УКРАЇНСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ НАУКИ І ТЕХНОЛОГІЙ

«ЗАТВЕРДЖУЮ»
Голова приймальної комісії,
ректор УДУНД, професор

_____Костянтин СУХИЙ
«_28_» ___05___ 2026 року

ПРОГРАМА

фахового вступного випробування
для здобуття ступеня магістра
за освітньо-професійною програмою « Комп'ютерні науки »
спеціальності F3 « Комп'ютерні науки »
(для пільгових категорій вступників на основі ЄФВВ)

Дніпро – 2026

РОЗРОБЛЕНО ТА ВНЕСЕНО:
ННІ «Придніпровська державна академія будівництва та архітектури»
(повне найменування закладу вищої освіти)

РОЗРОБНИКИ ПРОГРАМИ:
Пономарьова О.А., к.т.н., доцент, завідувачка кафедри комп'ютерних наук, інформаційних технологій та прикладної математики;
Вельмагіна Н.О., к.ф.-м. н., доцент, доцент кафедри комп'ютерних наук, інформаційних технологій та прикладної математики;
Шибко О.М., к.т.н., доцент, доцент кафедри комп'ютерних наук, інформаційних технологій та прикладної математики.

Програму схвалено на засіданні кафедри

Протокол від «__11__» __травня__ 2026 року № __11__

Завідувач кафедри _____ Олена ПОНОМАРЬОВА
(підпис) (Ім'я, ПРІЗВИЩЕ)

«__11__»__травня___2026 року

Програма погоджена групою забезпечення якості освітньо-професійної програми «Комп'ютерні науки» підготовки за другим (магістерським) рівнем вищої освіти зі спеціальності F3 « Комп'ютерні науки ».

Протокол від «__12__» __травня__ 2026 року № __5__

Гарант освітньої програми _____ /Ірина ДМИТРІЄВА
(підпис) (Ім'я, ПРІЗВИЩЕ)

«_27_»__травня__2026 року

1. МЕТА ТА ЗАДАЧІ ВСТУПНОГО ВИПРОБУВАННЯ

1.1. Метою вступного випробування є оцінювання рівня теоретичної та практичної підготовки вступників, які мають перший (бакалаврський) рівень вищої освіти, їхньої здатності опановувати поглиблені знання, а також визначення їхньої професійної відповідності та готовності до успішного розв'язання складних теоретичних і прикладних задач дослідницького та/або інноваційного характеру у сфері комп'ютерних наук.

1.2. Основними задачами вступного випробування є комплексна перевірка володіння математичними основами комп'ютерних наук, базовими концепціями алгоритмізації, моделювання та розроблення програмного забезпечення; визначення здатності вступника аналізувати й проєктувати складні системи, адмініструвати бази даних, а також застосовувати сучасні інформаційні технології для розв'язання прикладних і дослідницьких задач; оцінювання навичок аналізу великих обсягів даних і встановлення загальної готовності абітурієнта до самостійної науково-дослідницької та інноваційної діяльності в галузі.

1.3. Згідно з вимогами освітньо-професійної програми абітурієнти повинні:

знати:

- Теоретичні та математичні засади комп'ютерних наук;
- Сучасні теорії, концепції та принципи створення, реалізації та дослідження комп'ютерних систем (зокрема обчислювальних, інтелектуальних, розподілених і масштабованих);
- Моделі, методи та алгоритми комп'ютерних обчислень, способи подання, опрацювання та інтелектуального аналізу даних;
- Принципи формалізації предметної області проєктів та побудови інформаційних і концептуальних моделей;
- Методики розроблення, тестування, адміністрування та супроводу баз даних, знань і програмного забезпечення загалом;
- Міжнародні стандарти та моделі оцінювання якості програмних продуктів та інформаційних систем.

вміти:

- Застосовувати математичні методи й моделі для аналізу, дослідження та

оптимізації архітектурних рішень і формалізованих процесів;

- Розробляти концептуальні та інформаційні моделі предметної області відповідно до вимог замовника;
- Створювати, аналізувати й оптимізувати алгоритми розв'язування прикладних та теоретичних задач комп'ютерних наук, оцінювати їх ефективність;
- Розробляти програмне забезпечення з урахуванням наявних обмежень, інструментів командної роботи, вимог до архітектури та ресурсів;
- Збирати, формалізувати, структурувати, систематизувати та аналізувати дані (включно з великими обсягами даних) для прийняття якісних проєктних рішень;
- Проєктувати, супроводжувати та адмініструвати сучасні бази даних і знань;
- Виявляти й усувати проблемні ситуації, що виникають під час експлуатації програмного забезпечення, формулювати вимоги щодо його реінжинірингу чи модифікації.

2. ПЕРЕЛІК ДИСЦИПЛІН ТА ПИТАНЬ ДО ВСТУПНОГО ВИПРОБУВАННЯ

2.1 Об'єктно-орієнтоване програмування

- Основні концепції об'єктно-орієнтованої парадигми: поняття класу, об'єкта, стану та поведінки.
- Фундаментальні принципи ООП: інкапсуляція, успадкування, поліморфізм та абстракція (їх сутність та практичне призначення).
- Конструктори та деструктори: призначення, види конструкторів (за замовчуванням, з параметрами, копіювання), особливості їх виклику в ієрархіях класів.
- Керування доступом до елементів класу: модифікатори доступу (private, protected, public) та забезпечення інкапсуляції даних.
- Статичні члени та методи класу (модифікатор static): особливості використання та розподілу пам'яті.
- Механізм успадкування: одиночне та множинне успадкування, проблема «ромбоподібного» успадкування та методи її вирішення.
- Поліморфізм та динамічне зв'язування: віртуальні функції, таблиця віртуальних

методів (VMT).

- Абстрактні класи та інтерфейси: відмінності між ними, призначення чисто віртуальних функцій.
- Обробка виняткових ситуацій (Exception Handling): оператори try, catch, throw, створення власних класів винятків.
- Перевантаження операторів та функцій: правила, обмеження та особливості реалізації у сучасних мовах програмування.
- Шаблони функцій та шаблони класів (узагальнене програмування): концепція generic-типів та параметризованих класів.
- Основи керування пам'яттю в ООП: життєвий цикл об'єкта, динамічне виділення пам'яті, автоматичне збирання сміття (Garbage Collection) та «розумні» вказівники.
- Принципи проєктування SOLID: короткий зміст кожного принципу (S, O, L, I, D) та їх роль у створенні гнучкого програмного забезпечення.

2.2 Проєктування інформаційних систем

- Поняття інформаційної системи (ІС), її структура, класифікація та архітектурні складові.
- Життєвий цикл інформаційних систем: основні етапи, та процеси.
- Моделі життєвого циклу ІС: каскадна (водоспадна), ітераційна, інкрементна, спіральна (їхні переваги та недоліки).
- Сучасні гнучкі методології розроблення та проєктування ІС: принципи Agile, особливості фреймворків Scrum та Kanban.
- Передпроєктне обстеження предметної області: збір, аналіз, структурування та формалізація вимог до ІС (функціональні та нефункціональні вимоги).
- CASE-технології та їх роль в автоматизації процесів аналізу, проєктування та супроводу інформаційних систем.
- Моделювання бізнес-процесів: методології структурного аналізу та проєктування (SADT, нотації IDEF0, IDEF3, DFD, ERD).
- Об'єктно-орієнтований аналіз та проєктування з використанням мови UML: призначення, класифікація та структура діаграм.
- Проєктування функціональної структури ІС за допомогою UML: діаграми

прецедентів, взаємодії та станів.

- Проєктування статичної структури ІС за допомогою UML: діаграми класів, об'єктів та компонентів.
- Патерни проєктування в архітектурі ІС: архітектурні шаблони, креаційні, структурні та поведінкові патерни.
- Процеси тестування, верифікації та валідації ІС. Стратегії інтеграції компонентів системи та розгортання.
- Оцінювання якості, надійності, ефективності та безпеки проєктних рішень при створенні інформаційних систем.

2.3 Вебтехнології та вебдизайн

- Мова розмітки HTML5: семантична структура вебсторінки, нові елементи, робота з формами, мультимедійні можливості та API.
- Технологія каскадних таблиць стилів CSS3: селектори, пріоритетність стилів, блокова модель, позиціонування елементів.
- Клієнтське програмування на JavaScript: базовий синтаксис, об'єктна модель документа та об'єктна модель браузера, обробка подій.
- Асинхронна взаємодія у вебдодатках: технологія AJAX, використання Fetch API та обіцянок, концепція REST API, формати даних JSON та XML.
- Сучасні клієнтські фреймворки та бібліотеки (React, Angular, Vue.js): компонентний підхід, віртуальний DOM, керування станом програми.
- Серверні вебтехнології: принципи роботи серверних мов (Node.js, PHP, Python, Java), взаємодія з базами даних, збереження стану сесій.
- Вебдизайн та UX/UI: принципи композиції, типографіка, колористика, модульні сітки, юзабіліті та забезпечення доступності інтерфейсів.
- Системи керування вмістом: архітектура, призначення та принципи функціонування.
- Оптимізація та безпека вебдодатків: оптимізація швидкості завантаження сторінок, основні вразливості та методи захисту від них.

2.4 Інтелектуальний аналіз даних

- Концепція інтелектуального аналізу даних: визначення, передумови виникнення,

відмінності від класичного статистичного аналізу та OLAP-систем.

- Основні етапи процесу виявлення знань у базах.
- Попередня обробка даних: очищення від шумів та аномалій, інтеграція даних, заповнення пропусків, нормалізація та дискретизація характеристик.
- Задачі зменшення розмірності даних: метод головних компонент, відбір інформативних ознак.
- Задача класифікації: суть, математична постановка та сфери застосування.
- Алгоритми класифікації.
- Логістична регресія та метод опорних векторів у задачах бінарної та багатокласової класифікації.
- Задача кластеризації: критерії поділу на кластери, метрики відстані (евклідова, мангеттенська, косинусна).
- Алгоритми кластеризації.
- Пошук асоціативних правил.
- Основи інтелектуального аналізу текстової інформації та вебданих: токенізація, лематизація, модель мішка слів та метрика TF-IDF.
- Концепція Великих даних: технологічні особливості опрацювання великих масивів даних, парадигма MapReduce та стек технологій Apache Hadoop / Apache Spark.

2.5 Технологія створення програмних продуктів

- Поняття програмного продукту, його життєвий цикл та критерії якості згідно з міжнародними стандартами.
- Класичні моделі розроблення програмного забезпечення: каскадна, V-модель, інкрементна та ітераційна моделі (порівняльний аналіз, переваги та обмеження).
- Еволюційна та спіральна моделі розроблення ПЗ, управління ризиками під час їх застосування.
- Сучасні методології розроблення програмних продуктів: маніфест Agile, філософія та базові принципи гнучкої розробки.
- Фреймворк Scrum: ролі в команді (Product Owner, Scrum Master, Developers), артефакти (Product Backlog, Sprint Backlog, Increment) та основні події (планування, щоденний Scrum, огляд спринту, ретроспектива).