Міністерство освіти і науки України

Харківський національний університет імені В.Н. Каразіна

Кафедра моделювання систем і технологій

**«ЗАТВЕРДЖУЮ»**

Проректор з науково-педагогічної роботи

_____ Олександр ГОЛОВКО

“ ” _____ 2022 р.

РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

# «Мови прикладного програмування»

| | |
|---|---|
| рівень вищої освіти | перший (бакалаврський) |
| галузь знань | 12 Інформційні технології |
| спеціальність | 122 Комп’ютерні науки |
| освітня програма | Комп’ютерні науки |
| спеціалізація | |
| вид дисципліни | обов’язкова |
| факультет | Комп’ютерних наук |

2022 / 2023 навчальний рік

Програму рекомендовано до затвердження Вченою радою факультету комп'ютерних наук

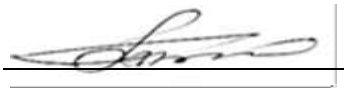
« 29 » серпня 2022 року, протокол № 14

РОЗРОБНИК ПРОГРАМИ: старший викладач кафедри моделювання систем і технологій **Паршенцев Богдан Володимирович**

Програму схвалено на засіданні кафедри моделювання систем і технологій

Протокол від « 29 » серпня 2022 року № 11

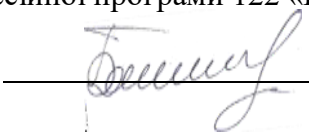
Завідувач кафедри моделювання систем і технологій



Микола ТКАЧУК

Програму погоджено з гарантом освітньо-професійної програми 122 «Комп'ютерні науки»

Гарант освітньо-професійної програми 122 «Комп'ютерні науки»

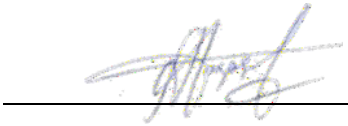


Сергій БОГУЧАРСЬКИЙ

Програму погоджено науково-методичною комісією факультету комп'ютерних наук

Протокол від « 29 » серпня 2022 року № 1

Голова науково-методичної комісії факультету комп'ютерних наук



Анатолій БЕРДНІКОВ

# ВСТУП

Програма навчальної дисципліни «Мови прикладного програмування» складена відповідно до освітньо-професійної програми підготовки першого (бакалаврського) рівня спеціальності 122 Комп'ютерні науки.

## 1. Опис навчальної дисципліни

1.1. Мета викладання навчальної дисципліни

Метою викладання навчальної дисципліни «Мови прикладного програмування» є підготовка кваліфікованих фахівців у галузі інформаційних технологій.

1.2. Основні завдання вивчення дисципліни

Основними завданнями вивчення дисципліни «Мови прикладного програмування» є забезпечити відповідні сучасним вимогам знання студентів як у теоретичних засновках RubyMine, Framework Rails, Ruby так і забезпечити практичні знання та уміння по використанню Ruby , Rails Framework і основним принципам їхнього застосування при розробці web застосунків і сайтах World Wide Web.

В ході вивчення дисципліни у студента повинні формуватися наступні компетентності.

*Загальні компетентності (ЗК):*

- здатність до пошуку, оброблення та аналізу інформації з різних джерел (ЗК7);
- здатність генерувати нові ідеї (креативність) (ЗК8);
- здатність приймати обґрунтовані рішення (ЗК9);
- здатність бути критичним і самокритичним (ЗК11);
- здатність оцінювати та забезпечувати якість виконуваних робіт (ЗК12);
- здатність зберігати та примножувати моральні, культурні, наукові цінності і досягнення суспільства на основі розуміння історії та закономірностей розвитку предметної області, її місця у загальній системі знань про природу і суспільство та у розвитку суспільства, техніки і технологій, використовувати різні види та форми рухової активності для активного відпочинку та ведення здорового способу життя (ЗК15).

*Спеціальні (фахові,предметні) компетентності (ФК):*

- здатність до виявлення статистичних закономірностей недетермінованих явищ, застосування методів обчислювального інтелекту, зокрема статистичної, нейромережевої та нечіткої обробки даних, методів машинного навчання та генетичного програмування тощо (ФК2);

- здатність здійснювати формалізований опис задач дослідження операцій в організаційно-технічних і соціально-економічних системах різного призначення, визначати їх оптимальні розв'язки, будувати моделі оптимального управління з урахуванням змін економічної ситуації, оптимізувати процеси управління в системах різного призначення та рівня ієрархії (ФК5);

- здатність до розробки мережевого програмного забезпечення, що функціонує на основі різних топологій структурованих кабельних систем, використовує комп'ютерні системи і мережі передачі даних та аналізує якість роботи комп'ютерних мереж (ФК13);

1.3. Кількість кредитів - 5

1.4. Загальна кількість годин – 150 годин.

| 1.5. Характеристика навчальної дисципліни | |
|---|---|
| Обов'язкова / <u>за вибором</u> | |
| Денна форма навчання | Денна форма навчання |
| Рік підготовки | |
| 3-й | 3-й |
| Семестр | |
| 5-й | 6-й |
| Лекції | |
| год. | 48 год. |
| Практичні, семінарські заняття | |
| год. | год. |
| Лабораторні заняття | |
| год. | 48 год. |
| Самостійна робота | |
| год. | 54 год. |
| у тому числі індивідуальні завдання | |
| 40 год. | |

1.6. Заплановані результати навчання

Згідно з вимогами освітньо-професійної програми студенти повинні досягти таких результатів навчання:

***знати:***

- теоретичні засновки Ruby, Rails, Framework;
- основи побудови ruby скриптів та прийоми, які використовуються для побудови веб застосунків;
- принципи проектування сайтів; знати особливості фреймворка Rails.

***вміти*:**

- проектувати веб застосунки за MVC архітектурою , з використанням Rails
- складати програми та знаходити помилки в Ruby-коді;
- використовувати мову Ruby для розробки скриптів;
- використовувати мову Ruby та бібліотеку ActiveRecord для розробки скриптів для запитів до бази даних.

В Результаті вивчення дисципліни у студента повинні формуватися наступні програмні результати навчання (ПРН):

- застосовувати знання основних форм і законів абстрактно-логічного мислення, основ методології наукового пізнання, форм і методів вилучення, аналізу, обробки та синтезу інформації в предметній області комп'ютерних наук (**ПРН1)**;
- використовувати знання закономірностей випадкових явищ, їх властивостей та операцій над ними, моделей випадкових процесів та сучасних програмних середовищ для розв'язування задач статистичної обробки даних і побудови прогнозних моделей (**ПРН3)**;
- проектувати, розробляти та аналізувати алгоритми розв'язання обчислювальних та логічних задач, оцінювати ефективність та складність алгоритмів на основі застосування формальних моделей алгоритмів та обчислюваних функцій **(ПРН5)**;

- використовувати методи чисельного диференціювання та інтегрування функцій, розв'язання звичайних диференціальних та інтегральних рівнянь, особливостей чисельних методів та можливостей їх адаптації до інженерних задач, мати навички програмної реалізації чисельних методів **(ПРН6)**;
- розуміти принципи моделювання організаційно-технічних систем і операцій; використовувати методи дослідження операцій, розв'язання одно- та багатокритеріальних оптимізаційних задач лінійного, цілочисельного, нелінійного, стохастичного програмування **(ПРН7);**
- володіти мовами системного програмування та методами розробки програм, що взаємодіють з компонентами комп'ютерних систем, знати мережні технології, архітектури комп'ютерних мереж, мати практичні навички технології адміністрування комп'ютерних мереж та їх програмного забезпечення (**ПРН14)**;
- розуміти концепцію інформаційної безпеки, принципи безпечного проектування програмного забезпечення, забезпечувати безпеку комп'ютерних мереж в умовах неповноти та невизначеності вихідних даних (**ПРН16)**;
- виконувати паралельні та розподілені обчислення, застосовувати чисельні методи та алгоритми для паралельних структур, мови паралельного програмування при розробці та експлуатації паралельного та розподіленого програмного забезпечення (**ПРН17)**.

## 2. Тематичний план навчальної дисципліни
## Розділ 1

**Тема 1. Rubymine, Framework Rails.**

Бренд Ruby - це динамічна мова програмування, що часто використовується для веб-розробки та скриптування. Середовище розробки: Ruby підтримує багато середовищ розробки, таких як RubyMine, Atom, Sublime Text і т. д.Каркас Ruby on Rails - це відкрите середовище розробки веб-додатків, яке базується на мові програмування Ruby. Бібліотека класів Ruby - це статичний компонент каркаса, який включає в себе велику кількість класів та методів, що використовуються в розробці програм. Загальномовне виконавче середовище Ruby - це динамічний компонент каркаса, який забезпечує виконання коду Ruby. Він забезпечує збірку сміття та інші функції, що допомагають в управлінні пам'яттю.

Керований код - Ruby є мовою програмування з керованим кодом, що означає, що вона дозволяє розробникам працювати на вищому рівні абстракції, не хвилюючись про деталі роботи з пам'яттю та інші низькорівневі завдання.Загальномовне специфікації - в Ruby є загальномовне специфікації, які включають Ruby Language Specification та Ruby Standard Library Specification. Ці специфікації допомагають забезпечити сумісність та зручність використання модулів у програмі.

**Тема 1.1 Мова Ruby і перші проекти.**

Створення мови. Його особливості. Рішення, проекти, простору імен. Консольні і web застосунки створені за допомогою Ruby, побудовані за замовчуванням.

**Тема 2. Особливості динамічних мов програмування.**

Загальний погляд. Система типів. Типи-значення і посилальні типи. Вбудовані типи. Перетворення змінних в об'єкти і vice versa. Операції "упакувати" і "розпакувати". Перетворення типів. Перетворення всередині арифметичного типу. Перетворення строкового типу. Перевіряються перетворення.

**Тема 2.1 Перетворення типів.**

Перетворення типів. Типи-значення і посилальні типи. Особлвості перетворення типів.

**Тема 2.2 Змінні і вираження.**

Оголошення змінних. Синтаксис оголошення. Ініціалізація. Час життя і область видимості.

Де оголошуються змінні? Локальні і глобальні змінні. Чи є глобальні змінні в Ruby? Константи.

**Тема 2.3 Вирази. Операції у вираженнях.**

Побудова виразів. Операції і їх пріоритети. Опис операцій.

**Тема 2.4 Присвоєння і вбудовані функції.**

Присвоєння. Класи Math, Random і вбудовані функції.

**Тема 2.5 Оператори мови Ruby.**

Оператори мови Ruby. Оператор присвоювання. Складовою оператор. Порожній оператор. Оператори вибору. If-оператор. Switch-оператор. Оператори переходу. Оператор send. Оператори break, continue. Оператори циклу. For-оператор. Цикли while. Цикл foreach. Ітератор map. Ітератор each.

**Тема 3. Процедури і функції - методи класу.**

Процедури і функції - дві форми функціонального модуля. Чим відрізняються ці форми? Процедури і функції - це методи класу. Опис методів (процедур і функцій). Синтаксис. Атрибути доступу. Статичні і динамічні методи. Формальні аргументи. Статус аргументів. Тіло методів. Виклик процедур і функцій. Фактичні аргументи. Семантика виклику. Поля класу або аргументи методу? Поля класу або функції без аргументів? Проектування класу Account. Функції з побічним ефектом. Перевантаження методів.

**Тема 3.1. Масиви мови Ruby.**

Загальний погляд на масиви. Порівняння з масивами C ++. Чому масиви Ruby краще, ніж масиви C ++. Види масивів - одномірні, багатовимірні. Динамічні масиви.

**Тема 3.2. Клас Array і нові можливості масивів.**

Сімейство класів-масивів. Батьківський клас Array і успадковані їм інтерфейси. Нові можливості масивів в Ruby. Як коректно працювати з масивами об'єктів?

**Тема 3.3 Хеші в Ruby**.

Хеші в Ruby та їх особливості.

**Тема 3.4. Символи Ruby. Класи Symbol.**

Символи Ruby. Клас Symbol. Особливості використання символів та відмінність від класа String

**Тема 3.5. Регулярні вираження.**

Регулярні вираження. Простір RegularExpressions і його класи. Регулярні вираження і мови. Теорія регулярних виразів. Практика застосування регулярних виразів. Розбір текстів і пошук за зразком. Властивості і методи класу Regex і інших класів, пов'язаних з регулярними вираженнями. Приклади застосування регулярних виразів.

**Тема 4. Класи. Дві ролі класу в ООП.**

Синтаксис опису класу. Поля і методи класу. Конструктори. Статичні поля і методи. Статичні конструктори. Поля тільки для читання. Закриті поля. Стратегії доступу до полів класу. Процедури властивості. Індексатори. Приклади.

**Тема 4.1 Структури.**

Структури - реалізація розгорнутих класів. Синтаксис структур. Порівняння структур та відкритих структур . Вбудовані структури. Struct/OpenStruct- окремий випадок класу. Особливості структур. Приклади.

**Тема 4.2. Відносини між класами.**

Клієнти і спадкоємці Класи. Відносини між класами. Ставлення клієнти - постачальники. Ставлення успадкування. Одиничне успадкування. Батьки і спадкоємці. Предки і нащадки. Що успадковують нащадки. Що можуть змінити нащадки. Одностороннє присвоювання. Контроль

типів і зв'язування - статична і динамічна. Поліморфізм. Проектування класів. Абстрактні класи. Класи поведінки.

**Тема 4.3. Модулі. Множинне спадкування.**

Модулі як окремий випадок класу. Множинне спадкування. Проблеми. Множинне включення модулів. Вбудовані модулі. Різниця між include та extend. Поверхневе і глибоке клонування і сериализация. Збереження і обмін даними.

**Тема 5. Функціональний тип в Ruby. Блоки, Proc та Lambda.**

Нове слово для старого поняття. Функціональний тип. Функції вищих порядків. Обчислення інтеграла і сортування. Два способи взаємодії частин при побудові складних систем. Блоки. Функції зворотного виклику. Спадкування і функціональні типи. Порівняння двох підходів. Модуль Proc та Lambda. Методи і властивості класу. Операції над блоками, Proc та Lambda. Комбінування блоків, Proc та Lambda. Масив викликів.

**Тема 5.1. Метапрограмування.**

Загальний погляд. Динамічне створення класів та методів. Як працює monkey patching. Динамічне відериття класів та об’єктів.

**Тема 6. Універсальність. Класи та Модулі.**

Спадкування і додавання міксін - взаємно доповнюють базові механізми створення сімейства класів. Родові параметри універсального класу. Синтаксис міксін. Обмеження міксін. Методи з родовими параметрами. Обмежена універсальність - обмеження, що накладаються на родові параметри. Види обмежень. Окремі випадки класів: структури.

**Тема 6.1. Налагодження і обробка виняткових ситуацій.**

Коректність і стійкість. Специфікація системи. Коректність і стійкість програмних систем. Виняткові ситуації. Обробка виняткових ситуацій. Життєвий цикл програмної системи. Три закони программотехніки. Налагодження. Створення надійного коду. Мистецтво налагодження. Налагодження і інструментальне середовище

.

**Тема 7. Введення в ActiveRecord. Міграції даних.**

Основи роботи з ActiveModel. Додавання моделей та підключення до баз даних. Додававання міграцій та особливості використовування кастомних міграцій

**Тема 8. Робота з асоціацями та SQL.**

Створення асоціацій між моделями Основні типи асоціацій в ActiveModel (has_one, belongs_to, has_many, has_and_belongs_to_many). Можливості використовування SQL у бібліотеці ActiveRecord

**Тема 9. Робота з ActiveRecord (продовження).**

Додавання валідацій, фільтрів та скоупов. Організація моделей та додавання кастомних методів для вирішення бізнес задач. Вивчення базових інтерфейсів для роботи з ActiveRecord. Розуміння колбеків.

**Розділ 2**

**Тема 1. Принципи роботи і структура Web-додатків на основі Rails**.

Розглядається архітектура сучасних Web-додатків, взаємодія клієнтської і серверної частин таких програм, принципи їх організації в середовищі Rails

**Тема 1.1. Основи роботи в RubyMine**

Розглядаються можливості інтегрованого середовища розробки RubyMine для створення додатків Ruby on Rails і налаштування її інтерфейсу для потреб конкретного користувача. Розглядається процес створення нового Web-додатки, способи навігації по його структурі, перегляду і редагування інформаційної частини в режимі роботи з вихідним кодом і дизайну, використання різних елементів управління на сторінках, а також процедур - обробників подій цих елементів.

**Тема 2. Основи Web-програмування з використанням Ruby on Rails**

Розглядаються питання формування відгуку сервера і структура HTML-файлу, який отримуємо при цьому, а також питання виведення інформації у вікні браузера клієнта і особливості цього процесу. Наводиться ряд прикладів, які демонструють можливості організації введення клієнтом інформації та передачі її на сервер для подальшої обробки Web-додатком. Розглядаються типові помилки, що виникають в процесі роботи з вихідним кодом Web-додатки, шляхи їх виявлення та виправлення. Вивчаються можливості динамічного створення елементів управління в Web-додатку і додавання їх на сторінку, створення обробників подій для них. Вивчається структура Web-додатки і таких її складових, як клас Model, Controller. Розглядаються принципи роботи з файлами cookies.

**Тема 3. Принципи розробки призначеного для користувача інтерфейсу інтернет-додатки.**

Розглядаються питання створення призначеного для користувача інтерфейсу інтернет-додатки. Розглядаються принципи позиціонування елементів призначеного для користувача інтерфейсу, перераховуються основні інтерфейсні елементи, які використовуються в Rails. Розглядаються принципи використання основних інтерфейсних елементів для виведення інформації на Web-сторінку, а також організації введення даних. Описуються можливості застосування каскадних таблиць стилів для оформлення зовнішнього вигляду елемента управління і всієї сторінки. Наводяться приклади динамічного керування вмістом сторінки з використанням елементів програмування, розглядаються найпростіші приклади здійснення прив'язки елементів управління до даних. Розглядаються базові принципи організації перевірки введених даних.

**Тема 4. Використання Master Page і навігація при побудові інтернет-додатків. Роутінг**

Розглядаються принципи використання майстер-сторінок при розробці Інтернет-додатків. Наводяться приклади побудови типових майстер-сторінок і сторінок вмісту. Розглядаються можливості організації перемикання між різними майстер-сторінками та програмного звернення до вмісту майстер-сторінок зі сторінок вмісту. Наводяться приклади використання вкладених майстер-сторінок.

**Тема 4.1 Навігація по Web-додаткам.**

Розглядаються питання організації навігації по сторінках Web-додатки, розбираються питання використання існуючих в Rails елементів управління, призначених для організації ефективних механізмів навігації по Web-додаткам. Розглядаються можливості організації покрокових процесів збору інформації подібно до традиційних майстрам.

**Тема 5. Використання шаблонів при оформленні Web-додатки**

Розглядаються питання стандартизації оформлення сторінок, включених в інтернет-додаток, за допомогою механізму шаблонів, підтримуваного Rails, наводяться приклади реалізації тем додатків для різних його елементів і сторінок. Порушуються питання динамічної зміни теми програми.

**Тема 6. Використання кешування в Web-додатках.**

Розглядаються питання використання різних видів кешування, підтримуваних RoR. Наводяться приклади застосування кешування на сторінках Rails додатків.

## 3. Структура навчальної дисципліни

| Назви розділів і тем | Кількість годин | | | | | |
|---|---|---|---|---|---|---|
| | денна форма | | | | | |
| | усього | у тому числі | | | | |
| | | л | п | лаб. | інд. | с. р. |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| **Розділ 1.** | | | | | | |
| Тема 1. Rubymine, Framework Rails. | 2 | 2 | | | | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Тема 2. Особливості динамічних мов програмування. | 8,25 | 4 | | 4 | | 0,25 |
| Тема 3. Процедури і функції - методи класу. | 9 | 4 | | 4 | | 1 |
| Тема 4. Класи. Дві ролі класу в ООП. | 6,5 | 2 | | 4 | | 0,5 |
| Тема 5. Функціональний тип в Ruby. Блоки, Proc та Lambda. | 4,25 | 2 | | 2 | | 0,25 |
| Тема 6. Універсальність. Класи та Модулі. | 4,25 | 2 | | 2 | | 0,25 |
| Тема 7. Введення в ActiveRecord. Міграції даних. Підготовка до контрольної роботи. | 11 | 2 | | 4 | | 5 |
| Тема 8. Робота з асоціацями та SQL. | 4,5 | 2 | | 2 | | 0,5 |
| Тема 9. Робота з ActiveRecord (продовження). | 6,25 | 4 | | 2 | | 0,25 |
| **Разом за розділом 1** | **56** | **24** | | **24** | | **8** |
| **Розділ 2.** | | | | | | |
| Тема 1. Принципи роботи і структура Web-додатків на основі Rails. | 8 | 4 | | 4 | | |
| Тема 2. Основи роботи в RubyMine | 8,25 | 4 | | 4 | | 0,25 |
| Тема 3. Основи Web-програмування з використанням Ruby on Rails | 8,25 | 4 | | 4 | | 0,25 |
| Тема 4. Використання Master Page і навігація при побудові інтернет-додатків. Роутінг. | 8,25 | 4 | | 4 | | 0,25 |
| Тема 5. Використання шаблонів при оформленні Web-додатки | 13 | 4 | | 4 | | 5 |
| Тема 6. Використання кешування в Web-додатках. | 8,25 | 4 | | 4 | | 0,25 |
| Індивідуальне науково-дослідне завдання | 40 | | | | 40 | |
| **Разом за розділом 2** | **94** | **24** | | **24** | **40** | **6** |
| **Усього годин** | **150** | **48** | | **48** | **40** | **14** |

## 4. Теми семінарських (практичних, лабораторних) занять

| | | |
|---|---|---|
| | | **Розділ 1** |
| № з/п | Назва теми | Кількість годин |
| 1 | Перетворення типів. Змінні і вираження. | 2 |
| 2 | Вирази. Операції у вираженнях. Оператори мови Ruby. | 2 |
| 3 | Процедури і функції. | 2 |
| 4 | Масиви мови Ruby. Клас Array і нові можливості масивів. | 2 |
| 5 | Регулярні вираження. | 2 |
| 6 | Структури і відкритті сруктури. | 2 |
| 7 | Міксіни. | 2 |
| 8 | Метапрограмування | 2 |
| 9 | Обробка виняткових ситуацій. | 2 |
| 10 | Робота з Acriverecord та SQL. | 2 |
| 11 | Міграції. | 2 |
| 12 | Організація асоціацій. | 2 |
| **Разом за розділом 1** | | **24** |
| **Розділ 2** | | |
| 13 | Основи Web-програмування з використанням Rails | 6 |

| | | |
|---|---|---|
| 14 | Принципи розробки призначеного для користувача інтерфейсу інтернет-додатки . | 6 |
| 15 | Використання патерну MVC при побудові інтернет-додатків. | 6 |
| 16 | Використання тем при оформленні Web-додатків. | 6 |
| **Разом за розділом 2** | | **24** |
| **Усього годин** | | **48** |

**5. Завдання для самостійної роботи**

| № з/п | Види, зміст самостійної роботи | Кількість годин |
|---|---|---|
| 1 | Завдання на самостійну роботу 1<br><br>**«Інтеграція Ruby з базами даних»**<br>**Мета роботи**:<br><br>Вивчити основні елементи ActiveRecord, властивості міграцій та елементи маніпуляції з даними, а також отримати практичні навички в розробці ruby додатків збазами даних.<br><br>1. Вивчити теоретичний матеріал.<br>2.Створити базу даних.<br>3. Створити з'єднання додатка та бази даних.<br>4. Вивчити функціонал seed та "засіяти" базу даних.<br>5.Додати моделі та базові валідації.<br>6. Протестувати програму. | 1,5 |
| 2 | Завдання на самостійну роботу 2<br>**«Інтеграція тестів в Rails-додатки»**<br>**Мета роботи:**<br>Освоїти технології тестувваннята виконанні тестів для забезпечення коректності та стабільності роботи веб-додатків, написаних з використанням Ruby on Rails..<br>1. Вивчити теоретичний матеріал.<br>2. Встановлення необхідних залежностей: для написання та виконання тестів у Ruby on Rails необхідно встановити деякі залежності, такі як фреймворк RSpec або гем Capybara.<br>3. Написання тестів: після встановлення необхідних залежностей можна приступити до написання тестів. В Rails тести можуть бути написані у форматі RSpec-файлів або використовуючи вбудований фреймворк для тестування.<br>4. Запуск тестів: після написання тестів їх потрібно запустити для перевірки коректності та стабільності роботи веб-додатку. Для запуску тестів можна використовувати спеціальні команди у консолі.<br>5. Продемонструвати роботу програми викладачеві.<br>5. Підготовка до контрольної роботи. | 15 |
| 3 | Завдання на самостійну роботу 3<br>**«Розгортання Rails додатка за допомогою Heroku»**<br>**Мета роботи:**<br>Освоїти технологію розгортання Rails додатка за допомогою Heroku.<br>1. Реєстрація на Heroku: для початку необхідно зареєструватися на Heroku та створити новий додаток.<br>2. Інсталювання Heroku CLI: Heroku CLI (Command Line Interface) - це інструмент для взаємодії з Heroku з терміналу. Для роботи з Heroku потрібно його інсталювати.<br>3. Налаштування змінних оточення: у додатку можуть бути змінні оточення, які відрізняються в залежності від середовища, на якому запущений додаток. | 1,5 |

| | | |
|---|---|---|
| | Наприклад, середовище розробки може використовувати іншу базу даних, ніж продуктивне середовище. Тому необхідно налаштувати змінні оточення для додатку на Heroku.<br>4. Продемонструвати процес розгортання додатка викладачеві. | |
| **Разом за розділом 1** | | **18** |
| | **Розділ 2** | |
| 1 | Зробити сторінку, яка буде розділена вертикально на дві частини. У лівій частині знаходяться форма введення тексту<br>у правій — текст «0 запитів» та історія запитів<br>При натисканні кнопки в лівій частині відбувається передача на сервер тексту, введеного в поле введення (без перезавантаження сторінки). Потім поле введення очищується. На сервері створюється новий екземпляр моделі із цим текстом. У відповідь від сервера приходить час створення цього екземпляра моделі, яка (у будь-якому форматі) додається до div у правій частині сторінки. Також оновлюється текст зверху діва «N запитів», де N — кількість запитів по кнопці, яке обробив сервер до цього моменту (де їх зберігати чи обчислювати — подумати самостійно). У цьому слово «запит» має правильно схилятися залежно від числа N. div у правій частині, в який додається час створення чергового запису — не повинен зростати вниз (фіксована висота), а повинна з'являтися лінійка прокручування збоку. | 0,25 |
| 2 | Створіть веб- додаток яке дозволяє вибудувати зростаючу послідовність з чисел переданих за поля введення форм. | 0,25 |
| 3 | Створіть веб- додаток, що дозволяє вибудувати спадаючу послідовність з чисел переданих за поля введення форм.<br>Створіть веб- додаток, що дозволяє обчислити середнє арифметичне, а так само знайти максимальний елемент послідовності чисел, переданих через поля введення форми. Розрахунково-графічна робота | 10 |
| 4 | Створіть веб- додаток що дозволяє обчислити суму елементів, а так само знайти мінімальний елемент послідовності чисел переданих через поля введення форми.<br>Створіть веб -додаток, що реалізує функцію зворотного зв'язку для вашого портфеля (формування і відправка mail). | 0,25 |
| 5 | Створити новий скріпт (task): Створіть нову задачу, яка буде виконувати певну дію, наприклад, очищення бази даних або відправку повідомлення електронною поштою. Запустіть цю задачу та переконайтеся, що вона виконується успішно.<br>Налаштування параметрів задачі: Додайте параметри до вашої задачі, які дозволяють налаштувати її поведінку. Наприклад, якщо задача повинна очистити базу даних, додайте параметр, який вказує на те, яку саме таблицю ви хочете очистити. Запуск задач з планувальника: Додайте ваші задачі до планувальника, такого як Cron або Whenever, щоб вони запускалися автоматично в певний час або за розкладом. Переконайтеся, що ваші задачі виконуються успішно та згідно зі зазначеним розкладом. | 0,25 |
| 6 | Створіть веб- додаток, в якому є форма реєстрації. Використовуйте реєстраційну форму з декількох полів (мінімум 4) і прапорця (checkbox) позначає згоду з правилами поведінки або ліцензійною угодою. Якщо всі поля заповнені і прапорець встановлений, то результатом роботи скрипта має бути повідомлення про успішну реєстрацію нового користувача. Підготовка до контрольної роботи | 5 |
| 7 | Індивідуальне науково-дослідне завдання | 20 |
| **Разом за розділом 2** | | **16** |
| **Усього** | | **54** |

**6. Індивідуальні завдання**

Контрольна робота –2

Розрахунково-графічна робота –2

Курсова робота

## 7. Методи навчання

Тематичні лекції, на яких дається основний систематизований матеріал курсу. Лекції представлено у вигляді презентацій Power Point на мультимедійному обладнанні. Відповіді на запитання студентів по кожному розділу теми, обговорення найбільш складних лекційних питань.

Як правило лекційні та лабораторні заняття проводяться аудиторне. А в умовах дії карантину заняття проводяться відповідно до Наказу ректора Харківського національного університету імені В.Н. Каразіна (аудиторне або дистанційно за допомогою платформ Google Meet або Zoom)

## 8. Методи контролю

Протягом навчального семестру проводиться поточний контроль знань, який складається з виконання 16 лабораторних робіт, 2 контрольні роботи, 2 розрахунково-графічних робіт та курсової роботи. Загальна сума балів, яку студент може набрати, складає – 60 балів протягом навчального семестру.

Максимальний бал за лабораторну роботу – 2 бала.
Максимальна бал за кожну контрольну роботу – 4 балів.
Максимальний бал за кожну розрахунково-графічну роботу – 2 бала.
Курсова робота – максимальний бал 16 (включається в кількість балів за семестр).
Підсумкова форма контролю – екзамен.

Екзамен проводиться у вигляді тестування по 100 запитанням. На екзамені кожен студент одержує свій індивідуальний набір питань, які формуються призволяще з бази даних усіх запитань курсу (більш 200). Успішність проходження тесту оцінюється за національною шкалою

До підсумкового контролю допускаються тільки студенти, що мають зроблені усі самостійні роботи, і які в змозі відповісти на питання викладача по їх роботам.

Результати поточного та підсумкового тестування фіксуються в базі даних сервера MySQL. На протязі тестування ведеться протокол, де студент чи викладач має змогу проаналізувати відповіді студента на запитання тесту. Протоколи також зберігаються в базі даних та можуть бути роздруковані

## 9. Схема нарахування балів

| | | | | | | |
|---|---|---|---|---|---|---|
| Розділ 1 | Розділ 2 | Контрольна робота | Індивідуальне завдання | Разом | Екзамен | Сума |
| Т1-Т9 | Т1-Т6 | | | | | |
| 18 | 12 | 10 | 20 | 60 | 40 | 100 |

При контролі навчальних здобутків контроль здійснюється по модулям курсу без розподілу по темам.

**Критерії поточної оцінки знань студентів**
**(<u>контрольна робота</u>, крок оцінювання 1 бал)**

| Кількість балів | Критерії оцінювання |
|---|---|
| 0 | Студент демонструє фрагментарні знання при незначному загальному їх обсязі (менше половини навчального матеріалу). |
| 1 | Студент знайомий з основними поняттями навчального матеріалу; може самостійно відтворити значну частину навчального матеріалу і робити певні узагальнення; вміє виконати просте навчальне завдання. |
| 2 | Студент демонструє вивчений матеріал у стандартних ситуаціях; пояснює основні процеси, що відбуваються під час роботи інформаційної системи та наводить власні приклади на підтвердження деяких тверджень; вміє виконувати навчальні завдання. |
| 3 | Студент демонструє міцні знання, самостійно визначає проміжні цілі власної навчальної діяльності, оцінює нові факти, явища; вміє самостійно знаходити додаткові відомості та використовує їх для реалізації поставлених перед ним навчальних цілей, судження його (її) логічні і достатньо обґрунтовані; має певні навички управління інформаційною системою. |
| 4 | Студент демонструє стійкі системні знання та продуктивно їх використовує; вміє вільно використовувати нові інформаційні технології для поповнення власних знань та розв'язування задач; має стійкі навички управління інформаційною системою у нестандартних ситуаціях. |

**Критерії поточної оцінки знань студентів**
**(<u>лабораторна</u> робота, крок оцінювання 0,5 бал)**

| Кількість балів | Визначення |
|---|---|
| 2 | Завдання по лабораторній роботі виконане самостійно в повному обсязі. Звіт оформлений акуратно відповідно до вимог методичних вказівок. При захисті звіту показано розуміння суті і змісту проведених досліджень |
| 1,5 | Завдання по лабораторній роботі виконане самостійно в повному обсязі. Звіт оформлений достатньо акуратно відповідно до вимог методичних вказівок. При захисті звіту були виявлені незначні помилки у знанні теоретичного матеріалу |
| 1 | Завдання по лабораторній роботі виконане в повному обсязі. Звіт оформлений достатньо акуратно, в оформленні звіту є незначні недоліки. При захисті звіту були виявлені незначні помилки у знанні теоретичного матеріалу |
| 0,5 | Завдання по лабораторній роботі виконане. Звіт оформлений з помилками і недоліками. При захисті звіту були виявлені суттєві помилки у знанні теоретичного матеріалу |
| 0 | Студент має фрагментарні знання при незначному загальному їх обсязі (менше половини навчального матеріалу) при відсутності сформованих умінь та навичок. |

**Критерії підсумкової оцінки знань студентів (<u>екзамен</u>)**

Екзамен проводиться у вигляді тестування по 100 запитанням. Максимальна кількість набраних балів 40.

Відповідь на кожне питання тесту оцінюється у 0,4 бали.

Час на відповіді – 40 хвилин.

Усі запитання, на які не було дано відповіді оцінюються у 0 балів.

**Шкала оцінювання**

| Сума балів за всі види навчальної діяльності протягом семестру | Оцінка для чотирирівневої шкали оцінювання |
|---|---|
| 90 – 100 | відмінно |
| 70-89 | добре |
| 50-69 | задовільно |
| 1-49 | незадовільно |

## 10. Рекомендована література

### Основна література


1. M. Metcalf, J. Reid., M. Cohen Modern Fortran Explained. – Oxford University Press, 2011. – 509 p.
2. S.J. Chapman Fortran 95 / 2003 for Scientists and Engineers. – 3-rd edition – McGraw-Hill, 2007. – 988 p.
3. S. Oliveira, D. E. Stewart Writing scientific software: a guide for good style – Cambridge University Press. 2006. – 316 p.
4. W. Cheney, D. Kincaid Numerical mathematics and computing – 6-th edition – Thomson Higher Education, 2008. – 789 p.
5. Shneiderman B. Designing the User Interface, 3-rd edn. – Reading, MA: Addison-Wesley, 1998.


### Допоміжна література


1. P. O. J. Scherer Computational Physics. Simulation of Classical and Quantum Systems – 2-nd edition – Springer, 2013. – 456 p.
2. T. Pang An Introduction to Computational Physics. – 2-nd edition – Cambridge University Press – 2006, – 402 p.
3. Nicholas J. Giordano Computational physics. – Prentice Hall Inc., New Jersey – 1997, – 419 p.


### Посилання на інформаційні ресурси в Інтернеті, відео-лекції, інше методичне забезпечення

http://www.c-sharpcorner.com

- Code::Blocks — The free C/C++ and Fortran IDE.
- SciDavis — SciDAVis is a free application for Scientific Data Analysis and Visualization.
- MagicPlot - Software for nonlinear fitting, plotting and data analysis.
- GFortran Manuals — GFortran documentation is included with the GCC documentation. The GNU Fortran compiler, part of GCC.
- Getting started with Fortran - The GNU Fortran Compiler
- The GNU Fortran Compiler The GNU Fortran Compiler