КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ КОМП'ЮТЕРНИХ НАУК ТА КІБЕРНЕТИКИ

Кафедра обчислювальної математики

«ЗАТВЕРДЖУЮ»
Заступник декана
з навчальної роботи
_____ Кашпур О.Ф.
«28» 08 2020 року

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ
ОБ'ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

для студентів

галузь знань	**12 «Інформаційні технології»**
спеціальність	**124 «Системний аналіз»**
освітній рівень	**бакалавр**
освітня програма	**«Системний Аналіз»**
вид дисципліни	**вибіркова**

Форма навчання	**денна**
Навчальний рік	**2021/2022**
Семестр	**3, 4**
Кількість кредитів ECTS	**7**
з них семестр 3	**4**
семестр 4	**3**
Мова викладання, навчання та оцінювання	**українська**
Форма заключного контролю	**залік, іспит**

Викладачі: **ас. Денисов С.В.** (лекції, лабораторні заняття)**,**
к.ф.-м.н., ас. Оноцький В.В. (лабораторні заняття),
ас. Тимошенко А.А. (лабораторні заняття)

Пролонговано: на 20__/20__ н.р. _____ (_____) «31» 08 2022р.
(підпис, ПІБ, дата)

на 20__/20__ н.р. _____ (_____) «__»___ 20__р.
(підпис, ПІБ, дата)

Розробники:

Клюшин Дмитро Анатолійович, д.ф.-м.н, професор кафедри обчислювальної математики,

Денисов Сергій Вікторович, асистент кафедри обчислювальної математики

ЗАТВЕРДЖЕНО
Зав. кафедри обчислювальної математики

_____ (Ляшко С.І)

Протокол № 1 від «27» серпня 2020 р.

Схвалено Гарантом освітньо-професійної програми першого рівня вищої освіти

«Системний аналіз» _____ М.М. Шарапов

«28» серпня 2020 року

Схвалено науково-методичною комісією факультету комп'ютерних наук та кібернетики

Протокол від «28» серпня 2020 року №1

Голова науково-методичної комісії _____ **(Омельчук Л.Л.)**
(підпис) **(прізвище та ініціали)**

«28» серпня 2020 року

1. Мета дисципліни – розвиток компетенції ефективного використання ООП для створення алгоритмічно та технологічно навантажених програмних проектів. Знайомство з підходами до розробки індустріального та академічного програмного забезпечення. Отримання навичок програмування мовою C/C++.

2. Попередні вимоги до опанування або вибору навчальної дисципліни (*за наявності*):

1. Знати: основні поняття та концепції програмування, алгебру та математичний аналіз на базовому рівні (об'єм першого курсу університету), суть поняття алгоритму.

2. Вміти: створювати програми будь-якою мовою, читати та аналізувати математичні тексти, реалізовувіти прості алгоритми.

3. Володіти елементарними навичками: роботи з комп'ютером, пошуку інформації в інтернеті, користування системами перекладу.

3. Анотація навчальної дисципліни:

Навчальна дисципліна "Об'єктно-орієнтоване програмування" є складовою освітньо-професійної програми підготовки фахівців за першим (*бакалаврським*) рівнем вищої освіти. В її рамках досліджуються переваги та недоліки об'єктно-орієнтованого підходу, розглядаються підходи до ефективної розробки програмного забезпечення, практикується використання інструментів індустріальної розробки програмних систем, систем контролю версій. Також відбувається розвиток вмінь реалізації алгоритмів та вивчаються основи аналізу складності.

Дисципліна викладається у **3** та **4** семестрах **2** курсу в обсязі **210 год. (7 кредитів ECTS)** зокрема: *лекції – 54 год., лабораторні – 48 год., консультації – 2 год., самостійна робота – 106 год.* У курсі передбачено **3** *частини* та **3** *контрольні роботи*. Семестр **3** завершується **заліком**, семестр **4** завершується **іспитом**.

В результаті вивчення навчальної дисципліни студент повинен:

знати основні поняття об'єктно-орієнтованого програмування, підходи до розробки індустріального програмного забезпечення, основні шаблони та антишаблони проектування, основні підходи оцінки складності алгоритмів.

вміти створювати програми мовою C++; застосовувати професійні інструменти розробки програм; використовувати системи контролю версій для індивідуальної та командної розробки; аргументовано обирати архітектуру, технології та алгоритми для розв'язання задач; реалізовувати самостійно потрібні варіанти алгоритмів.

Для допуску до дисципліни „Об'єктно-орієнтоване програмування" освітньо-професійної програми «Системний аналіз» студент повинен опанувати компетентності та результати навчання, які надає дисципліна „Програмування" програми «Системний аналіз».

4. Завдання (навчальні цілі):

Отримання компетентностей в професійній розробці програмного забезпечення для індустріальних та академічних проектів. Відповідно до освітньої кваліфікації бакалавра з системного аналізу дисципліна спрямована на досягнення таких компетентностей випускника:

- Здатність застосовувати знання у практичних ситуаціях (К02)
- Здатність планувати і управляти часом (К03)
- Здатність до пошуку, оброблення та аналізу інформації з різних джерел (К07)
- Здатність працювати автономно (К10)
- Здатність до комп'ютерної реалізації математичних моделей реальних систем і про цесів; проектувати, застосовувати і супроводжувати програмні засоби моделювання,

прийняття рішень, оптимізації, обробки інформації, інтелектуального аналізу даних (К22)

- Здатність використовувати сучасні інформаційні технології для комп'ютерної реалізації математичних моделей та прогнозування поведінки конкретних систем а саме: об'єктно-орієнтований підхід при проектуванні складних систем різної природи, прикладні математичні пакети, застосування баз даних і знань (К23)
- Здатність організовувати роботу з аналізу та проектування складних систем, створення відповідних інформаційних технологій та програмного забезпечення (К24)

5. Результати навчання за дисципліною:

Результат навчання (1. знати; 2. вміти; 3. комунікація; 4. автономність та відповідальність)		**Форми (та/або методи і технології) викладання і навчання**	**Методи оцінювання та пороговий критерій оцінювання (за необхідності)**	**Відсоток у підсумковій оцінці з дисципліни**
Код	**Результат навчання**			
РН1.1	*Знати основні поняття об'єктно-орієнтованого програмування*	*Лекція, лабораторне заняття*	*контрольна робота, іспит*	15%
РН1.2	*Знати основні підходи до оцінки складності алгоритмів*	*Лекція, лабораторне заняття*	*контрольна робота, іспит*	5%
РН1.3	*Знати принципи та найбільш поширені шаблони об'єктно-орієнтованого проектування програмного забезпечення*	*Лекція, лабораторне заняття*	*контрольна робота, іспит*	5%
РН2.1	*Вміти створювати програми мовою С++*	*Лабораторне заняття, самостійна робота*	*захист лабораторних робіт (ЛР), контрольна робота, іспит*	15%
РН2.2	*Вміти розбирати та реалізовувати алгоритми для розв'язання наближених до реальних задач*	*Лабораторне заняття, самостійна робота*	*захист ЛР*	20%
РН2.3	*Вміти використовувати інструментарій для професійної розробки програм*	*Лекція, лабораторне заняття, самостійна робота*	*захист ЛР, іспит*	10%
РН3.1	*Аргументувати власний вибір підходів до розв'язання задачі, спілкуватися з колегами з питань проектування та розробки програм*	*Лабораторне заняття*	*захист ЛР*	10%
РН4.1	*Організовувати свою*	*Самостійна*	*захист ЛР*	10%

	самостійну роботу для досягнення результату	*робота*		
РН4.2	*Відповідально ставитися до виконуваних робіт, нести відповідальність за їх якість*	*Лабораторна робота*	*захист ЛР*	10%

6. Співвідношення результатів навчання дисципліни із програмними результатами навчання

Результати навчання дисципліни **Програмні результати навчання**	**РН 1.1**	**РН 1.2**	**РН 1.3**	**РН 2.1**	**РН 2.2**	**РН 2.3**	**РН 3.1**	**РН 4.1**	**РН 4.2**
(з опису освітньої програми)									
ПР08. Володіти сучасними методами розробки програм і програмних комплексів та прийняття оптимальних рішень щодо складу програмного забезпечення, алгоритмів процедур і операцій.	+	+	+	+		+			
ПР09. Вміти створювати ефективні алгоритми для обчислювальних задач системного аналізу та систем підтримки прийняття рішень.		+		+	+			+	+
ПР13. Проектувати, реалізовувати, тестувати, впроваджувати, супроводжувати, експлуатувати програмні засоби роботи з даними і знаннями в комп'ютерних системах і мережах.				+	+	+	+	+	+
ПРСАПР 3. Вміти проектувати, реалізовувати, тестувати, впроваджувати, супроводжувати та експлуатувати програмне забезпечення комп'ютерних систем і мереж обробки даних і знань				+	+	+	+	+	+

7. Схема формування оцінки.

7.1 Форми оцінювання студентів:

- семестрове оцінювання:

Перший семестр

1. Контрольна робота 1: РН 1.1, РН 2.1 — 20 балів/10 балів.

2. Контрольна робота 2: РН 1.1, РН 1.3, РН 2.1 — 20 балів/10 балів.

3. Лабораторна робота 1 (проект): РН2.1, РН3.1, РН4.1, РН4.2 – 20 балів/10 балів.

4. Лабораторна робота 2 (проект): РН2.1, РН2.3, РН3.1, РН4.1, РН4.2 – 20 балів/10 балів.

5. Лабораторна робота 3 (проект): РН1.3, РН2.1, РН2.3, РН3.1, РН4.1, РН4.2 – 20 балів/10 балів.

Другий семестр

1. Контрольна робота 1: РН 1.2, РН 1.3, РН 2.1 — 15 балів/9 балів.

2. Лабораторна робота 1 (проект): РН2.1, РН2.2, РН2.3, РН3.1, РН4.1, РН4.2 – 15 балів/9 балів.

3. Лабораторна робота 1 (проект): РН2.1, РН2.2, РН2.3, РН3.1, РН4.1, РН4.2 – 15 балів/9 балів.
4. Лабораторна робота 1 (проект): РН2.1, РН2.2, РН2.3, РН3.1, РН4.1, РН4.2 – 15 балів/9 балів.

- підсумкове оцінювання

Перший семестр **(у формі заліку)**

Згідно пп. 4.6.1 та 7.1.5 «Положення про організацію освітнього процесу у Київському національному університеті імені Тараса Шевченка» залік виставляється на підставі поточного контролю (див. семестрове оцінювання) як сума оцінок/балів за всіма успішно оціненими результатами навчання; оцінки нижче від мінімального порогового рівня до підсумкової оцінки не додаються.
До заліку допускаються всі студенти.

Другий семестр **(у формі іспиту)**

- максимальна кількість балів які можуть бути отримані студентом: 40 балів;
- результати навчання які будуть оцінюватись: РН1.1, РН1.2, РН1.3, РН2.1, РН2.3;
- форма проведення і види завдань: письмова із захистом відповіді, два теоретичних питання (5 та 10 балів) та два практичних завдання (по 10 балів), і ще 5 балів додаються при наявності комітів практичних завдань з інтервалом до 10 хвилин в систему контролю версій.

Студент допускається до іспиту, якщо він під час семестру набрав більше 36 балів, причому лабораторні роботи було виконано як мінімум на 60%.

Для отримання загальної позитивної оцінки з дисципліни оцінка за іспит не може бути меншою 24 балів.

7.2. Організація оцінювання.

Терміни проведення форм оцінювання в 3 семестрі:

1. Контрольна робота 1: до 6 тижня семестру.
2. Контрольна робота 2: до 14 тижня семестру.
3. Лабораторна робота 1 (проект): до 3 тижня семестру.
4. Лабораторна робота 2 (проект): до 8 тижня семестру.
5. Лабораторна робота 3 (проект): до останнього тижня семестру.

Терміни проведення форм оцінювання в 4 семестрі:

1. Контрольна робота 1: до 10 тижня семестру.
2. Лабораторна робота 1 (проект): до 4 тижня семестру.
3. Лабораторна робота 2 (проект): до 9 тижня семестру.
4. Лабораторна робота 3 (проект): до до останнього тижня семестру.

Для контрольної роботи дозволяється одне перескладання, але без поважних причин (як то хвороба) можна отримати не більше 80% балів.

Лабораторні роботи можна здавати після визначеного терміну, але за перший тиждень запізнення віднімається 20% балів, і за кожен наступний по 10% (після 4-го тижня бали далі не віднімаються, на 60% можна здати завжди).

7.3 Шкала відповідності оцінок

Відмінно / Excellent	90-100
Добре / Good	75-89
Задовільно / Satisfactory	60-74

Незадовільно / Fail	0-59
Зараховано / Passed	60-100
Не зараховано / Fail	0-59

8. Структура навчальної дисципліни. Тематичний план лекцій і лабораторних занять

СЕМЕСТР 3

№ лекції	Назва лекції	Кількість годин		
		Лекції	Лаборат. занять	Сам. р-та
1.	**Тема 1.** Мотивація використання мов C/C++ та їх місце в сучасній індустрії розробки ПЗ. Переваги та проблеми технологій на базі цих мов.	2		2
2.	**Тема 2.** Основи синтаксису мови C. Переваги статичної типізації. Стандартні базові елементи — змінні, оператори, умовні конструкції, цикли, функції. Масиви та рядки в C.	2	4	4
3.	**Тема 3.** Компільовані мови програмування. Переваги та недоліки. Основи процесу збирання C++ проекту, знайомство з make та cmake. Препроцесор, директива #include. Розбиття проекту на файли, стандартні помилки.	4	2	4
4.	**Тема 4.** Повторення ідей ООП без прив'язки до мови. Мотивація використання та проблеми ООП. Базова реалізація ООП в C++ - основний синтаксис роботи з класами та успадкуванням. Основні принципи та шаблони в ООП архітектурі.	2		2
	Контрольна робота 1	2		
5.	**Тема 5.** Мотивація та історія розвитку систем контролю версій. Знайомство з git.	2	2	6
6.	**Тема 6.** Основи ручного керування пам'яттю в C++. Вказівники.	2	4	6
7.	**Тема 7.** Особливості та специфічні можливості класичного C++. Область видимості змінних, передача параметрів за вказівником та посиланням, стандартні підходи та проблеми.	2	2	4
8.	**Тема 8.** Заглиблення в ООП на C++. Нюанси роботи з успадкуванням, конструкторами та деструкторами. Реалізація поліморфізму. Множинне успадкування в C++. Списки ініціалізації. Статичні члени класу.	4	4	8
9.	**Тема 9.** Перевантаження операторів в C++	2	4	8
10.	**Тема 10.** Структурована обробка помилок (Exceptions). Мотивація. Можливі підходи до обробки помилок. Ієрархія винятків в C++. Класичні проблема з винятками в C++ (Cargill stack) та їх вирішення.	4	2	4
11.	**Тема 11.** Використання функцій як "об'єктів першого класу" в C++. Вказівники на фукнції, анонімні функції (лямбди). Поняття замикання (closure). Синтаксис в C++.	4	2	4

12.	**Тема 12.** Шаблони (узагальнення) в C++. Мотивація, сценарії використання. Проблеми.	1	2	4
	Контрольна робота 2	1		
	ВСЬОГО	34	28	56

Загальний обсяг *120 год.,* в тому числі:
Лекцій **– *34** год.*
Лабораторні заняття **- *28** год.*
Консультації – ***2** год.*
Самостійна робота **- 56** *год.*

Тематика лабораторних проектів:

1. Реалізація найпростіших алгоритмів на C++ (генератори псевдовипадкових чисел)
2. Реалізація класу довгої арифметики з ефективними алгоритмами
3. Візуалізація даних та геометричні алгоритми на C++

Перелік контрольних питань:

1. Змінні. Суть поняття, приклади використання, типи змінних, область видимості.
2. Умовні конструкції. Варіанти використання. Вкладення та комбінування умовних конструкцій. Тернарний оператор.
3. Цикли. Вкладення циклів. Достроковий вихід та продовження. Приклади циклів.
4. Рекурсія і особливості її використання. Приклади рекурсії.
5. Функції в C++
   a. призначення, базовий синтаксис,
   b. варіанти передачі параметрів (за значенням, за посиланням, через вказівник) та повернення значень.
   c. Локальні змінні, доступ до глобальних змінних. Статичні змінні в функції.
   d. Використання функціонального стилю - лямбда функції, збереження функцій в змінну, передача функцій в інші функції, функції другого порядку, замикання.
6. Масиви в C++.
   a. Статичний масив. Розуміння суті змінної, яка “містить” масив.
   b. Динамічний масив. Керування пам’яттю.
7. Declaration та definition для C++ змінних, функцій та класів.
8. Розбиття C++ проекту на файли – директива #include, особливості використання.
9. Поняття ООП. Мета, переваги, недоліки та проблеми. Основні концепції ООП.
10. Реалізація ООП в C++.
    a. Базовий синтаксис класів
    b. Реалізація інкапсуляції
    c. Створення та використання об’єктів
    d. Конструктори
    e. Деструктори
    f. Ключове слово this
    g. Успадкування. Нюанси інкапсуляції при успадкуванні. Зверенення до членів базових класів.
    h. Поліморфізм - реалізація та використання
    i. Статичні члени класу
    j. Перевантаження операторів. Унарні, бінарні. Оператор виводу в потік. Різниця між перевантаженням в класі та поза класом.
    k. Множинне успадкування
    l. Списки ініціалізації.

11. Шаблони (узагальнення, templates) в C++. Призначення, найпростіші сценарії використання.

12. Структурована обробка помилок (exceptions та робота з ними - створення класів exceptions, варіанти перехоплення, поведінка в ситуації з ієрархією класів)

13. Системи контролю версій – призначення, основні сценарії використання (на прикладі git).

СЕМЕСТР 4

№ лекції	Назва лекції	Кількість годин		
		Лекції	Лаборат. занять	Сам. р-та
13.	**Тема 1.** Важливість використання ефективних алгоритмів. О-нотація, оцінки складності — в гіршому випадку, в середньому та амортизована складність. Складність алгоритмів пошуку та сортування. Ідея сортування підрахунком та кишенькового сортування.	2	2	4
14.	**Тема 2.** Хеш-таблиця. Хеш-функції. Переваги та проблеми для вирішення задачі швидкого пошуку. Задача перевірка слова за словником.	2	2	6
15.	**Тема 3.** Пошук на структурах без прямого доступу. Бінарне дерево пошуку. Проблема балансу, підходи до вирішення (AVL та RB). Розвинення ідеї - B-дерева.	2	2	4
16.	**Тема 4.** Командна розробка програмного забезпечення. Класична схема роботи команд. Інструментарій. Використання гілок (branches) в git. Git flow.	1	2	8
	Контрольна робота 1	1		
17.	**Тема 5.** Автоматизоване тестування коду. Мотивація. Інструментарій для C++. Google test.	2	2	6
18.	**Тема 6.** Можливості сучасного C++. Семантика переміщення.	2	2	6
19.	**Тема 7.** Основи багатопоточного програмування на C++. Потоки та процеси. Проблеми — race condition, deadlock. Синхронізація.	4	4	8
20.	**Тема 8.** Використання загальновідомих бібліотек C++. STL та boost. Boost asio.	4	4	8
	ВСЬОГО	20	20	50

Загальний обсяг *90 год.,* в тому числі:
Лекцій **– *20 год.***
Лабораторні заняття **- *20 год.***
Самостійна робота **- 50** *год.*

Тематика лабораторних проектів:

1. Реалізація алгоритмів для реальних задач з графічним відображенням. Пошук шляхів на соціальних та транспортних мережах.
2. Клас матриці з набором класичних алгоритмів та перевантаженими операторами
3. Проста мережева графічна гра

Перелік питань для підготовки до іспиту:

1. Всі контрольні питання із списку першого семестру.
2. Поняття складності алгоритму. Види оцінок складності.
3. Основні структури даних та алгоритми для задачі пошуку елементу в колекції та сортування.
4. Використання гілок системи контролю версій при командній та самостійній розробці.
5. Мета та підходи до автоматизованого тестування коду.
6. Операція переміщення в C++. Призначення, сценарії використання, синтаксис, проблеми.
7. Використання багатопоточності. Переваги та проблеми. Синхронізація.
8. Можливості бібліотек STL та Boost. Boost asio для роботи з мережею.

9. Рекомендовані джерела:

Основні

1. Бєлов Ю.А., Карнаух Т.О., Коваль Ю.В., Ставровський А.Б. Вступ до програмування мовою C++. Організація обчислень. – К. : Видавничо-поліграфічний центр "Київський університет", 2012. – 175 с

2. Повна документація з C++ - https://en.cppreference.com/

3. Буч Г. Обьектно-ориентированное программирование. – Киев, Диалектика, 1993

4. Герберт Шилдт. C++. Полное руководство. Классическое издание. – М.: Издательский дом “Вильямс”, 2016.

5. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ. – М.: Издательский дом “Вильямс”, 2006.

6. Клюшин Д.А. Полный курс C++. Профессиональная работа. – М.: Издательский дом “Вильямс”, 2004.

7. Скотт Мейерс, Эффективный и современный C++: 42 рекомендации по использованию C++11 и C++14. – М.: Диалектика-Вильямс, 2020. – 304 с

8. Кнут Д. Искусство программирования. Т. 1-3. 3-е изд. – М.: Издательский дом “Вильямс”, 2000.

Додаткові

1. Ахо А., Хопкрофт Д., Ульман Д. Структуры данных и алгоритмы. – М.: Вильямс, 2000.

2. Вандервурд Д., Джосаттис Н. Шаблоны C++: справочник разработчика. – М.: Издательский дом “Вильямс”, 2003. – 544 с.

3. https://www.coursera.org/learn/analysis-of-algorithms

4. https://www.coursera.org/learn/algorithms-on-graphs/programming/nSER4/advanced-shortest-paths