заняття призначається ще один викладач і навчальна група ділиться на дві підгрупи. Кожний студент працює самостійно, виконуючи індивідуальне завдання для лабораторного дослідження.

Лабораторне заняття включає проведення поточного контролю підготовленості студентів до виконання конкретної лабораторної роботи, виконання завдань теми заняття, оформлення індивідуального звіту з виконаної роботи та його захист перед викладачем.

Виконання лабораторної роботи оцінюється викладачем. Підсумкові оцінки, отримані студентом за виконання лабораторних робіт, враховуються при виставленні семестрової підсумкової оцінки з дисципліни.

Тематика проведення лабораторних занять наведена у табл. 3.

Таблиця 3

Перелік тем лабораторних занять

Назва теми	Назва лабораторного заняття та питання, що опрацьовуються	Кількість годин	Література
1	2	3	4
Модуль 1. Об'єктно-орієнтований аналіз та проектування			
Тема 1. Основні положення об‘єктно-орієнтованого підходу	1. Проектування класів з використанням мови UML	4	основна [1; 2]; додаткова [7]
Тема 2. Основи об'єктно-орієнтованого проектування мовою UML			
Модуль 2. Технологія ООП			
Тема 3. Введення до платформи Microsoft .NET та мови C#	2. Основи використання мови C#	6	основна [2; 3]; додаткова [4; 6]
Тема 4. Реалізація головних концепцій об‘єктно-орієнтованого програмування у мові C#.	3. Розроблення додатків з використанням базових елементів ООП	8	основна [2; 3]; додаткова [4; 6]
	4. Застосування спадкування та поліморфізму	4	основна [2; 3]; додаткова [4; 6]
	5. Перевантаження операцій	4	основна [2; 3]; додаткова [4; 6]

Продовження табл. 3

1	2	3	4
Модуль 3. Оброблення виключень і бібліотеки класів			
Тема 5. Обробка виняткових ситуацій	–	–	–
Тема 6. Стандартні бібліотеки класів Microsoft .NET Framework	6. Використання основних бібліотек Microsoft .NET Framework	8	основна [2; 3]; додаткова [4 – 6]
	7. Використання регулярних виразів	6	основна [2; 3]; додаткова [4 – 6]
Модуль 4. Об'єктно-орієнтоване програмування Windows-додатків на платформі Microsoft .NET			
Тема 7. Розроблення DLL-бібліотек	8. Проектування бібліотеки класів	6	основна [3]; додаткова [4]
Тема 8. Основи програмування, керованого подіями	9. Використання делегатів та подій	6	основна [2; 3]; додаткова [4]
Тема 9. Основи використання технології Windows Forms	10. Використання основних концепцій Windows Forms	4	основна [2; 3]; додаткова [4 – 6]
Тема 10. Розроблення графічних інтерфейсів користувача за допомогою технології Windows Forms	11. Використання елементів управління у додатках з графічним інтерфейсом користувача	4	основна [2; 3]; додаткова [6]

Закінчення табл. 3

1	2	3	4
Тема 11. Використання графічних можливостей платформи Microsoft .NET	12. Використання графіки у додатках Win-Forms.	4	основна [2; 3]; додаткова [6]
Тема 12. Розгортання програмного продукту	13. Розгортання додатків на платформі Microsoft .NET	4	основна [2; 3]
Тема 13 Огляд інших технологій розроблення Windows-додатків на платформі Microsoft .NET	–	–	–
Разом годин		68	

6. Самостійна робота студентів

6.1. Основні форми самостійної роботи студентів

Для опанування матеріалу дисципліни "Об‘єктно-орієнтоване програмування" окрім лекційних та лабораторних занять, тобто аудиторної роботи, значну увагу необхідно приділяти самостійній роботі.

Основні форми самостійної роботи студента:

1. Вивчення лекційного матеріалу.
2. Вивчення окремих тем або питань, що передбачені для самостійного опрацювання.
3. Вивчення основних термінів та понять за темами дисципліни.
4. Підготовка до лабораторних занять.
5. Контрольна перевірка кожним студентом знань за питаннями для самодіагностики.
6. Підготовка до проміжного та підсумкового модульного контролю.
7. Систематизація вивченого матеріалу перед іспитом.
8. Виконання індивідуального навчально-дослідного завдання.
9. Оформлення звітів з лабораторних робіт.
10. Робота з опрацювання та вивчення рекомендованої літератури.
11. Робота над курсовим проектом.

6.2. Питання для самостійного опрацювання

Модуль 1. Об'єктно-орієнтований аналіз та проектування

Тема 1. Основні положення об‘єктно-орієнтованого підходу

1. Ознаки складних програмних систем.
2. Переваги об‘єктно-орієнтованого підходу до розроблення програмних систем.
3. Відношення між об’єктами.
4. Відношення між класами.

Література: основна [1; 2].

Тема 2. Основи об'єктно-орієнтованого проектування мовою UML

1. Історія розвитку мови UML.
2. Види UML-діаграм.
3. CASE-засоби.

Література: додаткова [7].

Модуль 2. Технологія ООП

Тема 3. Введення до платформи Microsoft .NET та мови C#

1. Історія та етапи розвитку платформи Microsoft.NET
2. Переваги платформи Microsoft .NET над аналогами.
3. Призначення та особливості віртуальної машини.
4. Призначення основних просторів імен бібліотеки класів платформи Microsoft .NET.
5. Використання мови XML у програмних проектах .NET.
6. Варіанти об‘явлення та використання головного методу програми на мові C#.
7. Способи використання директиви using.
8. Форматування даних при їх виведенні.
9. Використання утілити ILDasm.
10. Налагодження програм у середовищі Microsoft Visual Studio.
11. Використання засобів SDK щодо розроблення програм.

Література: основна [2; 3]; додаткова [4; 6].

Тема 4. Реалізація головних концепцій об‘єктно-орієнтованого програмування у мові C#

1. Призначення та використання перечислень.
2. Призначення та використання структур.
3. Використання списків аргументів змінної довжині.
4. Розробка рекурсивних методів.
5. Призначення та елементи класу Array.

6. Використання аргументів командного рядка.
7. Використання внутрішніх класів.
8. Пізнє та раннє зв‘язування.

Література: основна [2; 3]; додаткова [4; 6].

Модуль 3. Обробка виключень і бібліотеки класів

Тема 5. Обробка виняткових ситуацій

1. Призначення та використання елементів класу System.Exception.
2. Використання вкладених блоків try.
3. Розробка власних класів виключень.
4. Генерування користувальницьких виключень.
5. Перехоплення користувальницьких виключень.

Література: основна [2; 3]; додаткова [4; 6].

Тема 6. Стандартні бібліотеки класів Microsoft .NET Framework

1. Перетворення типів, які визначаються користувачем.
2. Особливості буферизованих потоків введення-виведення даних.
3. Хеш-таблиці та хеш-функції.
4. Сортування вмісту контейнеру.
5. Розроблення користувальницьких контейнерів.
6. Засоби форматування рядків.
7. Алгоритм роботи "збирача сміття".
8. Поняття про "покоління" об‘єктів.
9. Розробка користувальницьких атрибутів.
10. Обмеження на застосування атрибутів.
11. Критерії вибору формату серіалізації.
12. Налагодження параметрів серіалізації за допомогою атрибутів.

Література: основна [2; 3]; додаткова [4 – 6].

Модуль 4. Об'єктно-орієнтоване програмування Windows-додатків на платформі Microsoft .NET

Тема 7. Розроблення DLL-бібліотек

1. Проблема "Аду DLL" та напрями її рішення.

Література: основна [3]; додаткова [4].

Тема 8. Основи програмування, керованого подіями

1. Інтерфейси зворотного виклику.
2. Коваріантність делегатів.
3. Реалізація групових викликів.
4. Реєстрація подій у середовищі Visual Studio.

Література: основна [2; 3]; додаткова [4].

Тема 9. Основи використання технології Windows Forms

1. Призначення та застосування класу Application.
2. Функціональні можливості класу Control.
3. Використання "колекції" візуальних елементів управління форми.
4. Застосування оброблювачів подій рівня форми.
5. Розміщення візуальних елементів управління на формі.
6. Ієрархія класів-предків класу Form та його функціональні можливості.
7. Основні властивості та події класу Control.
8. Склад та призначення основних файлів проекту типу Windows Application у середовищі Visual Studio.

Література: основна [2; 3]; додаткова [4; 6].

Тема 10. Розроблення графічних інтерфейсів користувача за допомогою технології Windows Forms

1. Використання класу Timer.
2. Використання візуального елементу управління CheckedListBox.
3. Основні елементи класу TreeView та особливості його використання.

4. Основні елементи класу DataGridView та особливості його використання.

5. Основні елементи класу DataTable та особливості його використання.

6. Спадкування форм.

7. Основні властивості перерахування DialogResult.

Література: основна [2; 3]; додаткова [6].

Тема 11. Використання графічних можливостей платформи Microsoft .NET

1. Особливості застосування системи координат GDI+.
2. Допоміжні типи даних простору імен System.Drawing.
3. Перевірка влучення у задану область графічного зображення.
4. Використання штрихових пензлів.
5. Використання градієнтних пензлів.

Література: основна [2; 3]; додаткова [6].

Тема 12. Розгортання програмного продукту

1. Структура маніфесту модуля компіляції.
2. Використання "суворих" імен модуля компіляції.
3. Варіанти розгортання додатків .NET.
4. Створення інсталяторів за допомогою середовища Visual Studio.

Література: основна [2; 3].

Тема 13. Огляд інших технологій розроблення Windows-додатків на платформі Microsoft .NET

1. Спільне використання технологій Windows Forms та Windows Presentation Foundation.

Література: додаткова [8].

6.3. Контрольні запитання для самодіагностики

Модуль 1. Об'єктно-орієнтований аналіз та проектування

Тема 1. Основні положення об‘єктно-орієнтованого підходу

1. Прості і складні програмні системи.
2. Як можна боротися зі складністю програм?
3. Сутність об'єктно-орієнтованої декомпозиції.
4. Характеристики об'єкта.
5. Об'єктна модель предметної області.

Література: основна [1; 2].

Тема 2. Основи об'єктно-орієнтованого проектування мовою UML

1. Основні поняття візуального моделювання.
2. Графічні нотації моделювання, які широко використовуються.
3. Основні розробники мови UML.
4. Види відносин на діаграмі класів.

Література: додаткова [7].

Модуль 2. Технологія ООП

Тема 3. Введення до платформи Microsoft . NET та мови C#

1. Охарактеризуйте структуру програми мовою C#.
2. Перелічіть вбудовані типи даних C#.
3. Призначення методів класу Math.
4. Порядок створення проекта консольного додатку C# у середовищі Visual Studio.
5. Порядок відкриття проекта у середовищі Visual Studio.
6. Порядок добавлення файлу до проекту у середовищі Visual Studio.

7. Порядок компіляції та запуску програми на виконання у середовищі Visual Studio.
8. Порядок налагодження програми у середовищі Visual Studio.
9. Порядок використання утиліти Object Browser**.**

Література: основна [2; 3]; додаткова [4; 6].

Тема 4. Реалізація головних концепцій об‘єктно-орієнтованого програмування у мові C#

1. Пояснити визначення методу: *public static void Main(string[ ] args) { }.*
2. Принципи об’єктно-орієнтованого підходу.
3. Синтаксис опису класу.
4. Особливості статичних елементів класу.
5. Специфікатори доступу до елементів класу у C#.
6. Порядок ініціалізації об'єкта класу.
7. Призначення та використання посилання this.
8. Агрегація й наслідування.
9. Синтаксис наслідування у C#.
10. Порядок виклику конструкторів при наслідуванні.
11. Перевантаження "базового" методу.
12. Перевизначення "базового" методу.
13. Принцип поліморфізму.
14. Переваги концепції поліморфізму.
15. Поняття про абстрактні класи та їх призначення.
16. Правила використання абстрактних класів.
17. Поняття про інтерфейси та їх призначення.
18. Правила використання інтерфейсів.

Література: основна [2; 3]; додаткова [4; 6].

Модуль 3. Оброблення виключень і бібліотеки класів

Тема 5. Обробка виняткових ситуацій

1. Проблеми "традиційного підходу" до обробки помилок під час виконання програми.

2. Переваги обробки виключень порівняно з "традиційним підходом" до обробки помилок.

Література: основна [2; 3]; додаткова [4; 6].

Тема 6. Стандартні бібліотеки класів Microsoft .NET Framework

1. Особливості байтових і символьних потоків введення-виведення.

2. Базові класи байтових потоків введення-виведення .NET і їх основні методи.

3. Базові класи символьних потоків .NET і їх основні методи.

4. Призначення основних класів символьних потоків введення-виведення .NET.

5. Призначення основних класів байтових потоків введення-виведення .NET.

6. У чому різниця між буферізованими й небуферізованими потоками потоків введення-виведення .NET?

7. Состав бібліотеки контейнерів .NET.

8. Основні інтерфейси бібліотеки контейнерів .NET.

9. Коротка характеристика структури даних "динамічний масив" і її реалізація в .NET.

10. Коротка характеристика структури даних "двузв’язний список" і її реалізація в .NET.

11. Поняття про хеш-таблицю й хеш-функцію.

12. Коротка характеристика структури даних "асоціативний масив" і її реалізації в .NET.

13. Ітератори і їхнє використання.

14. Засоби розбору рядків на лексеми бібліотеки .NET.

15. Особливості реалізації рядкового типу даних у платформі .NET.

16. Класи стандартної бібліотеки .NET для представлення рядків та особливості їхнього використання.

17. Елементи програми до яких можливе застосування атрибутів.

18. Використання атрибутів умовної компіляції.

19. Створення класів, об‘єкти яких можливо серіалізувати.

20. Формати серіалізації у .NET.

Література: основна [2; 3]; додаткова [4 – 6].

Модуль 4. Об'єктно-орієнтоване програмування Windows-додатків на платформі Microsoft .NET

Тема 7. Розроблення DLL-бібліотек

1. Створення DLL-бібліотек на платформі Microsoft. NET.

Література: основна [3]; додаткова [4].

Тема 8. Основи програмування, керованого подіями

1. Оголошення та використання делегатів у мові C#.
2. Поняття групового делегату.
3. Поняття події.

Література: основна [2, 3]; додаткова [4].

Тема 9. Основи використання технології Windows Forms

1. Загальна структура додатку з графічним інтерфейсом користувача на платформі .NET.
2. Порядок додавання елемента управління на форму.
3. Ієрархія класів простору імен System.Windows.Forms.
4. "Колекція" візуальних елементів управління форми.
5. Використання компоненту MenuStrip.
6. Використання компоненту ToolStrip.

Література: основна [2; 3]; додаткова [4; 6].

Тема 10. Розроблення графічних інтерфейсів користувача за допомогою технології Windows Forms

1. Обробка подій пунктів меню.
2. Обробка подій кнопок панелі інструментів.
3. Акселератори та підказки, яки спливають.
4. Властивості та використання компоненту StatusStrip.
5. Властивості та використання компоненту Button.
6. Властивості та використання компоненту TextBox.
7. Властивості та використання компоненту Label.
8. Властивості та використання компоненту GroupBox.
9. Властивості та використання компоненту RadioButton.
10. Властивості та використання компоненту CheckBox.

11. Властивості та використання компоненту TabControl.
12. Властивості та використання компоненту TreeView.
13. Файлове введення-виведення. Стандартні діалоги SaveFile-Dialog, OpenFileDialog.
14. Використання компоненту ErrorProvider.
15. Основні властивості елементу управління "дерево".
16. Основні властивості елементу управління "таблиця".

Література: основна [2; 3]; додаткова [6].

Тема 11. Використання графічних можливостей платформи Microsoft .NET

1. Поняття контексту пристрою у .NET.
2. Поняття про логічну систему координат.
3. Простори імен GDI+ та їх призначення.
4. Обробка повідомлення перемальовування.
5. Програмне генерування повідомлення перемальовування.
6. Використання пензлів.
7. Використання пер.
8. Використання шрифтів.
9. Робота з графічними зображеннями.

Література: основна [2; 3]; додаткова [6].

Тема 12. Розгортання програмного продукту

1. Структура модулю компіляції.
2. Поняття про приватні та поділювані модулі компіляції.
3. Призначення глобального кешу модулів компіляції.
4. Види проектів інсталяторів.
5. Поняття про культури та регіони.
6. Призначення та використання файлів ресурсів.
7. Поняття про систему безпеки платформи .NET.
8. Поняття про групи коду.
9. Рівні політики безпеки.
10. Призначення та використання конфігураційних файлів.

Література: основна [2; 3].

Тема 13. Огляд інших технологій розроблення Windows-додатків на платформі Microsoft .NET

1. Функціональність спадкоємців класів System.Windows.Controls.Panel, System.Windows.Controls.Control, System.Windows.Shapes.-Shape.
2. Прив'язка даних у WPF.

Література: додаткова [8].

6.4. Індивідуальне навчально-дослідне завдання

Індивідуальне навчально-дослідне завдання (ІНДЗ) передбачає: систематизацію, закріплення, розширення теоретичних знань і практичних навичок з дисципліни та застосування їх при вирішенні конкретних виробничих ситуацій; розвиток навичок самостійної роботи з науково-технічною документацією.

ІНДЗ з дисципліни "Об‘єктно-орієнтоване програмування" видається студенту викладачем на початку вивчення дисципліни. ІНДЗ виконується студентом самостійно. Студент має надати ІНДЗ для перевірки наприкінці семестру, але не пізніше терміну проведення підсумкового модульного контролю. Оцінка за виконання ІНДЗ враховується при виставленні загальної оцінки з дисципліни.

Тематика ІНДЗ має носити проблемний характер. Студент має право самостійно обрати тему та зміст роботи з обов'язковим її узгодженням з викладачем.

Інакше тема має бути запропонована викладачем (варіанти тем наведені далі).

У процесі виконання ІНДЗ студент має опрацювати не менше п'яти літературних джерел з посиланнями на використання певної інформації з них у тексті роботи. При цьому робота має носити творчій характер і бути спрямованою на вирішення певної проблеми чи на висловлення особистого погляду автора роботи на питання, яке розглядається в роботі.

ІНДЗ складається з: титульної сторінки; змісту; вступу; основної частини; висновків; списку використаної літератури, додатків до індивідуального завдання (при необхідності).

Вступ має розкривати актуальність обраної студентом теми, її проблематику, мету написання роботи.

Основна частина роботи (може складатися з декількох підрозділів) повинна містити характеристику сучасного стану проблеми, погляд різних авторів на цю проблему, позитивні та негативні наслідки проблеми.

Висновки мають містити аналіз отриманих результатів.

Обсяг ІНДЗ повинен становити у друкованому варіанті 8 – 10 сторінок.

6.4.1. Тематика ІНДЗ

Загальна тема: "Дослідження особливостей розроблення об'єктно-орієнтованих програм на платформі Microsoft .NET".

Варіанти завдань:

1. Об‘єктно-орієнтоване програмування: історія виникнення, основоположники, необхідність.
2. Основні принципи об‘єктно-орієнтованого програмування.
3. Класи і об'єкти, співвідношення між ними.
4. Створення об'єктів у C#, конструктори.
5. Руйнування об'єктів у C#, збирання "сміття".
6. Перевантаження операцій у C#.
7. Спадкування в C#.
8. Агрегація в C#.
9. Поліморфізм у C#.
10. Статичні дані в C#.
11. Перевантаження методів у C#.
12. Статичні методи в C#.
13. Перевизначення методів у C#.
14. Використання посилання this у C#.
15. Абстрактні класи в C#.
16. Інтерфейси в C#.
17. Інкапсуляція в C#.
18. Використання конструкторів при спадкуванні.
19. Управління доступом до елементів класу.
20. Клас System.Object: призначення, основні елементи, використання.
21. Класи і структури в C#.

22. Використання коментарів у програмах на C#.

23. Використання UML-діаграм класів у системі програмування Microsoft Visual Studio.

24. Доступ до даних класу за допомогою методів Set ... (Get ...) і властивостей у C#.

25. Платформа Microsoft .NET: історія розвитку, архітектура, засоби розроблення програм, компіляція і виконання програм.

26. Платформа Microsoft .NET: система типізації, бібліотека базових класів .NET Framework 4.0.

27. Підтримка математичних операцій в Microsoft .NET: призначення, основні елементи, використання.

28. Підтримка "незмінних" рядків у Microsoft .NET: призначення, основні елементи, використання.

29. Підтримка "змінюваних" рядків у Microsoft .NET: призначення, основні елементи, використання.

30. Основи використання мови XML при розробленні додатків для Microsoft .NET.

31. Регулярні вирази в Microsoft .NET і основні елементи їхньої мови.

32. Механізми передачі параметрів методів у C#.

33. Система обробки виключень в Microsoft .NET.

34. Підтримка нетипізованих динамічних структур даних у Microsoft .NET: призначення, основні елементи, використання.

35. Підтримка типізованих динамічних структур даних у Microsoft .NET: призначення, основні елементи, використання.

36. Покажчики та їх використання в C#.

37. Атрибути в Microsoft .NET: призначення, основні елементи, використання.

38. Використання масивів у C#. Клас System.Array: призначення, основні елементи, використання.

39. Підтримка операцій з датою і часом в Microsoft .NET: призначення, основні елементи, використання.

40. Підтримка консольного введення-виведення даних у Microsoft .NET: призначення, основні елементи, використання.

41. Підтримка форматного виведення даних у Microsoft .NET: призначення, основні елементи, використання.

42. Підтримка файлового введення-виведення даних у Microsoft .NET: призначення, основні елементи, використання.

43. Псевдовипадкові числа і їх підтримка в Microsoft .NET

44. Підтримка архівації при введенні-виведенні даних у Microsoft .NET: призначення, основні елементи, використання.

45. Мова запитів LINQ в Microsoft .NET: призначення, основні елементи, використання.

46. Використання засобів командного рядка для компіляції і виконання програм для Microsoft .NET

47. Перетворення між типами даних в C#: призначення, основні способи, використання.

6.5. Курсове проектування

6.5.1. Завдання курсового проектування

Курсове проектування є завершальним етапом вивчення дисципліни "Об‘єктно-орієнтоване програмування".

Робота над курсовим проектом сприяє систематизації, поглибленню й закріпленню знань, отриманих студентами при вивченні даної дисципліни. У процесі курсового проектування студенти розвивають навички практичного застосування отриманих знань при створенні комплексного додатка з використанням сучасних інструментальних засобів розробки. При цьому студент повинен показати вміння користуватися спеціальною літературою, державними стандартами, довідниками та іншими матеріалами з інформаційних технологій.

У розробленому курсовому проекті студент повинен показати знання:

основних елементів предметної області відповідно до постановки завдання;

основних концепцій об‘єктно-орієнтованого програмування;

сучасних інструментальних засобів, призначених для розроблення об‘єктно-орієнтованих додатків.

Студент повинен показати вміння з:

аналізу постановки завдання;

декомпозиції програмної системи на підсистеми;

розроблення загальної архітектури програмної системи;

деталізації загальної архітектури програмної системи у термінах об‘єктно-орієнтованого програмування;

програмної реалізації системи, що розробляється, на мові програмування;

застосування стандартних бібліотек мови програмування;

документування вихідного коду програми;

використання засобів розроблення програм та отримання довідкової інформації;

розроблення та оформлення програмної документації.

Робота над курсовим проектом певною мірою визначає загальнотеоретичну та спеціальну підготовку студента і в остаточному підсумку готує його до майбутнього виконання більш складного й завершального етапу навчального процесу – дипломного проектування. Студент повинен розглядати роботу над курсовим проектом як своєрідну "репетицію" дипломного проектування.

6.5.2.Організація курсового проектування

Відповідно до навчального плану вивчення дисципліни "Об‘єктно-орієнтоване програмування" у 4 семестрі передбачається виконання студентами курсового проекту.

Керівництво курсовим проектуванням здійснюється викладачами кафедри інформаційних систем, яки приймають участь у викладанні цієї дисципліни.

Якісне виконання курсового проекту вимагає чіткої організації роботи студента з моменту вибору теми проекту й до його захисту. Студентові надається право вільного вибору теми проекту з урахуванням його схильностей і можливостей найбільше повно застосувати отримані знання.

Після затвердження теми курсового проекту студентові видається завдання на курсове проектування. У ньому наводиться тема курсового проекту, вихідні дані до проекту, зміст пояснювальної записки, завдання на розроблення додатку, строки початку й закінчення роботи над курсовим проектом, обумовлені графіком навчального процесу.

Студент розробляє зміст курсового проекту, обговорює його з керівником, підготовляє вхідні дані і приступає до проектування. У процесі проектування студент повинен регулярно відвідувати консультації керівника, подавати на перевірку йому робочі матеріали.

Курсовий проект студент повинен виконувати самостійно. Оформлений відповідно до пред'явлених вимог проект, студент здає на перевірку керівникові за тиждень до строку захисту.

Захист курсових проектів організовується кафедрою інформаційних систем не пізніше, ніж за тиждень до закінчення семестру.

6.5.3. Структура, зміст і обсяг курсового проекта

Курсовий проект складається з пояснювальної записки та графічного матеріалу, підготовленого у вигляді презентації, яка демонструється при захисті проекта. Обсяг пояснювальної записки становить близько 40 сторінок надрукованого на ПК тексту. Таблиці, діаграми, відеограми, машинограми, вихідні документи можна винести в додатки.

Рекомендується така структура пояснювальної записки:

титульний аркуш;

завдання на курсове проектування;

зміст;

вступ (1 – 2 стор.);

перший розділ. Містить коротку характеристику призначення розробки, постановку завдання, вимоги до програми та програмної документації, етапи розробки (15 стор.);

другий розділ. Містить опис архітектури розробленого додатка, відомості про призначення, умови виконання, настройку та перевірку програми, повідомлення користувачу, що можуть з‘явитися у процесі роботи програми (20 стор.);

висновки (1 стор.);

використана література;

додатки.

6.5.4. Методичні рекомендації щодо оформлення проекту

Важливе значення при роботі над курсовим проектом має його оформлення, до якого пред’являються певні вимоги. Весь матеріал курсового проекту треба розташувати в певній послідовності.

Титульний аркуш оформляється за встановленою формою.

У змісті приводяться заголовки розділів, підрозділів із зазначенням сторінок, з яких вони починаються. При цьому заголовки повинні бути наведені в суворій відповідності з текстом.

Текстовий матеріал курсового проекту друкується на ПК на папері формату А4 (210 х 297 мм). Текст повинен відповідати правилам грамматики й стилістики.

При написанні текстового матеріалу сторінки повинні бути відформатовані наступним чином: ліве поле – 30 мм, праве – 10 мм, верхнє та нижнє – 20 мм.

Абзац повинен починатися з відстані 25 мм від лівого краю сторінки.

Не дозволяється розміщати заголовки й підзаголовки в нижній частині сторінки, якщо на ній не більше 4 рядків наступного тексту.

Кожний розділ курсового проекту повинен починатися з нової сторінки, назви підрозділів, параграфів, пунктів – з абзацу.

Підкреслення найменувань розділів, підрозділів, параграфів не допускається. Відстань між заголовками розділів, підрозділів, параграфів і наступним текстом повинна бути на 5 мм більше відстані між рядками тексту. Заголовки структурних елементів проекту "ЗМІСТ", "ВСТУП", "ВИСНОВКИ", "ВИКОРИСТАНА ЛІТЕРАТУРА" і заголовки розділів треба писати великими друкованими літерами без крапки наприкінці. При друку назви розділів центруються.

Заголовки підрозділів, параграфів і пунктів треба починати з великої букви також без крапки наприкінці. Переноси в середині слова в заголовках не допускаються.

Розділи, підрозділи, параграфи, пункти проекту треба нумерувати арабськими цифрами. Розділи мають порядкову нумерацію, наприклад: 1., 2., 3., і т. д. Підрозділи повинні мати порядкову нумерацію в межах розділу. Номер підрозділу включає номер розділу й порядковий номер підрозділу, які розділяються крапкою, наприклад: 1.1., 1.2., 1.3., і т. д. Номер параграфа включає номер розділу, підрозділу, порядковий номер параграфа, і розділяються вони крапкою, наприклад: 1.1.1., 1.1.2., 1.1.3., і т. д.

Сторінки курсового проекту повинні бути пронумеровані арабськими цифрами в правому верхньому куті без крапки. Нумерація сторінок наскрізна від титульного аркуша до останнього аркуша тексту, включаючи ілюстрації, таблиці, графіки. На титульному аркуші, у завданні на курсовий проект і змісті нумерація сторінок не проставляється.

Викладені у тексті матеріали повинні наочно доповнювати й підтверджувати ілюстрації (схеми, рисунки, графіки, діаграми). Ілюстрації повинні відбивати тему курсового проекту. Студентові необхідно продумати, який матеріал проілюструвати. Це діаграми класів, схеми алгоритмів, схеми інформаційних зв’язків тощо.

Усі ілюстрації іменуються рисунками, позначаються словом "Рис.", їм привласнюється порядковий номер (у межах номера розділу). Рисунки потрібно виконувати на одній сторінці й розташовувати відразу після згадування в тексті.

Таблицю необхідно розташовувати безпосередньо після тексту, у якому вона згадується вперше або на наступній сторінці. На всі таблиці повинні бути посилання. Таблиці послідовно нумеруються в межах розділу проекту. Над правим верхнім кутом таблиці міститься напис "Таблиця" із вказівкою її порядкового номера. Таблиця повинна мати найменування, яке розташовується на наступному рядку після слова "Таблиця".

Перерахування, при необхідності, можуть бути наведені усередині пунктів, їх варто нумерувати порядковою нумерацією арабськими цифрами з дужкою й писати малими літерами з абзацу.

Формули необхідно виділяти з тексту в окремий рядок, залишаючи нижче й вище формули один вільний рядок. Формули треба нумерувати порядковою нумерацією в межах розділу проекту арабськими цифрами в круглих дужках у крайньому правому положенні на рядку. Пояснення значень символів числових коефіцієнтів формули потрібно приводити безпосередньо під формулою в тій же послідовності, у якій вони подані.

Значення кожного символу й числового коефіцієнта необхідно давати з нового рядка. Перший рядок пояснення починати зі словами "де" без двокрапки.

Використана в процесі роботи над курсовим проектом спеціальна література вказується наприкінці проекту перед додатком. Літературу необхідно приводити за абеткою.

Кожна література відбивається в списку в наступному порядку: порядковий номер, у списку прізвище й ініціали автора, назва книги (для статті – її заголовок, назва збірника, журналу, його номер) видавництво, місце й рік випуску.

У тексті пояснювальної записки повинні бути посилання на літературу. При цьому наводиться її порядковий номер, записаний у квадратні дужки.

Додаток потрібно оформляти як продовження проекту. Кожен додаток повинен починатися з нової сторінки й мати змістовний заголовок, написаний великими друкованими літерами. У правому верхньому куті над заголовком повинно бути написане: "Додаток". Додатки позначаються послідовно буквами: А, Б, В, Г, Д, З, К, Л, М, за винятком букв Г, Є, І, Ї, Й, О, Ч, Ь українського алфавіту.

7. Індивідуально-консультативна робота

Індивідуально-консультативна робота здійснюється за графіком індивідуально-консультативної роботи у формі індивідуальних занять, консультацій, перевірки виконання індивідуальних завдань, перевірки та захисту завдань, що винесені на поточний контроль тощо.

З теоретичної частини дисципліни індивідуально-консультативна робота проводиться у вигляді:

1) індивідуальних консультацій, на яких студент отримує відповідь від викладача на конкретні запитання або пояснення певних теоретичних положень чи аспектів їх практичного застосування;

2) групових консультацій, на яких викладач розглядує типові приклади з використання концепцій об‘єктно-орієнтованого програмування.

З практичної частини дисципліни індивідуально-консультативна робота проводиться у вигляді:

1) індивідуальних консультацій, на яких викладач розглядає лабораторні завдання, стосовно яких виникли запитання у студента;

2) групових консультацій, на яких викладач розглядає практичні ситуації, які потребують колективного обговорення.

Індивідуально-консультативна робота для комплексної оцінки засвоєння програмного матеріалу проводиться у вигляді:

1) індивідуального захисту самостійних та індивідуальних завдань;

2) підготовки рефератів для виступу на науковому семінарі;

3) підготовки рефератів для виступу на науковій конференції.

8. Методики активізації процесу навчання

При викладанні навчальної дисципліни "Об‘єктно-орієнтоване програмування" для активізації навчально-пізнавальної діяльності студентів передбачено застосування таких навчальних технологій, як: проблемні лекції, робота в малих групах, семінари-дискусії, презентації (табл. 4).

Таблиця 4

Використання навчальних технологій
для активізації процесу навчання

Методики активізації процесу навчання	Практичне застосування навчальних технологій
1	2
Проблемні лекції направлено на розвиток логічного мислення студентів, коло питань теми обмежується двома-трьома ключовими моментами, використовується досвід закордонних навчальних закладів з роздачею студентам під час лекцій друкованого матеріалу та виділенням головних висновків з питань, що розглядаються. При читанні лекцій студентам даються питання для самостійного розміркування, на які лектор відповідає сам, не чекаючи відповідей студентів	Проблемна лекція з питання: "Основні концептуальні засади об‘єктно-оріентованого програмування" (за темою 1). Проблемна лекція з питань: Реалізація принципу поліморфізму в у мові програмування C# (за темою 4). Проблемна лекція з питань: "Технологія Windows Forms та її застосування щодо розробки сучасних додатків" (за темою 9)
Робота в малих групах дає змогу структурувати лабораторні заняття за формою і змістом, створює можливості для участі кожного студента в роботі за темою заняття, забезпечує формування особистісних якостей та досвіду соціального спілкування	Робота в малих групах при розробленні комплексного додатку з використанням основних бібліотек .NET (Лабораторна робота за модулем 3)

Закінчення табл. 4

1	2
Семінари-діскусії передбачають обмін думками і поглядами учасників з приводу даної теми, а також розвивають мислення, допомагають формувати погляди і переконання, виробляють вміння формулювати думки и висловлювати їх, вчать оцінювати пропозиції інших людей, критично підходити до власних поглядів	Проблемне повідомлення та дискусія при роботі студентів з питання: "Делегати та події" (Лабораторна робота за модулем 4)
Презентації – виступи перед аудиторією, що використовуються для представлення певних досягнень, результатів роботи групи	Презентація розробленої постановки задачі для розв’язання на ПК, архітектури додатку, демонстрація роботи розробленого додатку на даних контрольного приклада (Лабораторна робота за модулем 4)

9. Система поточного та підсумкового контролю знань студентів

Система оцінювання знань, вмінь та навичок студентів передбачає виставлення оцінок за усіма формами проведення занять.

Перевірка та оцінювання знань студентів може проводитись у таких формах:

1. Оцінювання роботи студентів в процесі лабораторних занять.
2. Оцінювання виконання ІНДЗ.
3. Проведення поточно-модульного контролю.
4. Розроблення курсового проекту.
5. Проведення підсумкового іспиту.

Загальна модульна оцінка складається з поточної оцінки, яку студент отримує під час лабораторних занять, оцінки за виконання ІНДЗ та оцінки за виконання модульної контрольної роботи.

Порядок оцінювання знань студентів під час лабораторних занять та виконання ІНДЗ проводиться за 12-бальною шкалою за такими критеріями:

1) розуміння,ступінь засвоєння матеріалу навчальної дисципліни;

2) ознайомлення з рекомендованою літературою, а також із сучасною літературою з питань, що розглядаються;

3) уміння поєднувати теорію з практикою при розробці програм, розв'язанні задач, проведенні розрахунків при виконанні завдань, винесених для самостійного опрацювання, та завдань, винесених на розгляд в аудиторії;

логіка, структура, якість оформлення, стиль викладу матеріалу в письмових роботах і при виступах в аудиторії, вміння обґрунтовувати свою позицію, здійснювати узагальнення інформації та робити висновки.

Оцінка "відмінно" (10 – 12 балів) ставиться за умови відповідності звіту з виконаного лабораторного завдання (ІНДЗ) студента та його усної відповіді при захисті завдання усім чотирьом зазначеним критеріям. Відсутність тієї або іншої складової знижує оцінку на відповідну кількість балів.

При оцінюванні увага також приділяється якості, самостійності та своєчасності здачі виконаних завдань викладачу (згідно з графіком навчального процесу).

Проведення поточно-модульного контролю

Поточно-модульний контроль здійснюється та оцінюється за двома складовими: практичний модульний контроль і теоретичний модульний контроль. Оцінка за практичну складову модульного контролю виставляється за результатами оцінювання знань студента під час лабораторних занять та виконання ІНДЗ. Теоретичний модульний контроль здійснюється у письмової формі.

Для підведення підсумків роботи студентів із змістовного модуля виставляється підсумкова оцінка з поточно-модульного контролю, яка враховує оцінки за практичний та теоретичний модульний контроль.

Проведення підсумкового іспиту

Іспит здійснюється у на комп‘ютерах за екзаменаційними білетами. Екзаменаційний білет складається з двох завдань. Результаті іспиту оцінюються за 12-бальною шкалою. Підсумкова оцінка за іспит є сумою оцінок за кожне завдання. Кожне завдання оцінюється від 0 до 6 балів відповідно до такої шкали (табл. 5).

Таблиця 5

Шкала оцінювання завдань

Бали	Критерії оцінювання
6	Завдання виконано в повному обсязі. Програма працює правильно. Інтерфейс та вихідний код програми задовольняють встановленим вимогам
5	Завдання виконано. Програма працює правильно, але одна з її функціональних можливостей реалізована з порушенням вимог, вказаних в завданні
4	Завдання в основному виконано. Програма працює, але дві з її функціональних можливостей реалізовані з порушенням вимог, вказаних в завданні
3	Завдання виконано, але не в повному обсязі. Програма працює, але не реалізована одна з функціональних вимог, вказаних в завданні, або три з функціональних можливостей програми реалізовані з порушенням вимог, вказаних в завданні
2	Завдання не виконано. Програма запускається, але не реалізовані дві з функціональних вимог, вказаних в завданні, або чотири з функціональних можливостей програми реалізовані з порушенням вимог, вказаних в завданні
1	Програма запускається та частково відповідає постановці завдання, але не реалізовані більше двох з функціональних вимог, вказаних в завданні, або більш чотирьох з функціональних можливостей програми реалізовані з порушенням вимог, вказаних в завданні. Програма не запускається або завершується аварійно, але є програмний код, розроблений студентом, який відповідає постановці завдання
0	Програма відсутня. Програма не містіть програмного коду, розробленого студентом. Програма не відповідає постановці завдання. Програма має явні ознаки несамостійності її розробки

Невиконання або суттєве порушення будь-якої з загальних вимог до програми, наведених далі, знижує кількість балів з завдання на 1.

Вимоги до інтерфейсу користувача

1. Інтерфейс користувача має відповідати постановці завдання.
2. Назви елементів інтерфейсу повинні бути виконані українською або російською мовою.
3. Інтерфейс консольної програми повинен складатися з текстових повідомлень українською або російською мовою, які відносяться до введення та виведення основних та допоміжних даних.

Вимоги до вихідного тексту

1. Додержання принципу інкапсуляції щодо рівнів доступу до полів, властивостей та методів класів.
2. Вихідний код кожного з класів програми повинен міститися в окремому файлі.
3. Наявність коментарів (для класів – призначення класу; для методів – призначення методу, опис параметрів та значення, що повертається) з обов‘язковим використанням відповідних документаційних XML-тегів.

Для підведення підсумків роботи студента з навчальної дисципліни виставляється загальна оцінка, яка враховує оцінки з кожного виду контролю (чотири оцінки поточно-модульного контролю за роботу протягом двох семестрів та оцінки за результатами іспиту):

$$П = 0,6 \cdot Е + 0,4 \cdot (М1 + М2 + М3 + М4),$$

де П – підсумкова оцінка з навчальної дисципліни;

Е – екзаменаційна оцінка;

М1, М2, М3, М4 – оцінки за модулями.

Підсумкова оцінка з дисципліни згідно з методикою переведення показників успішності знань студентів Університету в систему оцінювання за шкалою ECTS конвертується в підсумкову оцінку за шкалою ECTS (табл. 6).

Таблиця 6

Переведення показників успішності знань студентів у систему оцінювання за шкалою ECTS

Відсоток студентів які зазвичай успішно досягають успішної оцінки	Оцінка за шкалою ECTS		Оцінка за бальною шкалою, що використовується в ХНЕУ	Оцінка за національною шкалою
10	відмінне виконання	A	12 – 11	відмінно
25	вище середнього рівня	B	10	
30	взагалі робота правильна, але з певною кількістю помилок	C	9 – 7	добре
25	непогано, але із значною кількістю помилок	D	6	задовільно
10	виконання задовольняє мінімальні критерії	E	5 – 4	
–	потрібне повторне перескладання	FX	3	незадовільно
–	повторне вивчення дисципліни	F	2–1	

Приклад екзаменаційного білета з дисципліни

Міністерство освіти і науки, молоді та спорту України
Харківський національний економічний університет

Напрям підготовки: 6.050101
Дисципліна: "Об‘єктно-орієнтоване програмування" (2 курс)

ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ № 1

Завдання 1.

Є діаграма класів: геометрична фігура (Shape) – абстрактний базовий клас, прямокутник (Rectangle) та прямокутний трикутник (Triangle) – похідні класи.

Клас Shape має абстрактний метод GetArea(), який обчислює та повертає значення площі геометричної фігури. Клас Rectangle має поле A – довжина прямокутника та B – ширина прямокутника. Клас Triangle має поля C, D – катети прямокутного трикутнику. Кожний з похідних класів перевизначає метод GetArea() базового класу.

Розробити програму, яка використовує принцип поліморфізму при обчисленні площ прямокутника та прямокутного трикутника. Для цього програма повинна:

1. Створити динамічний масив (об‘єкт класу System.Collections.-ArrayList).
2. Створити через посилання на тип Shape по одному об‘єкту класів Rectangle та Triangle (значення довжини та ширини прямокутника, катетів прямокутного трикутника ввести з консолі) та додати їх до динамічного масиву.
3. За допомогою оператору циклу foreach для кожного елемента динамічного масиву через посилання на тип Shape викликати метод GetArea().
4. Вивести значення площ прямокутнику та прямокутного трикутника на консоль.

Завдання 2.

Задано матрицю цілих чисел, яка складається з 4 рядків та 4 стовпчиків. Розробити програму з графічним інтерфейсом користувача для знаходження добутку та кількості додатних (>0) елементів матриці.

Программа повинна:

1. Для введення елементів матриці використовувати необхідну кількість компонентів TextBox.
2. Дозволяти користувачу вводити тількі цілочисельні значення елементів матриці в інтервали [-100, 100], інакше повідомляти його про відповідну помилку (компонент ErrorProvider).
3. Виводити вихідні дані (елементи матриці) та результати роботи програми в багаторядковий компонент TextBox.
4. Для організації взаємодії з користувачем використовувати панель меню (компонент MenuStrip) з меню "Файл" (пункти: "Вийти з програми") та меню "Матриця" (пункти: "Обчислити", "Очистити матрицю").

Розроблені до кожного завдання проекти (кожний у своїй папці з іменами
"Task 1", "Task 2") зберегти разом у окремій папці на диску.

Затверджено на засіданні кафедри "Інформаційних систем"
протокол № 4 від 07.12.2011 р.

Зав. кафедрою_____проф. В. С. Пономаренко

Екзаменатор _____доц. Ю. Е. Парфьонов

10. Рекомендована література

Основна

1. Объектно-ориентированный анализ и проектирование с примерами приложений / [Гради Буч, Роберт Максимчук, Майкл Энгл и др.] ; пер. с англ. – М. : ООО "ИД Вильямс", 2008. – 720 с.

2. Об’єктно-орієнтоване програмування : конспект лекцій для студентів напряму підготовки "Комп’ютерні науки" всіх форм навчання / [Парфьонов Ю. Е., Федорченко В. М., Лосєв М. Ю. та ін.]. – Х. : Вид. ХНЕУ, 2010. – 312 с.

3. Троелсен Э. Язык программирования C# 2010 и платформа.NET 4.0 / Эндрю Троелсен ; пер. с англ. – М. : ООО "ИД Вильямс", 2011 – 1392 с.

Додаткова

4. Нэш Т. C# 2010: ускоренный курс для профессионалов / Трей Нєш ; пер. с англ. – М. : ООО "ИД Вильямс", 2010 – 592 с.

5. Фридл Дж. Регулярные выражения / Джон Фридл ; пер. с англ. – СПб. : Питер, 2003 – 464 с.

6. Шилдт Г. C# 3.0: руководство для начинающих / Герберт Шилдт ; пер. с англ. – М. : ИД "Вильямс", 2009 – 688 с.

7. Леоненков А. Самоучитель UML 2 / Александр Леоненков. – СПб. : BHV, 2007 – 576 с.

8. Маки А. Введение в .NET 4.0 и Visual Studio 2010 для профессионалов / Алекс Маки ; пер. с англ. – М. : ООО ИД "Вильямс", 2010. – 416 с.

Ресурси мережі Internet

9. База знань Microsoft Developer Network (MSDN) [Електронний ресурс]. – Режим доступу : http://msdn.microsoft.com/ru-ru.

10. Офіційний сайт компанії Microsoft щодо технологій WPF та Windows Forms [Електронний ресурс]. – Режим доступу : http://windowsclient.net.

11. Internet-інститут інформаційних технологій [Електронний ресурс]. – Режим доступу : www.intuit.ru.

12. База знань Russian Software Developer Network (RSDN) [Електронний ресурс]. – Режим доступу : www.rsdn.ru.

Зміст

НАВЧАЛЬНЕ ВИДАННЯ

Робоча програма
навчальної дисципліни
"ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ"
для студентів напряму підготовки 6.050101
"Комп'ютерні науки"
всіх форм навчання

Укладачі: **Парфьонов** Юрій Едуардович
Щербаков Олександр Всеволодович
Федорченко Володимир Миколайович

Відповідальний за випуск **Пономаренко В. С.**

Редактор **Пушкар І. П.**

Коректор **Мартовицька-Максимова В. А.**

План 2012 р. Поз. № 278.
Підп. до друку Формат 60×90 1/16. Папір MultiCopy. Друк Riso.
Ум.-друк. арк. 3,0. Обл.-вид. арк. 3,75. Тираж прим. Зам. №

Видавець і виготівник – видавництво ХНЕУ, 61166, м. Харків, пр. Леніна, 9а

Свідоцтво про внесення до Державного реєстру суб'єктів видавничої справи
Дк № 481 від 13.06.2001 р.