Міністерство освіти і науки України
Тернопільський національний технічний університет
імені Івана ПУЛЮЯ

кафедра програмної інженерії

М Е Т О Д И Ч Н І В К А З І В К И

щодо самостійної роботи студентів
та модульного контролю знань
з дисципліни

ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

для студентів першого рівня вищої освіти
за спеціальністю No 121 Інженерія програмного забезпечення

Тернопіль 2018

Методичні вказівки щодо самостійної роботи студентів та модульного контролю знань з дисципліни “Об’єктно-орієнтоване програмування” для студентів першого рівня вищої освіти за спеціальністю No 121 Інженерія програмного забезпечення. / Уклад.: М.Р. Петрик, О.Ю.Петрик - Тернопіль: ТНТУ 2018 - 28 с.

Призначені для полегшення засвоєння дисципліни “Об’єктно-орієнтоване програмування” і контролю знань студентів. Складається з урахуванням модульної системи навчання, рекомендацій до самостійної роботи і індивідуальних завдань, тем практичних та лабораторних занять, тестів, екзаменаційних питань, типової форми та вимог для комплексної перевірки знань з дисципліни

Укладачі: М.Р. Петрик, О.Ю.Петрик
Відповідальний за випуск: М.Р.Петрик
Рецензент: Н.Б.Гащин

Розглянуто на засіданні кафедри програмної інженерії, протокол №3 від 12.09.2018р.

Схвалено на засіданні методичної ради факультету комп’ютерно-інформаційних систем і програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя, протокол №2 від 24 жовтня 2018р.

ВСТУП

Метою викладання дисципліни “Об’єктно-орієнтоване програмування” є вивчення сучасного підходу до розробки програмного забезпечення на основі об’єктно-орієнтованих парадигм і технології, набуття студентами знань про об’єктно-орієнтований підхід у програмуванні, засвоєння концепції об'єктно-орієнтованого підходу та його практичне застосування, освоєння можливостей мови С++ з акцентуванням уваги на вирішенні об’єктно-орієнтованих проблем проектування сучасних програмних систем.

За результатами вивчення дисципліни студент повинен продемонструвати такі результати навчання:

1) розуміння основних положень об'єктної моделі, застосування об’єктної моделі, складових частин об'єктно-орієнтованого підходу;
2) знання принципів розробки складних програмних систем, ролі ієрархії, декомпозиції та абстракції у процесі проектування,
3) знати природу об'єкта і класів, відносини між об'єктами та класами, основи класифікації об'єктів, основи об'єктно-орієнтованого підходу до програмування мовою С++
4) вміти застосовувати методологію об'єктно-орієнтованого підходу на складних програмних системах;
5) проводити декомпозицію предметної області, виділяти класи та об'єкти, виділяти ієрархію та вміло використовувати засіб абстракції, правильно будувати відносини між класами, об'єктами у програмних системах;
6) використовувати основи аналізу, проектування та програмування у визначеній предметній області, застосовувати мови об'єктно-орієнтованого спрямування для проектування та програмування на прикладі мови С++.

Вивчення навчальної дисципліни передбачає формування та розвиток у студентів компетентностей:

загальних:

- базові знання фундаментальних наук, в обсязі, необхідному для освоєння загальнопрофесійних дисциплін.
- Володіння основами методів та технологій системного аналізу.
- Дотримання професійної етики програмної інженерії.
- Здатність виявляти, ставити та вирішувати проблеми, приймати обґрунтовані рішення.
- Здатність до пошуку, оброблення та аналізу інформації з різних джерел, проведення досліджень на відповідному рівні.

- Здатність організувати роботу відповідно до вимог безпеки життєдіяльності та володіння основами організації праці на базі знань трудового законодавства і норм охорони праці
- Здатність оцінювати та забезпечувати якість виконуваних робіт.
- Здатність працювати в міждисциплінарній команді.
- Здатність працювати в як індивідуально, так і в команді, мотивувати людей та рухатися до спільної мети.

Фахових:

- Базові уявлення про основи моделювання програмного забезпечення, типи моделей, основні концепції уніфікованої мови моделювання UML.
- Базові уявлення про сучасні психологічні принципи людино-машинної взаємодій, засоби розробки людино-машинного інтерфейсу.
- Верифікація та валідація програмного забезпечення.
- Володіння основами конструювання програмного забезпечення.
- Володіння основами методів та технологій об'єктно-орієнтованого програмування.
- Здатність аналізувати, проектувати та прототипувати людино-машинний інтерфейс.
- Здатність забезпечувати захищеність програм і даних від несанкціонованих дій.
- Здатність застосовувати та створювати компоненти багаторазового використання.
- Здатність розв'язувати математичні, фізичні та економічні задачі шляхом створення відповідних застосувань.
- Здатність розробляти специфікації вимог користувачів до програмного забезпечення.
- Здатність створення технічної документації до програмного проекту.

Для успішного засвоєння програми необхідне знання матеріалу з дисципліни “Основи програмування” (мови C/C++).

Знання, вміння та навички, отримані студентами під час вивчення даної дисципліни будуть необхідними для їхньої професійної діяльності, а також використовуватимуться при написанні курсових та дипломних проектів з програмної інженерії.

1. МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ДО САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ І ТЕМИ ІНДИВІДУАЛЬНИХ ЗАВДАНЬ

1.1 СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ ЗА МОДУЛЬНОЮ СИСТЕМОЮ

Курс "Об'єктно-орієнтоване програмування" є невід'ємною складовою в підготовці спеціаліста з програмної інженерії. Курс розрахований на студентів денної і заочної форм навчання.

Курс об’єднує в собі три складові навчання: теорію, практику і контроль засвоєння матеріалу. Курс наповнений стислими теоретичними відомостями, систематизованими завданнями до лабораторних робіт, перевіреними прикладами програмної реалізації та наборами контрольних тестів. Завдання до лабораторних робіт підібрані індивідуально, за кількістю студентів, виходячи з їхнього початкового рівня підготовки в галузі програмування, реалізація яких, надіюсь, сприяє формуванню логічного мислення, та культури програмування, формує творчий стиль програміста та закладає наукові основи оволодіння сучасними технологіями програмування.

Згідно навчального плану вивчення дисципліни студентами денної і заочної форм навчання триває два семестри.

Курс вивчається протягом двох семестрів, в першому семестрі вивчення передбачає виконання 9 лабораторних робіт, 9 тематичних тестів, 2-х модульних контролів, підсумковий контроль - залік; у другому семестрі - виконання 9 лабораторних робіт, 2-х модульних контролів, підсумковий контроль - екзамен.

Погодинний розподіл тем курсу наведено у таблиці 1.

Таблиця 1 - Погодинний розподіл тем курсу

Назви змістових модулів і тем	Кількість годин											
	денна форма						Заочна форма					
	Усього	у тому числі					Усього	у тому числі				
		л	п	лаб	інд	с.р.		л	п	лаб	інд	с.р
1	2	3	4	5	6	7	8	9	10	11	12	13
Модуль1												
Змістовий модуль1. Основи об'єктно-орієнтованого програмування												
Тема1. Введення в ООП	18	6	–	6	–	6	10	–		–	–	10
Тема2. Операторні функції приведення типів	12	4	–	-	–	8	12	–	–	–	–	12
Тема3. Друзі класу	16	4		6	–	6	11	1	-	–	–	10
Тема4. Перевантаження операцій	20	6	–	6	–	8	17	1	-	2	–	14
Тема5. Успадкування	20	6	–	6	–	8	16	2	-	2	–	12
Тема6. Віртуальні функції і поліморфізм	16	4		6	–	6	12	–	–	-	–	12
Тема7. Виключні ситуації	14	4	–	4	–	6	12	–	–	2	–	10
Разом за змістовим модулем 1	116	34	–	34	–	48	90	4	-	6	–	80
Змістовий модуль2. Потокові класи і об'єкти												
Тема1. Ієрархія потокових класів	12	4	–	-	–	8	18	–	–	–	–	18
Тема2. Форматування потоків	12	4	–	-	–	8	18	–	–	–	–	18
Тема3. Файлові потоки	20	4	–	8	–	8	22	2	-	2	–	18
Тема4. Шаблони (параметризовані типи)	18	4	–	6	–	8	22	2		2	–	18
Тема5. Рядки	20	4		6	–	10	18	–	–	–	–	18
Разом за змістовим модулем 2	82	20		20	–	42	98	4	-	4	–	90

Продовження таблиці 1

Модуль2												
Змістовний модуль3. Стандартна бібліотека шаблонів (STL)												
Тема1. Введення в STL	22	6	–	6	–	10	26	2	-	2	–	22
Тема2. Асоціативні контейнери	22	6	–	6	–	10	30	2	-	4	–	24
Тема3. Проекти	22	8	–	4	–	10	22	–	–	–	–	22
Тема4. Еволюція процесу створення ПЗ	18	6	–	+ 6	–	6	24	2	-	-	–	22
Разом за змістовим модулем 3	84	26		22	–	36	102	6	-	6	–	90
Змістовний модуль4. Програмування під багатозадачні операційні системи на прикладі ОС Windows												
Тема1. Основи програмування під Windows	30	6	–	4	–	20	32	2	-	-	–	30
Тема2. Поняття створення простих додатків MFC	38	8	–	10	–	20	32	2	-	-	–	30
Тема3. Додатки, що використовують засоби Windows Forms	40	8	–	12	–	20	36	2	-	4	–	30
Разом за змістовим модулем 4	108	22	–	26	–	60	100	6	-	4	–	90
Усього годин	390	102	–	102	–	186	390	20	-	20	–	350

1.2 КОНТРОЛЬНІ ПИТАННЯ ДЛЯ САМОПЕРЕВІРКИ

Питання для самоперевірки складені за матеріалами всієї дисципліни “Об’єктно-орієнтоване програмування” і є для студентів допоміжним засобом вивчення пропонованого курсу. Нижче приводяться контрольні питання щодо засвоєння дисципліни.

Змістовий модуль1. Основи об’єктно-орієнтованого програмування

Тема1. Введення в ООП

Призначення курсу. Структура курсу. Рекомендована література. Вимоги і критерії оцінювання. Розвиток технології і мов програмування. Суть об’єктно-орієнтованого підходу до організації програми. Етапи об’єктно-орієнтованого програмування.
Основні принципи об'єктно-орієнтованого програмування. Абстракція, інкапсуляція, наслідування й поліморфізм. Класи і об’єкти.
Синтаксис оголошення класу. Дані класу як механізм реалізації стану об’єкта. Методи класу як механізм реалізації поведінки об’єкта. Специфікатори доступу для забезпечення інкапсуляції.
Статичні елементи класу. Доступ до елементів класу. Вказівники на елементи класу.
Конструктори й деструктор. Конструктор за замовчуванням. Конструктор з параметрами. Конструктор копіювання.
Рекомендації щодо складу класу.

Тема2. Операторні функції приведення типів

Явне приведення типів. Оператори const_cast, dynamic_cast, reinterpret_cast, static_cast і особливості їх застосування.
Стандартні приведення типів.
Приведення вказівників і посилань.

Тема3. Друзі класу

Дружні функції. Доступ до захищених членів класу. Дружні функції і автоматичне приведення типів.
Дружні класи.

Тема4. Перевантаження операцій

Основи перевантаження операторів. Вказівник this.
Перевантаження унарних і бінарних операторів. Обмеження на перевантаження операторів. Перевантаження операторів присвоєння й індексування.
Перевантажені операції new, delete і ->.
Перевантаження операторів як членів класу.
Статичні та віртуальні методи. Множинне перевантаження. Поліморфізм методів. Перевизначення операторів з допомогою дружніх функцій.

Тема5. Успадкування

Поняття наслідування класів. Базові і похідні класи. Форми наслідування. Конструктори похідного класу. Одиничне наслідування. Ієрархія класів. Рівні успадкування.
Множинне наслідування. Композиція. Конструктори при множинному успадкування. Включення і композиція.
Правила доступу для друзів класів і похідних класів. Роль наслідування при проектуванні програм.

Тема6. Віртуальні функції і поліморфізм

Невизначеність при множинному наслідуванні. Віртуальні функції. Правила опису і використання віртуальних методів. Пізнє зв’язування.
Абстрактні класи і чисті віртуальні функції.
Віртуальні базові класи.
Відмінності структур і об'єднань від класів.

Тема7. Виключні ситуації

Технологія обробки виключних ситуацій. Класи виключень. Специфікація виключень.
Ієрархія класів - виключень. Оператор викиду виключень throw. Оператори перехвату виключень try. Оператор обробки виключень catch.
Синтаксис і семантика генерації обробки виключень.

Змістовний модуль2. Потокові класи і об’єкти

Тема1. Ієрархія потокових класів

Поняття потоку введення / виведення даних в C++.
Класифікація потоків. Ієрархія класів потоків.
Підключення потоків. Операції введення-виведення.
Стани потоків.

Тема2. Форматування потоків

Форматування потоків введення / виведення за допомогою функцій класу IOS. Функції width(), precision(), fill(). Маніпулятори. Прості

маніпулятори. Параметризовані маніпулятори. Маніпулятори, визначені користувачем. Форматування потоків введення / виведення за допомогою маніпуляторів. Методи обміну з потоками.

Тема3. Файлові потоки

Файли послідовного доступу з текстовою і бінарною організацією. Стандартні потоки. Файлові потоки. Рядкові потоки. Потоки і типи, визначені користувачем.

Тема4. Шаблони (параметризовані типи)

Шаблони функцій. Шаблон простої функції. Шаблони функцій з параметрами. Вимоги до фактичних параметрів шаблону. Шаблони класів. Наслідування в шаблонах класів. Створення класів об'єктів з допомогою шаблонів

Тема5. Рядки

Стандартний клас String. Загальні відомості. Введення-виведення об'єктів класу String. Опрацювання рядків з допомогою об'єктів класу String.
Конструктори і присвоєння рядків. Операції і функції для роботи з рядками.

Змістовний модуль3. Стандартна бібліотека шаблонів (STL)

Тема1. Введення в STL

Контейнери. Алгоритми. Ітератори, Послідовні контейнери. Вектори. Списки. Черги.
Ітератори як інтелектуальні вказівники.
Спеціалізовані ітератори

Тема2. Асоціативні контейнери

Множини і мультимножини. Відображення та мультивідображення.
Технологія збереження об'єктів користувача.
Функціональні об'єкти

Тема3. Проекти

Багатофайлові програми. Проекти і робочі області.
Робота над проектом. Програми з консольною графікою.
Технологія відладки програм. Покрокове трасування

Тема4. Еволюція процесу створення ПЗ

Об'єктно-орієнтоване програмне забезпечення. Моделювання варіантів використання. Предметна область програмування. Діаграми дій UML класів. Технологія написання коду

Змістовний модуль4. Програмування під багатозадачні операційні системи на прикладі ОС Windows

Тема1. Основи програмування під Windows

Основні поняття і терміни, що використовуються при розробці Windows-додатків. Елементи вікна. Windows – програми і операційна система. Програмування подій. Повідомлення Windows. Windows API.
Типи даних Windows. Структура і організація Windows-програм.

Тема2. Поняття створення кросплатформнх додатків MFC / Qt

Огляд бібліотек класів MFC/ Qt. Основні складові додатку на базі MFC/ Qt.: обробка подій, блоки діалогу, елементи інтерфейсу користувача. Архітектура «документ-представлення».
Види додатків на базі MFC/ Qt

Тема3. Додатки, що використовують засоби Windows Forms / Віджети

Загальне представлення про Windows Forms / Віджети і менеджери компонування діалогових вікон.
Індивідуальне налаштування графічного інтерфейсу користувача; додавання на форму елементів управління, контекстного меню, створення обробників подій, створення діалогового вікна, обробка клацання мишки.

1.3 ТЕМИ ЛАБОРАТОРНИХ ЗАНЯТЬ

Ціль проведення лабораторних занять - закріплення теоретичних положень, викладених на лекціях. Студентам пропонуються лабораторні завдання, проблеми їх вирішення, довідкова література і методичні вказівки, розроблені для самостійної роботи. Викладач контролює хід рішення практичних задач і оцінює уміння студентів застосовувати знання, одержані на лекціях. Зміст лабораторних занять наведено в табл. 2.

Таблиця 2 - Тематика лабораторних занять курсу

№	Тема заняття	Кількість годин	
		ДФН	ЗНФ
1	Програмування на основі абстрактних типів даних	6	-
2	Дружні функції і класи	6	-
3	Перевантаження операцій	6	2
4	Включення	6	2
5	Успадкування. Віртуальність	6	-
6	Виключні ситуації	4	2
7	Потоки	8	2
8	Шаблони	6	2
9	Ознайомлення з рядками	6	
10	Алгоритми роботи з рядками	6	-
11	Ознайомлення з STL	4	2
12	Контейнер vector	6	4
13	Контейнер List	4	-
14	Контейнер deque	4	-
15	Контейнери set і multiset	6	-
16	Контейнери map і multimap	6	-
17	Створення додатку засобами Windows Forms	10	4
18	Об’єктно-орієнтоване проектування	8	
Усього годин		102	20

1.4. ТЕМИ ДЛЯ САМОСТІЙНОГО ВИВЧЕННЯ

Згідно навчального плану і робочої програми більша половина планованих годин для вивчення дисципліни відведена на самостійну роботу студентів, тому доцільним є більш конкретно зупинитись на оцінюванні даного виду роботи.

Самостійна робота, власне, включає опрацювання лекційного матеріалу, рішення індивідуальних завдань до лабораторних робіт, оформлення звітів, підготовка лабораторних робіт до захисту, підготовка до складання екзамену та до тестування, а також вивчення окремих теоретичних питань.

Питання, що виносяться на самостійне опрацювання, оголошуються на кожному першому занятті з теми. Контроль за самостійним вивченням окремих теоретичних питань з теми здійснюється шляхом включення цих питань в тестові тематичні завдання, модульний контроль та підсумковий модульний контроль з дисципліни.

Детальніше про розподіл годин на самостійну роботу див. Таблиця 3

Таблиця 3 - Розподіл годин на самостійну роботу

№	Найменування робіт	Кількість Годин		Рекомендована література
		ДФН	ЗФН	
1	Опрацювання лекційного матеріалу.	30	10	
2	Розв’язування індивідуальних завдань лабораторних робіт.	46	10	
3	Оформлення звітів, підготовка лабораторних робіт до захисту.	18	30	
4	Підготовка до складання екзамену, тестування.	20	40	

Продовження таблиці 3

5	Опрацювання окремих розділів програми, які не виносяться на лекції:	72	260	
5.1	Рекомендації щодо складу класу. Статичні елементи класу. Вказівники на елементи класу. Конструктор копіювання.	8	20	[4, с. 351-383] [1, с. 12-17]
5.2	Операторні функції перевантаження операторів. Перевантаження операцій присвоєння, new і delete, приведення типу, виклику функції, операції індексації.	8	30	[5, с. 312-361]
5.3	Операторні функції приведення типів.	8	30	[5, с. 523-526] [8, с. 457-473]
5.4	Прапори і методи форматування потоків. Маніпулятори. Методи обміну з потоками. Помилки потоків. Маніпулятори, потоки і типи, визначені користувачем.	8	30	[4, с. 429-462]
5.5	Адаптери послідовних контейнерів	8	30	[4, с. 777-826]
5.6	Стандартні узагальнені алгоритми	8	30	[4, с. 840-872]
5.7	Програмування для Windows: графіка	8	30	[4, с. 495- 556] [6, с. 945-1007]
5.8	Робота з діалогами і елементами управління	8	30	[6, с. 1061-1177]
5.9	Створення простого додатку для роботи з базами даних в Qt	8	30	[7, с. 599-611]
Разом		186	350	

2. КОНТРОЛЬ ЗНАНЬ СТУДЕНТІВ

Контроль знань включає в себе – виконання індивідуальних завдань, оформлення звіту з лабораторних робіт, захист звітів із лабораторних робіт, поточне тематичне комп'ютерне тестування, тестування і рішення практичних задач з кожного модуля, підсумковий модульний контроль з дисципліни.

Підсумковий модульний контроль передбачає комп'ютерне тестування знань теоретичних питань з дисципліни і рішення практичного завдання згідно варіанту.

Типові тести для контролю знань

Кожна правильна відповідь на питання передбачає 1 бал:

1.Методологія програмування, основана на представленні програми у вигляді сукупності об'єктів, кожен з яких є екземпляром певного класу, а класи утворюють ієрархію наслідування називається:

а)об'єктно-орієнтованим програмуванням;
б) програмуванням на основі абстрактних типів даних.

2.До найважливіших інструментальних засобів ООП належать:

а)абстрагування типів, інкапсуляція і скриття даних, поліморфізм, наслідування;
б) абстрагування типів, інкапсуляція і скриття даних, поліморфізм, віртуальність;
в)інкапсуляція і скриття даних, поліморфізм, наслідування, дружність.

3. Змінна, яка містить дані і правила (методи) обробки цих даних називається:

а)об'єктом;
б) класом.

4. Дані, оголошені після відкритої фігурної дужки за замовчуванням мають специфікатор private, якщо клас оголошено зі службовим словом:

а)class;
б) struct.

5. Знайдіть помилку:

а)поля класу можуть мати будь-який тип, крім типу цього ж класу (але можуть бути вказівниками або посиланнями на цей клас);
б) поля класу можуть бути описаними з модифікатором const, при цьому вони ініціалізуються лише один раз (з допомогою конструктора) і не можуть змінюватись;

в)поля класу можуть бути описаними з модифікатором static, але не як auto, register чи extern;
г)при описі класу допускається ініціалізація полів.

6.Для визначення методів класу використовується форма:

а)<тип> <ім'я_класу>::< ім'я методу> (<параметри>){ < тіло функції> }
б) <ім'я_класу ><тип>::< ім'я методу> (<параметри>){ < тіло функції> }
в)<тип>< ім'я методу>::<ім'я_класу> (<параметри>){ < тіло функції> }

7. Кожен об’єкт містить:

а)свій екземпляр полів і методів класу;
б) свій екземпляр полів, а методи класу знаходяться в пам’яті в одному екземплярі і використовуються всіма об’єктами спільно.

8.Який загальний формат оператор-функції – елемента класу:

а)тип_результату імя_класу::operator#(список аргументів) { // виконувана операція
}
б) тип_результату operator#(список_параметрів){ // виконувана операція
}

9.Що правильне:

а)при перевантаженні можна змінювати пріоритет операторів;
б) допускається зміна числа операндів оператора;
в)не можна перевантажувати оператори: . :: .* ?: і оператори препроцесора #, ##;
г)не можна перевантажувати оператори: [], (), new, delete і інші.

10.Що неправильне:

а)оператор-функції за винятком присвоєння, наслідуються похідним класом;
б) оператор-функції не можуть мати параметрів по замовчуванню;
в)з допомогою оператор-функції можна створити новий оператор;
г)не можна змінювати значення оператора, пов’язаного з вбудованими типами даних мови С++.

11. Знайдіть помилку:

а)Дружня функція оголошується всередині класу, до елементів якого їй потрібний доступ, з ключовим словом friend. В якості параметра їй має передаватись об’єкт або посилання на об’єкт класу, оскільки вказівник this їй не передається.
б) Дружня функція може бути звичайною функцією або методом іншого, раніше оголошеного класу. На неї не поширюється дія

специфікаторів доступу, місце розміщення її оголошення в класі не принципове.

в)Одна функція може бути дружньою одночасно до декількох класів.

г)Дружня функція наслідується. Тому якщо в базовий клас включається дружня функція, то вона буде дружньою і до похідних класів.

12. У визначенні класу члени класу з ключовим словом private доступні:

а)будь-якій функції програми;

б) у випадку, якщо ви знаєте пароль;

в)методам цього класу;

г)тільки відкритим членам класу.

13. Метод класу завжди має доступ до даних:

а)об'єкта, членом якого він є;

б) класу, членом якого він є;

в)будь-якого об'єкта класу, членом якого він є;

г)класу, оголошеного відкритим.

14. Константний метод, викликаний для об'єкту класу:

а)може змінювати як неконстантні, так і константні поля;

б) може змінювати тільки неконстантні поля;

в)може змінювати тільки константні поля;

г)не може змінювати як неконстантні, так і константні поля.

15. Абстрактний клас використовується, коли:

а)не планується створювати похідні класи;

б) є декілька зв'язків між двома похідними класами;

в)з його допомогою забороняється створення будь-яких об'єктів;

16. Чи істинне твердження, що дружня функція має доступ до скритих даних класу, навіть не будучи його методом?

а)Так

б) Ні

17. Статична функція:

а)повинна викликатися, коли об'єкт знищується;

б) дуже пов'язана з індивідуальним об'єктом класу;

в)може бути викликана з використанням імені класу і імені функції;

г)використовується для створення звичайного об'єкта.

18. Чи може клас мати декілька конструкторів і деструкторів?

а)тільки один конструктор і тільки один деструктор;

б) довільну кількість конструкторів і тільки один деструктор ;

в)довільну кількість деструкторів і тільки один конструктор ;

г)довільну кількість конструкторів і деструкторів.

Кожна правильна відповідь на питання передбачає 2 бали:

1. Встановіть відповідність:

а) Інкапсуляція (пакетування)	а) об’єднання даних і коду, який маніпулює цими даними, а також захищає дані і код від зовнішніх втручань або неправильного використання, в одному об'єкті.
б) Наслідування (успадкування)	б) спосіб повторного використання програмного забезпечення, при якому нові класи створюються на основі вже наявних (базових) класів шляхом копіювання їх атрибутів і функцій (повторне використання кодів).
в) Поліморфізм	в) це властивість, яка дозволяє використовувати одне ім’я для позначення дій, спільних для споріднених класів. При цьому конкретизація виконуваних дій здійснюється залежно від типу даних, які опрацьовуються.
г) Абстрактний тип даних	г) тип даних, властивості якого (область і операції) визначені незалежно від якоїсь конкретної реалізації.

2. Встановіть відповідність між атрибутами і областю дії членів класу:

а) private	а) члени класу визначені тільки всередині класу і поза його межами використовуватися не можуть;
б) public	б) члени класу визначені як для членів класу, так і поза його межами (при умові вказівки імені класу);
в) protected	в) члени класу визначені як для членів класу, так і для його нащадків і друзів.

3. Встановіть відповідність:

а) Конструктор	а) метод класу, має таке ж ім'я, що і клас, призначений для ініціалізації об’єкта класу і викликається автоматично при його створенні;
б) Деструктор	б) метод класу, який має таке ж ім'я, що і клас, призначений для звільнення ресурсів при знищенні об’єкта.

4. Знайдіть помилки:

а) конструктор не повертає значення;
б) не можна отримати вказівник на конструктор;
в) клас може мати декілька конструкторів з різними параметрами для різних видів ініціалізації;
г) параметри конструктора можуть бути довільного типу, крім типу цього класу;
д) значення параметрам можна задавати по замовчуванню всім конструкторам;

5. Конструктор копіювання викликається:

а) При описі нового об’єкта з ініціалізацією іншим об’єктом

б) При передачі об’єкта у функцію за значенням і поверненні об’єкта з функції

в)При перевантаженні операцій

6. Встановіть відповідність між операторами і їх призначенням:

а) const_cast<тип>(вираз)	а) використовується для знищення атрибута const;
б) dynamic_cast<тип>(вираз)	б) призначений для здійснення динамічного приведення виразу до вказівника або посилання на наявний і визначений в даному контексті тип або (void*);
в) reinterpret_cast<тип>(вираз)	
г) static_cast<тип>(вираз)	в) використовується для приведення вказівника одного типу до вказівника іншого типу, в цілий тип і цілої змінної до типу вказівника. Найчастіше використовується для приведення значення-результату з типом void*, до конкретнішого типу.
	г) можна використовувати для приведення вказівника на базовий клас до типу вказівника на похідний клас;

7. Якщо перенавантажується операція арифметичного присвоєння, то результат:

а)передається об'єкту праворуч від операції;

б) передається об'єкту зліва від операції;

в)передається об'єкту, що викликав операцію;

г)має повертатись з функції

8.Ключове слово friend записується в:

а)класі, що дозволяє доступ до іншого класу;

б) класі, що вимагає доступ до іншого класу;

в)розділі прихованих і загальнодоступних компонентів класу;

г)основній програмі при виклику методу класу.

2. Нехай існує клас С з об'єктами obj1, obj2 і obj3. Вираз obj3=obj1-obj2 правильний. Тут перевантажена операція повинна:

а)приймати два аргументи;

б) повертати значення;

в)створювати іменований тимчасовий об'єкт;

г)використовувати об'єкт, що викликав операцію, як операнд.

3. ЕКЗАМЕНАЦІЙНІ ПИТАННЯ

Екзаменаційні білети з дисципліни “Об’єктно-орієнтоване програмування” складені за всім курсом із внесенням питань, що представлені в розділі 1.2 “Контрольні питання для самоперевірки”.

Кількість білетів - 40 шт. Білети включають всі розділи курсу та є практично рівноцінними за своє складнітю.

Формуляр екзаменаційного білету:

Форма № Н-5.05

Тернопільський національний технічний університет імені Івана Пулюя

Освітньо-кваліфікаційний рівень бакалавр
Спеціальність 121 «Інженерія програмного забезпечення» Семестр I
Навчальна дисципліна «Об’єктно-орієнтоване програмування»

ЕКЗАМЕНАЦІЙНИЙ БІЛЕТ № 1

1. Шаблони функцій. Шаблон простої функції. Шаблони функцій з параметрами. Вимоги до фактичних параметрів шаблону.
2. Невизначеність при множинному наслідуванні. Віртуальні функції. Правила опису і використання віртуальних методів. Пізнє зв’язування.
3. Задача.

Затверджено на засіданні кафедри програмної інженерії
Протокол № 1 від „___” __ 201_ року

Завідувач кафедри _____

Екзаменатор _____

4. КОНТРОЛЬНІ ПИТАННЯ З ДИСЦИПЛІНИ

1) Поняття об’єктно-орієнтованого програмування. Визначення програмування на основі абстрактних типів даних Інструментальні засоби ООП: інкапсуляція, поліморфізм, наслідування.

2) Загальний формат оголошення класу. Загальний формат визначення методів класу. Різниця між класом, оголошеним зі службовим словом struct і класом, оголошеним зі службовим словом class

3) Що таке об’єкт? Як оголосити об’єкт класу? Яка умова є обов’язковою для присвоєння одного об’єкта іншому?

4) Чи можна адресу об’єкта передати у функцію в якості аргументу?

5) Що резервує пам’ять – оголошення чи визначення класу?

6) Яка область дії членів класу, визначених з атрибутом public?

7) Що означає атрибут private при оголошенні класу?

8) Хто має доступ до даних класу, оголошених з атрибутом protected?

9) Дайте визначення конструктора. Коли він викликається?

10) Що таке деструктор? Коли він викликається?

11) Скільки конструкторів можна оголосити в класі?

12) Скільки деструкторів можна оголосити в класі?

13) Які Ви знаєте види конструкторів?

14) Які Ви знаєте властивості конструкторів?

15) Що таке конструктор копіювання?

16) У яких випадках викликається конструктор копіювання?

17) Що таке конструктор з параметрами? Що таке конструктор по замовчуванню? Сформулюйте основні властивості конструкторів.

18) Який тип можуть мати параметри конструктора?

19) Що таке вказівник this? Яка його функція при ООП?

20) Чи правильне твердження, що для кожного об’єкта класу створюється своя копія даних класу, а методи класу існують для всіх об’єктів у одному екземплярі?

21) Наведіть приклад використання вказівника this.

22) Як задати параметри для методу по замовчуванню?

23) Дайте визначення дружньої функції.

24) Як оголосити дружню функцію?

25) Дружня функція є методом класу?

26) Чи може дружня функція бути дружньою більше, ніж до одного класу?

27) Чи істинне твердження, що дружня функція має доступ до скритих даних класу, навіть не будучи його методом?

28) Що таке дружній клас?

29) Які Ви знаєте способи перевантаження функцій?

30) Запишіть основну форму унарної оператор-функції – методу класу

31) Запишіть основну форму унарної оператор-функції – дружньої до класу

32) Запишіть загальний формат бінарної оператор-функції – методу класу

33) Запишіть загальний формат бінарної оператор-функції – дружньої до класу

34) Які Ви знаєте обмеження на перевантаження операторів?

35) Що таке базовий клас?

36) Що таке похідний клас?

37) Що таке безпосередній базовий клас?

38) Які Ви вивчали приклади поліморфізму?

39) Запишіть формат опису конструктора при множинному наслідуванні

40) Поясніть різницю між простим і множинним наслідуванням.

41) Який атрибут має перевагу: атрибут при наслідуванні чи атрибут у безпосередньому базову класі?

42) Який формат опису класу при простому наслідуванні?

43) Який клас називається абстрактним?

44) Який клас називається поліморфним?

45) У базовому класі встановлений статус доступу public. Яким має бути модифікатор доступу, щоб в похідному класі одержати статус доступу protected?

46) Чи є кожен об’єкт похідного класу об’єктом відповідного базового класу?

47) Чи є кожен об’єкт базового класу об’єктом класів, породжених даним базовим класом?

48) Який формат опису класу при множинному наслідуванні?

49) Поясніть різницю між дружньою, віртуальною і перевантаженою функціями

50) Які Ви знаєте атрибути доступу при наслідуванні?

51) Як відбувається виклик конструкторів при наслідуванні?

52) Як відбувається виклик деструкторів при наслідуванні?

53) Запишіть формат опису конструктора при простому наслідуванні

54) Наведіть приклад ситуації, що приводить необхідності оголошення класу віртуальним.

55) Які Ви знаєте види множинного наслідування? Охарактеризуйте їх (можна схематично).

56) Нехай базовий клас містить метод func(), а похідний клас не має методу з таким ім'ям. Чи може об'єкт похідного класу мати доступ до методу func()?

57) Допустимо, що базовий і похідний класи включають методи з однаковими іменами. Який з методів буде викликаний об'єктом похідного класу, якщо не використана операція дозволу імені?

58) Наведіть приклад ситуації, що приводить необхідності використання віртуальної функції.

59) Визначення потоку. Класифікація потоків введення / виведення за напрямком передачі даних і за способом створення

60) Схема наслідування класів потоків

61) Засоби форматування потоків виведення

62) Загальний формат і призначення функції width(). Навести приклад використання

63) Загальний формат і призначення функції precision().Навести приклад використання

64) Загальний формат і призначення функції fill().Навести приклад використання

65) Загальний формат і призначення функції setf ().Навести приклад використання

66) Загальний формат і призначення функції unsetf().Навести приклад використання

67) Форматування потоків введення / виведення за допомогою маніпуляторів

68) Маніпулятори введення, визначені користувачем. Загальний формат

69) Маніпулятори виведення, визначені користувачем. Загальний формат

70) Схема роботи з файлами через потоки

71) Створення потоку введення, виведення, введення-виведення.

72) Функції, що дозволяють зв'язати потік з файлом. Прототип функції open(). Режими відкриття файла через потоки. Методи для запису у файл write і put: прототип, приклад.

73) Методи переміщення поточної позиції введення (виведення) seekp і seekg: прототип, приклад

74) Методи визначення позиції покажчика зчитування чи запису tellg() і tellp(). Методи читання з файлу read() і get(): прототип, приклад. Які Ви знаєте способи форматування потоків?

75) Шаблон функції. Загальна форма визначення шаблону функції

76) Кількість та типи параметрів шаблону функції

77) Шаблон простої функції. Шаблони функцій з параметрами. Вимоги до фактичних параметрів шаблону

78) Основні властивості параметрів шаблона функції. Способи перевантаження шаблонних функцій. Загальна форма визначення шаблону класу. Оголощення об'єкта шаблону класу. Шаблони і наслідування. Шаблони і друзі.

79) Поняття механізму виключних ситуацій. Приклади виключень.

80) Випадки, коли необхідно передбачити опрацювання виключних ситуацій

81) Оператори для реалізації механізму обробки виключень. Оператор перехвату виключень try. Оператори обробки виключень catch. Оператори викиду виключень throw. Синтаксис і семантика генерації обробки виключень

82) Контейнери. Алгоритми. Ітератори, Послідовні контейнери.
83) Вектори.
84) Списки.
85) Черги.
86) Ітератори як інтелектуальні вказівники.
87) Спеціалізовані ітератори
88) Множини і мультимножини.
89) Відображення та мультивідображення. Технологія збереження об'єктів користувача.
90) Функціональні об'єкти
91) Багатофайлові програми. Проекти і робочі області.
92) Робота над проектом. Програми з консольною графікою. Технологія відладки програм. Покрокове трасування
93) Об'єктно-орієнтоване програмне забезпечення. Моделювання варіантів використання. Предметна область програмування.
94) Діаграми дій UML класів. Технологія написання коду

95) Основи програмування під Windows Основні поняття і терміни, що використовуються при розробці Windows-додатків. Елементи вікна. Windows – програми і операційна система. Програмування подій. Повідомлення Windows. Windows API.

96) Типи даних Windows. Структура і організація Windows-програм.

97) Поняття створення кросплатформнх додатків MFC / Qt

98) Огляд бібліотек класів MFC/ Qt. Основні складові додатку на базі MFC/ Qt.: обробка подій, блоки діалогу, елементи інтерфейсу користувача. Архітектура «документ-представлення».
99) Види додатків на базі MFC/ Qt

100) Додатки, що використовують засоби Windows Forms / Віджети

101) Загальне представлення про Windows Forms / Віджети і менеджери компонування діалогових вікон.
102) Індивідуальне налаштування графічного інтерфейсу користувача; додавання на форму елементів управління, контекстного меню, створення обробників подій, створення діалогового вікна, обробка клацання мишки.
103) Робота з діалогами і елементами управління Windows Forms
104) Створення простого додатку для роботи з базами даних в Qt

ПЕРЕЛІК РЕКОМЕНДОВАНОЇ ЛІТЕРАТУРИ

1. «Об’єктно-орієнтоване програмування» для студентів 121 - Інженерія програмного забезпечення, рівень вищої освіти перший (бакалаврський) / Уклад. Петрик М.Р., Петрик О.Ю. –Тернопіль: ТНТУ імені Івана Пулюя, 2014. [Електронний ресурс]. - Режим доступу: URL: *https://dl.tntu.edu.ua/content.php? course=1472*

2. О. Петрик. Об’єктно-орієнтоване програмування в середовищі С++: Навчальний посібник. Лабораторний практикум – Тернопіль, видавництво ТНТУ імені Івана Пулюя, 2011. – 188 с.

3. Об’єктно-орієнтоване програмування. Вимоги до курсового проектування. /Уклад.: М.Р. Петрик, О.Ю. Петрик - Тернопіль: ТНТУ 2015 - 32 с.

4. Б.Страуструп. ПРОГРАММИРОВАНИЕ: принципы и практика использования С++,«Вильямс», 2011. - 1248 с.

5. Р. Лафоре. Объектно-ориентированное программирование в С++. 4-е издание. Издательство: Питер. Серия: Классика computer science, 2005.- 928 с.

6. Айвор Хортон. Visual C++ 2010: полный курс. Изд-во: Диалектика-Вильямс, 2011. – 1216 с.

7. Шлее М. Qt 5.10. Профессиональное программирование на C++. — СПб.: БХВ-Петербург, 2018. — 1072 с.: ил.

8. Шилдт Г. Полный справочник по C++. 4-е издание. - М.: Изд.дом «Вильямс», 2006г.–800с.

9. Дейтел Х., Дейтел П. Как программировать на С++. 5-е изд. - М.:ООО „Бином-Пресс”, 2008г.–1456с

10. Культин Н. Самоучитель. Основы программирования в Microsoft Visual C++ 2010, Изд-во: BHV-СПб, 2010. -384 с.

11. Дэвид Р. Мюссер, Жилмер Дж. Дердж, Атул Сейни. Книга С++ и STL: справочное руководство, 2-е издание, Изд-во: Диалектика-Вильямс, 2010, 432 с.

12. Галовиц Яцек «C++ 17 STL. Стандартная библиотека шаблонов». - СПб.: Питер, 2018. - 432 с.

НАВЧАЛЬНЕ ВИДАННЯ МЕТОДИЧНІ ВКАЗІВКИ ЩОДО САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ ТА МОДУЛЬНОГО КОНТРОЛЮ ЗНАНЬ

з дисципліни “Об’єктно-орієнтоване програмування”

для студентів першого рівня вищої освіти

за спеціальністю No 121 Інженерія програмного забезпечення

Укладачі: Петрик Михайло Романович,

Петрик Оксана Юліанівна