**Кафедра кібербезпеки та математичного моделювання**

|  |  |
| --- | --- |
| Назва навчальної дисципліни | Технології програмування |
| Мова викладання | Українська |
| Курс та семестр вивчення | 2 курс, 1семестр  спеціальність |
| Викладач | Дюба Ігор Миколайович, викладач кафедри кібербезпеки та математичного моделювання |
| Профайл викладача |  |
| Контакти викладача | [idyuba@gmail.com](mailto:idyuba@gmail.com)  063 837 56 34 |
|  |  |

**Анотація курсу**

«Технології програмування» є обов’язковою дисципліною освітньої програми «Кібербезпека Вивчення дисципліни дозволяє студентам оволодіти базовими принципами для вивчення мови програмування, що можуть бути використанні в майбутньому при необхідності переходу на нову мову програмування, досвід використання якої відсутній.

Предметом вивчення навчальної дисципліни є принципи використання можливостей мови програмування для побудови алгоритмів вирішення завдань. Основною метою дисципліни є отримання студентами практичного досвіду використання нової мови програмування.

Програма навчальної дисципліни включає наступні теми:

**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 6. 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 7. Об'єктно-орієнтоване 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.

**Мета курсу**

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.

**Цілі курсу**

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.

**Очікувані результати навчання з дисципліни**

У результаті вивчення навчальної дисципліни студент повинен:

**знати**:

• базові елементи та конструкції мов програмування;

• основний синтаксис мов програмування python;

• 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;

• точно описувати симптоми помилок і чітко задавати питання.

**Обсяг курсу**. 4 кредитів (120 годин, з них 30 годин – аудиторні)

|  |  |
| --- | --- |
| **Вид заняття** | **Загальна к-сть годин** |
| Лекції | 16 годин (8 лекцій) |
| Лабораторні | 14 годин (7 лабораторних занять) |
| Самостійна робота (опрацювання лекційного матеріалу, основної та додаткової літератури; виконання практичних завдань. | 90 годин |

**Пререквізити**: відсутні

**Система оцінювання та вимоги**

|  |  |
| --- | --- |
| **Загальна система оцінювання курсу** | Оцінювання результатів навчання ЗВО з курсу «Технології програмування» здійснюється за 100-бальною шкалою та шкалою ЄКТС відповідно до «Положення про поточне та підсумкове оцінювання знань ЗВО НУ «Чернігівська політехніка» і охоплює:  – поточний контроль – 60 балів;  – підсумковий контроль (екзамен) – 40 балів (40%).  Структура екзаменаційного білету: завдання на оцінювання теоретичних знань – 20 балів; завдання на оцінювання практичних навичок (2 задачі по 10 балів)  Мінімальна загальна кількість балів для отримання позитивної оцінки з дисципліни – 60 балів. |
| **Вимоги до РГР** | Індивідуальна розрахунково-графічна робота передбачає виконання практичних завдань до кожної лекції. Стосовно результату виконання формується звіт, що завантажується в систему дистанційного навчання MOODLE. При відсутності розрахунково-графічної роботи або її неякісному виконанні (оцінка – менше 75% від максимальної кількості балів) здобувач вищої освіти до екзамену не допускається |
| **Лабораторні заняття** | Усне або письмове експрес-опитування; виконання та захист лабораторних робіт. |
| **Умови допуску до підсумкового контролю** | Здобувач вищої освіти допускається до підсумкового контролю у разі виконання всіх видів навчальної роботи, передбачених робочою програмою, та набрання мінімально необхідної кількості балів (35). В іншому випадку він буде мати академічну заборгованість, яку має право ліквідувати у порядку, передбаченому «Положенням про поточне та підсумкове оцінювання знань ЗВО НУ «Чернігівська політехніка». |

**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/