Міністерство освіти і науки України

Чернігівський національний технологічний університет

Навчально-науковий інститут електронних та інформаційних технологій

Кафедра кібербезпеки та математичного моделювання

“**ЗАТВЕРДЖУЮ**”

Завідувач кафедри

Ю.М. Ткач

“\_\_\_\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 р.

## РОБОЧА ПРОГРАМА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

**ОК 14 – Технології програмування**

**Освітня програма «Кібербезпека»**

**Рівень вищої освіти – перший (бакалаврський)**

Спеціальність  *125 - “Кібербезпека”*

Мова навчання: *українська*

Статус дисципліни: *обов'язкова*

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Форма навчан. | Рік навч. | Сем. | Розподіл годин | | | | | | Разом | За тиждень | | ІНДЗ | Контр. |
| Всього ауд. | Лек | Прак | Лаб. |  | СРС | Ауд. | СРС |
| 2 | 1 | 30 | 16 |  | 14 |  | 90 | 120 | 3 | 9 | РГР | Е |
| Разом | | 30 | 16 |  | 14 |  | 90 | 120 | 3 | 9 |  |  |

Чернігів – 2023 рік

Робоча програма  *Технології програмування*

(назва навчальної дисципліни)

для студентів галузі знань *12 – Інформаційні технології*спеціальності *125 - Кібербезпека*

Розробник робочої навчальної програми:

*ст. викладач кафедри кібербезпеки та математичного моделювання* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(Семендяй С.М.)*

(підпис) (прізвище та ініціали)

*асистент кафедри кібербезпеки та математичного моделювання* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ *(**Дюба І.М)*

(підпис) (прізвище та ініціали)

Робочу програму схвалено на засіданні кафедри *кібербезпеки та математичного моделювання*

Протокол від “*\_\_*” *\_\_\_\_\_\_\_\_\_* *2023* року № *\_\_*

Завідувач кафедри *кібербезпеки та математичного моделювання*

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ( Ю.М. Ткач )

(підпис) (прізвище та ініціали)

# Abstract

### 2023/2024 Sem. 1

### Course Description

The course aims to form an understanding of modern programming paradigms during software implementation of tasks. In the process of listening to the course, students will familiarize themselves with the principles of acquiring theoretical knowledge in the field of a new programming language, the application of theoretical knowledge in practice, conducting a surface analysis of the task and breaking it down into sub-tasks, the principles of choosing between existing options for solving the task, checking the work results.

The Python programming language was chosen for this course due to its ease of use and rapid deployment. Python is a universal language widely used around the world for a variety of purposes. Among the advantages of the Python language, it is possible to highlight the portability of written programs to computers of different architectures and with different operating systems, the brevity of writing algorithms, and the ability to obtain effective program code based on the speed of execution. Topics include basic language concepts, algorithmic structures, functional programming, handling exceptions, using libraries, writing your own code tests, working with files, and basic object-oriented programming concepts.

Among the main goals of this course is for students to gain practical experience in learning a new programming language, identify typical approaches to learning a new programming language, and use the experience to learn new programming languages in the future.

An important aspect of the course is gaining practical experience in reading someone else's code, which is the main job in industrial programming. Reading someone else's code allows you to develop abstract thinking, because during familiarization, an idea is formed about what exactly this or that software implementation does. Reading someone else's code allows you to highlight certain ideas for use in your own software implementations. When getting a job, young specialists are first of all suggested to familiarize themselves with the existing code for further modification or error correction.

# Опис навчальної дисципліни

|  |  |  |
| --- | --- | --- |
| **Найменування показників** | **Галузь знань, напрям підготовки, освітньо-кваліфікаційний рівень** | **Характеристика навчальної дисципліни** |
| **денна форма навчання** |
| Кількість кредитів – 4 | Галузь знань  *12 «Інформаційні технології»* | *Обов’язкова* |
|  |
| Модулів – *1* | Спеціальність:  *125 «Кібербезпека»* | **Рік підготовки:** |
| Змістових модулів – *3* | 2-й |
| Індивідуальне науково-дослідне завдання – розрахункова робота | **Семестр** |
| Загальна кількість годин – *120* | *1-й* |
| **Лекції** |
| Тижневих годин:  аудиторних – *3*;  самостійної роботи і індивідуальної студента – *9* | Освітньо-кваліфікаційний рівень:  *Бакалавр* | *16* год. |
| **Практичні, семінарські** |
| - |
| **Лабораторні** |
| 14 год. |
| **Самостійна робота** |
| 90 год. |
| **Вид контролю:** |
| Екзамен |

**Примітка**.

Співвідношення кількості годин аудиторних занять до самостійної і індивідуальної роботи становить:

для денної форми навчання – 1:3

Передумови для вивчення дисципліни відсутні.

# Мета навчальної дисципліни

Курс має на меті сформувати розуміння сучасних парадигм програмування під час програмної реалізації завдань. В процесі слухання курсу студенти ознайомляться з принципами здобуття теоретичних знань в галузі нової мови програмування, застосування теоретичних знання на практиці, проведення поверхневого аналізу завдання та розбивку на під задачі, принципами вибору між існуючими варіантами для вирішення завдання, перевірки результатів роботи.

Мова програмування Python вибрана для цього курсу завдяки своїй легкості використання та швидкому розгортанню. Python – це універсальна мова, що широко використовується в усьому світі для самих різних цілей. Серед переваг мови Python можна виділити переносимість написаних програм, на комп’ютери різної архітектури та з різними операційними системами, лаконічність запису алгоритмів, можливість отримати ефективний код програм за швидкістю виконання. Теми включають базові поняття мови, алгоритмічні структури, функціональне програмування, робота з винятковими ситуаціями, використання бібліотек, написання тестів власного коду, роботу з файлами та базові поняття об’єктно орієнтованого програмування.

Серед головних цілей цього курсу отримання практичного досвіду студентами в опанування нової мови програмування, виділення типових підходів для опанування нової мови програмування та використання досвіду для вивчення нових мов програмування в майбутньому.

Важливим аспектом курсу є набуття практичного досвіду в читанні чужого коду, що є основною роботу в промисловому програмуванні. Читання чужого коду дозволяє розвивати абстрактне мислення, оскільки під час ознайомлення формується уявлення про те, що саме виконує та чи інша програмна реалізація. Читання чужого коду дозволяє виділити певні ідеї для використання у власних програмних реалізаціях. При влаштування на роботу молодим спеціалістам в першу чергу пропонується ознайомитись з існуючим кодом для подальшої модифікації чи виправлення помилок.

КЗ 1. Здатність застосовувати знання у практичних ситуаціях

ПРН 15. Використовувати сучасне програмно-апаратне забезпечення інформаційно-комунікаційних технологій;

# Очікувані результати навчання з дисципліни

У результаті вивчення навчальної дисципліни студент повинен**:**

***знати:***

* базові елементи та конструкції мов програмування;
* основний синтаксис мов програмування python;
* базові принципи роботи з файлами;
* принципи обробки виняткових ситуацій;
* базові принципи тестування написаного коду;
* базові принципи роботи з файлами;
* базові принципи об’єктно орієнтованого підходу;

***вміти***:

* програмувати функціонально;
* представляти та обробляти інформацію;
* оцінювати правильність, структуру та стиль коду;
* вивчати нові мови програмування;
* читати документацію, робити висновки із специфікацій;
* тестувати програми , знаходити помилки та виявляти проблемні випадки;
* точно описувати симптоми помилок і чітко задавати питання.

# Критерії оцінювання результатів навчання

Критерієм успішного проходження здобувачем освіти підсумкового оцінювання є досягнення ним мінімальних порогових рівнів оцінок за кожним запланованим результатом навчання навчальної дисципліни;

Мінімальний пороговий рівень оцінки визначається як 60% від максимально можливих балів за допомогою якісних критеріїв, що описані в п. 12.

З тими студентами, які до проведення підсумкового семестрового контролю не встигли виконати всі обов’язкові види робіт та мають підсумкову оцінку до 19 балів (за шкалою оцінювання), проводяться додаткові індивідуальні заняття, за результатами яких визначається, наскільки глибоко засвоєний матеріал, та чи необхідне повторне вивчення дисципліни.

Дисципліну можна вважати такою, що засвоєна, якщо студент знає та вміє те, що зазначено в очікуваних результатах навчання.

# Засоби діагностики результатів навчання

Засобами оцінювання та методами демонстрування результатів навчання:

* екзамен;
* стандартизовані тести;
* розрахунково-графічна робота;
* захист результатів виконаних лабораторних робіт.

**Тема 0. Від ідеї до імплементації**

Все починається з ідеї. Процес перетворення команд з людської мови в інструкції для обчислювальної системи. Підготовчі дії перед початком роботи: оцінка існуючих засобів, розробка архітектури, визначення набору інструментів. Вибір мови програмування, що відповідає критеріям. Основні принципи роботи інтерпретатора. Набір інструментів для роботи: інтерпретатор, середа розробки, git клієнт.

**Тема 1. Функції та змінні**

Вітаю всесвіт, як початок вивчення мові програмування. Набір базових функцій під кожну мову програмування. Аргументи для специфікації виконання функції. Можливі помилки при використанні функцій. Необмежене використання функцій. Коментарі. Результат роботи функції. Поняття змінної. Типи даних. Рядки. Числа. Булеві значення. Написання власних функцій.

**Тема 2. Умовні переходи**

Виконання інструкцій в залежності від результату відповіді на питання. Порівняння як основа запитання. Ключове слово if. Ключове слово else. Ключове слово elif. Компонування декількох питань. Ключове слово or. Ключове слово end. Змінні типу bool.

**Тема 3. Цикли**

Можливість виконувати одні і ті самі дії певну кількість раз. Ключове слово while. Ключове слово for. Складна структура даних список. Довжина списку. Основні функції для роботи зі списком. Складна структура даних словник. Комбіноване використання списків та словників. Складна структура даних кортеж. Відмінність між структурами даних.

**Тема 4. Виняткові ситуації**

Дії, що призводять до появи виняткових ситуацій. Синтаксичні помилки, що виникають в процесі написання коду. Помилки реального часу виконання. Ключове слово try. Ключове слово except. Exception як узагальнення всіх помилок. Ключове слово else для визначення інструкцій безпомилкового виконання. Ключове слово pass. Ключове слово raise.

**Тема 5. Бібліотеки**

Бібліотека як набір файлів з кодом для використовувати у розробці. Сторонні та власні бібліотеки. Модуль як механізм забезпечення повторного використання коду. Ключове слово import. Ключове слово from. Пакет як варіант сторонньої бібліотеки. Веб ресурс PyPI. Встановлення пакетів реалізовано за допомогою скрипту pip.

**Тема 7. Робота з файлами**

Механізм довготривалого зберігання та використання інформації. Функція open(). Функція close(). Функції read() та write(). Ключове слово with. Файли в форматі CSV та однойменний модуль. Основні методи роботи з CSV файлами.

**Тема 8. Об’єктно орієнтовне програмування**

Процедурний підхід написання коду. ООП – інша парадигма вирішення завдань в програмуванні. Використання абстракцій як підхід в написання коду. Можливість створення нових типів даних з вказанням імені. Ключове слово class для опису власних типів даних, що є основною характеристикою ООП. Побудова абстракція object. Атрибути класу для опису властивостей абстракції object. Можливість задавати поведінку абстракції використовуючи підходи поняття methods. Ключове слово \_\_init\_\_. Ключове слово \_\_str\_\_. Розробка власної поведінки для абстракції. Можливість посиленого контролю за атрибутами абстракції використовуючи підходи поняття properties.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Структура навчальної дисципліни (*приклад заповнення*) | | | | | | | | | | | |
| **Назви змістових модулів і тем** | | **Кількість годин для денної/заочної форми навчання** | | | | | | | | | |
| **Всього** | | **У тому числі** | | | | | | | |
| **Лек.** | | **Прак.** | | **Лаб.** | | **Сам.роб.** | |
| **1** | | **2** | | **3** | | **4** | | **5** | | **6** | |
| **Модуль 1** | | | | | | | | | | | |
| **Змістовий модуль 1. Базові засади …** | | | | | | | | | | | |
| 1 | *Функції та змінні* |  | 15 |  | 2 |  |  |  | 2 | 11 |  |
| 2 | *Умовні переходи* |  | 13 |  | 2 |  |  |  |  | 11 |  |
| 3 | *Цикли* |  | 15 |  | 2 |  |  |  | 2 | 11 |  |
| 4 | *Виняткові ситуації* |  | 13 |  | 2 |  |  |  |  | 11 |  |
| 5 | *Бібліотеки* |  | 18 |  | 2 |  |  |  | 4 | 12 |  |
| 6 | *Тестування коду* |  | 13 |  | 2 |  |  |  |  | 11 |  |
| 7 | *Робота з файлами* |  | 15 |  | 2 |  |  |  | 2 | 11 |  |
| 8 | *Об’єктно орієнтовне програмування* |  | 18 |  | 2 |  |  |  | 4 | 12 |  |
|  | **Разом за змістовим модулем 1** |  |  |  |  |  |  |  |  |  |  |
|  | **Усього годин за дисципліну** | **120** | | **16** | |  |  | **14** | | **90** |  |

Теми практичних занять

|  |  |  |  |
| --- | --- | --- | --- |
| Теми лабораторних занять (*приклад заповнення*) | | | |
| **№ з/п** | **Назва теми** | **Кількість годин (д/з)** | |
| 1 | Робота з списками | 4 |  |
| 2 | Робота з файлами | 2 |  |
| 3 | Об’єктно орієнтовне програмування | 4 |  |
| 4 |  | 4 |  |
| **Разом** | | **14** |  |

# Самостійна робота

|  |  |  |
| --- | --- | --- |
| **№**  **з/п** | **Назва теми** | **Кількість**  **годин** |
| 1 | Подання інформації | 10 |
| 2 | Розгалужена структура | 10 |
| 3 | Циклічні структури | 10 |
| 4 | Рядки символів | 10 |
| 5 | Словник, кортеж | 10 |
| 6 | Рекурсія | 10 |
| 7 | Лямбда функції | 10 |
| 8 | Винятки | 10 |
| 9 | ООП | 10 |
|  | **Разом** | 90 |

# Індивідуальні завдання

Робочим планом передбачено виконання розрахунково-графічної роботи. Докладна інформація щодо змісту, варіантів завдань, порядку оформлення та захисту курсової роботи міститься в методичних рекомендаціях.

# Методи контролю

Поточний контроль проводиться шляхом спілкування із студентами під час лекцій, рішення задач під час практичних занять та консультацій, а також опитувань студентів під час захисту лабораторних робіт [13.1, 13.3 – 13.5].

За темами наприкінці п’ятого та десятого тижнів семестру проводяться самостійні роботи. Варіанти завдань до самостійних та контрольної роботи знаходяться у пакеті документів на дисципліну. Кожне завдання містить теоретичне запитання та практичне (задачу). За результатами самостійних та контрольної роботи виставляються модульні оцінки, причому бали, які набрані студентом під час поточного контролю, дораховуються до модульних оцінок. Підсумкові оцінки поточного контролю доводяться до відома студентів до початку сесії. Студенти, які повністю виконали навчальний план з дисципліни і за результатами поточного контролю отримали 60 балів, можуть залишити набрану кількість балів, як підсумкову семестрову оцінку.

У випадку, якщо студент хоче підвищити оцінку, отриману за балами, набраними протягом семестру, він складає екзамен з двома питаннями: одним теоретичним та одним практичним. До екзаменаційної відомості виставляються оцінки поточного контролю, до яких дораховується сума балів, набраних за результатами заліку. На заліку студенту пропонуються два питання: теоретичне (до 15 балів) та практичне (до 25 балів). Питання до заліку знаходяться в методичних вказівках до самостійної роботи студентів [13.6].

За результатами поточного семестру в екзаменаційну відомість виставляються оцінки поточного контролю, до яких дораховується сума балів, набраних за результатами екзамену. На екзамені студенту пропонуються два питання: теоретичне (до 15 балів) та практичне (до 25 балів). Екзаменаційні білети знаходяться в пакеті документів на дисципліну.

Набрана кількість балів переводиться у семестрову оцінку відповідно до шкали оцінювання, що наведена в наступному розділі. Усі колізії, пов’язані із несвоєчасним отриманням семестрової оцінки, вирішуються відповідно до Положення про оцінювання студентів.

# Розподіл балів, які отримують студенти

|  |  |
| --- | --- |
| **Поточний контроль за модулями** | |
| **Модуль за тематичним планом дисципліни та форма контролю** | **Кількість балів** |
|
| **Змістовий модуль 1. Назва модуля 1** | 0 … 100 |
| 1. Підготовленість до лабораторних робіт, самостійність та своєчасність їх виконання, відповіді на захисті 2. Виконання та захист РГР. 3. Фінальний іспит. | 0 … 40  0 … 20  0 … 40 |

**Шкала оцінювання: національна** та ECTS

|  |  |  |  |
| --- | --- | --- | --- |
| **Сума балів за всі види навчальної діяльності** | **Оцінка ECTS** | **Оцінка за національною шкалою** | |
| **для екзамену, курсової роботи** | **для заліку** |
| 90 – 100 | **А** | відмінно | зараховано |
| 82-89 | **В** | добре |
| 75-81 | **С** |
| 66-74 | **D** | задовільно |
| 60-65 | **Е** |
| 0-59 | **FX** | незадовільно | не зараховано |

# Інструменти, обладнання та програмне забезпечення, використання яких передбачає навчальна дисципліна

Лекційний матеріал подається у вигляді презентацій за допомогою медіа-проектора. Під час лекцій аналізуються проблемні ситуації, організується зворотний зв'язок з аудиторією шляхом формулювання запитань і стислих відповідей з обох сторін.

Виконання лабораторних робіт відбувається на комп’ютерах із використанням спеціалізованого програмного забезпечення.

# Методичне забезпечення

* <https://eln.stu.cn.ua/course/view.php?id=2731>

# Рекомендована література

**Базова**

1. C Programming Language. Prentice Hall, 1988. 274 p.
2. Python Basics: A Practical Introduction to Python 3 / D. Bader et al. Real Python (realpython.com), 2021. 635 p.
3. Ramalho L. Fluent Python: Clear, Concise, and Effective Programming / ed. by M. Blanchette, R. Roumeliotis. O’Reilly Media, 2015. 792 p.
4. Lutz M. Learning Python. 5th ed. 2013. 1540 p.
5. Barry P. Head First Python: A Brain-Friendly Guide. O'Reilly Media, 2016. 624 p.

# Додаткова

1. Modern Operating Systems: Forth Edition. Pearson, 2015.
2. Matthes E. Python Crash Course, 2nd Edition: A Hands-On, Project-Based Introduction to Programming. No Starch Press, 2019. 544 p.
3. Automate the Boring Stuff with Python: Practical Programming for Total Beginners. San Francisco, USA : No Starch Press, 2015. 504 p.
4. Think Python: How to Think Like a Computer Scientist / ed. by M. Blanchette. 2nd ed. Sebastopol, California, United States of America : O’Reilly Media, 2015. 222 p.
5. Slatkin B. Effective Python: 90 Specific Ways to Write Better Python. Addison-Wesley Professional, 2019. 480 p.
6. Gorelick M., Ozsvald I. High Performance Python: Practical Performant Programming for Humans. O'Reilly Media, Incorporated, 2020. 450 p.

# Інформаційні ресурси

* <https://cs50.harvard.edu/>
* <https://www.python.org/doc/>