МІЖРЕГІОНАЛЬНА
АКАДЕМІЯ УПРАВЛІННЯ ПЕРСОНАЛОМ

МАУП

МЕТОДИЧНІ РЕКОМЕНДАЦІЇ ЩОДО ЗАБЕЗПЕЧЕННЯ САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ

з дисципліни
“ПРОГРАМУВАННЯ”

(для бакалаврів)

Київ 2008

Підготовлено доцентом кафедри прикладної математики та програмування
М. П. Дяченко

Затверджено на засіданні кафедри прикладної математики та програмування
(протокол № 1 від 30.08.07)

Схвалено Вченою радою Міжрегіональної Академії управління персоналом

Дяченко М. П. Методичні рекомендації щодо забезпечення самостійної роботи студентів з дисципліни “Програмування” (для бакалаврів). — К.: МАУП, 2008. — 16 с.

Методична розробка містить пояснювальну записку, тематичний зміст дисципліни, а також список літератури. Призначена для самостійної роботи студентів денної форми навчання, які вивчають дисципліну “Програмування”.

ПОЯСНЮВАЛЬНА ЗАПИСКА

Процес навчання в МАУП передбачає здобуття студентами знань як на аудиторних заняттях, так і під час самостійної та індивідуальної роботи на навчальній базі закладу та за його межами.

Метою навчальної дисципліни "Програмування" є опанування знаннями, уміннями та навичками дослідження задач прикладної математики наявними вільними засобами програмування, що поєднують у собі технологічність інтерпретуючих систем, швидкодію систем компілюючого типу зі здатністю об'єктно-орієнтовних систем налагоджуватись під конкретну предметну сферу застосувань. За базовий інструмент береться інтерактивне середовище вільної об'єктно-орієнтованої мови програмування Python з розвинутими бібліотеками класів для швидких математичних багатопоточних обчислень. Найважливішими завданнями курсу є засвоєння знарядь та засобів програмування, механізмів обміну інформацією між внутрішніми об'єктами середовища програмування та його зовнішнім середовищем через віконний інтерфейс користувача, стандартизовані формати відображення даних у вигляді текстових, табличних і графічних документів, їх довгочасного зберігання в базах даних та швидкого візуального відтворення, редагування і передавання на відстань.

Специфіка курсу зумовлена поєднанням різноманітних засобів інформаційних технологій у межах одного середовища (мовні засоби, склеюючі можливості, графіка, бази даних, формати даних, передавання даних тощо), а також високою активністю світового співтовариства Python-користувачів та його розробників. Останнє приводить до швидкого поточного удосконалення середовища в часі, що в одноразовому курсі відтворити неможливо: використання Python у практичній роботі має супроводжуватись постійною систематичною самоосвітою. Тому на аудиторні заняття доцільно винести базові питання засвоєння мови та її інтеграційних можливостей, ознайомлення з основними напрямами практичного застосування мови та мовного розвитку, а вивчення мовних розширень засобами спеціалізованих бібліотек класів опрацювати самостійно. Оскільки новітні інформаційні матеріали є, в більшості своїй, іншомовними, то вивчення курсу стимулює водночас і вивчення іноземної мови.

В процесі самостійної роботи над вивченням курсу студент оволодіває новітніми засобами програмування, засвоює принципи об'єктно-

орієнтовного мислення стосовно програмування прикладних задач математики, осягає внутрішній зв'язок між різноманітними засобами програмування, набуває досвіду їх практичного застосування. Разом з тим він знайомиться з тенденціями неформальної організації колективної роботи в середовищі Інтернету на прикладі створення та розвитку світовим співтовариством програмістів вільного інтерактивного середовища програмування Python, усвідомлює зв'язок між навчальними дисциплінами, включаючи і іноземні мови, без знання яких неможливо набути належного професійного рівня.

ЗМІСТ САМОСТІЙНОЇ РОБОТИ з дисципліни "ПРОГРАМУВАННЯ"

Тема 1. Загальні відомості про середовище програмування та його підготовка до роботи

Основні питання, які необхідно опрацювати і засвоїти.

1. Цілі, передумови створення вільної мови програмування Python і особливості її розвитку. Концептуальні особливості та порівняльна характеристика Python з іншими мовами програмування. Системи компілюючого і інтерпретуючого типів. Засоби поєднання Фортран- та C(C++)- бібліотек в інтерпретуючому середовищі Python.
2. Сайти локалізації дистрибутивів Python, його розширень та бібліотек. Отримання дистрибутивів та їх інсталяція на комп'ютері.
3. Windows розширення інтерпретатора Python. Організація системи допомоги і документації. Інтерактивний та стандартний режими роботи користувача в середовищі Python.
4. Модульність програм, зберігання модулів у зовнішній пам'яті, організація пошуку модулів та способи імпорту модулів. Засоби зв'язку середовища програмування з операційною системою. Навігація по файловій системі операційної системи.

Література [1–4]

Студент повинен знати:

- основні інформаційні сайти поточного розвитку мови Python та її бібліотечних розширень;
- концептуальні особливості мови, сфери її переважного використання;

- переваги та недоліки мови в застосуванні до математичних обчислень, засоби усунення або пом'якшення дії останніх (засоби компіляції в режимі виконання Python-скриптів та побудови компільованих розширень);
- командне вікно інтерпретатора, віконну оболонку користувача для програмування Windows застосувань на базі MFC (Pythonwin);
- загальні можливості основних С-розширень мови для математичних обчислень та їх графічного відображення (Numpy, Scipy, Matplotlib, Dislin).

Студент повинен вміти:

- завантажувати з Інтернет систему програмування Python на ПК та встановлювати її для ОС MS Windows і налагоджувати під власні потреби, використовуючи як стандартні, так і нестандартні ресурси середовища Python;
- користуватись стандартною документацією в складі дистрибутиву (із домашньої сторінки http:/python.org.) системи програмування та вести цілеспрямований пошук необхідних ресурсів в Інтернеті;
- працювати у власному та розширеному для msWindows віконному pyWin32 середовищах (http://starship.python.net/crew/mhammond/) як в інтерактивному, так і стандартному режимах інтерпретатора Python;
- імпортувати ресурси стандартних і нестандартних бібліотек модулів в свою програму.

Індивідуальне завдання

Тип завдання: Підготовка до семінару на тему "Області ефективного використання мови Python та засоби підвищення ефективності".

Мета завдання: перевірка знань студентів, набутих у процесі вивчення теми.

Самостійна робота: проаналізувати за літературними джерелами можливості стандартного пакета Python.

Тема 2. Базові засоби програмування

Основні питання, які необхідно опрацювати і засвоїти.

1. Мінімальна програма на Python. Структура програми та її документування (заголовки і коментарі). Модуль, речення, оператор. Відступи, їх роль у стилі програмування на Python. Доступ

до тексту документування із програм та його використання для генерації технічної документації.

2. Вбудовані типи даних: числа (цілі, дійсні та комплексні), рядки, списки, контейнери, словники. Поняття змінної в Python як посилання на область пам'яті з об'єктними даними. Зміст операція присвоєння. Особливості адресації змінних та копіювання змінних. Змінні, що допускають модифікацію в режимі виконання та не допускають такого. Вирази. Динамічне формування виразів. Функції eval() та exec().
3. Ресурси модуля: константи, змінні та функції. Форма існування модуля. Процедурне програмування. Lambda-числення та його застосування для динамічного визначення функцій. Видимість даних, області видимості. Простір імен. Інтроспекція власних об'єктів в Python-скриптах. Оператори інтроспекції.
4. Організація керування потоком виконання скриптів. Цикли та операції з умовами. Вкладені умовні оператори. Оператори продовження та переривання циклів. Ітератори та генератори. Механізм винятків у керуванні обчислювальним процесом і забезпеченні його надійності.
5. Формати відображення даних (з вирівнюванням і без нього): числових (цілих і з плаваючою комою), рядкових та структурованих.

Література [1–4]

Студент повинен знати:

- базові синтаксичні конструкції для відображення в скриптах обчислювальних алгоритмів (вирази, операції керування обчислювальним процесом);
- формати для візуального відображення результатів обчислень на екрані дисплея;
- базові типи, вбудовані в мову програмування, та їх методи;
- засоби дослідження програмних об'єктів (інтроспекція) та області їх видимості;
- засоби формування ресурсів модуля (константи, змінні, функції);
- засоби забезпечення надійності обчислювального процесу (механізм винятків).

Студент повинен вміти:

- раціонально використовувати базові синтаксичні конструкції для конструювання програм (з дотриманням рекомендо-

ваного стилю програмування) і їх накопичувати в зовнішній пам'яті;
- використовувати механізм винятків для блокування аварійних ситуацій;
- оформляти відображення результатів розрахунків та супутнього тексту з застосуванням системи форматів Python;
- оформляти програмні модулі з чітким документуванням їх змісту;
- відлагоджувати програмні модулі з застосуванням засобів Python.

Індивідуальне завдання

Тип завдання: Самостійна розробка цілісної програми за індивідуальним алгоритмом.

Мета завдання: перевірка знань студентів, набутих у процесі вивчення теми.

Самостійна робота: Засвоїти та закріпити знання базових конструкцій на контрольних прикладах навчальних посібників.

Тема 3. Робота з файлами

Основні питання, які необхідно опрацювати і засвоїти.

1. Файли. Створення, відкриття та закриття файлів. Режими доступу до файлів.
2. Операції читання з файлів та запису в файли рядками і довільними порціями символів.
3. Навігація по даним файла. Курсор, керування положенням курсору. Способи запису та читання довільних структур даних. Використання винятків для безпечної роботи з даними файлів.
4. Спеціалізовані (серіалізація об'єктів) засоби обміну скриптів з зовнішньою памяттю даними довільної структури. Модуль pickle.
5. Каталоги. Створення та знищення каталогів. Поточний каталог. Навігація по системі каталогів, способи пошуку окремих файлів та груп файлів. Обробка імен файлів.
6. Засоби накопичення статистика за використанням файлів.
7. Ясність та прозорість текстів Python-програм. Рекомендований стиль програмування мовою Python.

Література [1–6]

Студент повинен знати:

- основні інформаційні сайти поточного розвитку мови Python та її розширень;
- можливості віконної оболонки користувача Pythonwin;
- концептуальні особливості мови, область її переважного використання;
- переваги та недоліки програмування задач прикладної математики в середовищі Python, засоби усунення або пом'якшення дії останніх (інтеграція з іншими мовами, just-in-time компіляція, спеціалізовані розширення);
- структуру Python-скриптів, засоби їх документування, зберігання та особливості стилю програмування;
- базові синтаксичні конструкції для відображення в скриптах обчислювальних алгоритмів та організації обміну даними з зовнішньою пам'яттю в процесі їх виконання.

Студент повинен вміти:

- встановлювати систему програмування Python на ПК;
- користуватись стандартною документацією в складі системи;
- складати, відлагоджувати і зберігати скрипти в середовищі Python з застосуванням стандартного і інтерактивного режимів роботи та дотриманням рекомендованого стилю програмування;
- використовувати із середовища інтерпретатора файлову систему msWindows для організації обміну даними між скриптами та зовнішньою пам'яттю комп'ютера;
- використовувати механізм винятків для блокування аварійних ситуацій;
- оформляти відображення результатів розрахунків та супутнього тексту з застосуванням системи форматів Python;
- оформляти скрипти з чітким документуванням та коментарями їх змісту.

Індивідуальне завдання

Тип завдання: Класифікація задач, розв'язування яких підтримується в середовищі Python (семінар).

Мета завдання: перевірка знань студентів, набутих у процесі вивчення теми.

Самостійна робота: проаналізувати за літературними джерелами можливості стандартного пакета Python.

Тема 4. Об'єктно-орієнтовне програмування

Основні питання, які необхідно опрацювати і засвоїти.

1. Поняття об'єктно-орієнтовного програмування (ООП). Об'єкти, їх стан та поведінка.
2. Конструктори об'єктів (класи). Атрибути і методи класів. Інкапсуляція, успадкування, поліморфізм. Абстрактні класи. Множинне успадкування. Змішування класів (mixing). Змінні класів, методи класів. Статичні методи. Специфіка успадкування класів, екземпляри яких не допускають модифікації. Метод класів New.
3. Методи і атрибути об'єктів (екземплярів) класів. Класифікація стандартних об'єктів (контейнери, ітератори, асоціації, генератори). Слабкі посилання. Глибоке копіювання об'єктів. Серіалізація об'єктів.
4. Метакласи. Застосування метакласів. Мультиметоди.
5. Критичний аналіз концепції ООП. Переваги та недоліки ООП. Області переважного використання ООП.

Література [1–4]

Студент повинен знати:

- принципи і поняття ООП, організацію класів та способи успадкування їх ресурсів класами-спадкоємцями, синтаксис і способи побудови успадкованих класів і змішування, а також застосування метакласів;
- синтаксис і способи конструювання об'єктів класів та їх динамічну модифікацію;
- способи використання класів і метакласів для підвищення рівня програмування та адаптації середовища програмування до потреб конкретної проблемної області;
- засоби серіалізації об'єктів (збереження їх в зовнішній памяті та відновлення в середовищі інтерпретації);
- межі переважного використання ООП в програмуванні прикладних задач;
- можливості стандартних бібліотек класів Python (вбудованих в середовище).

Студент повинен вміти:

- структурувати прикладну задачу і конструювати ієрархію класів, що її розв'язують, застосовуючи успадкування ресурсів батьківських класів та метод змішування;

- застосовувати ресурси класів для програмування конкретних завдань, що покриваються класом (ієрархією класів);
- редагувати (за необхідності) об'єкти в процесі виконання програм;
- редагувати, зберігати об'єкти в зовнішній пам'яті та відновлювати їх в оперативній пам'яті в процесі обчислень, використовуючи файлові операції та методологію pickling;
- використовувати ресурси вбудованих класів для розв'язування конкретних задач прикладної математики.

Тема 5. Паралельні обчислення

Основні питання, які необхідно опрацювати і засвоїти.

1. Поняття потоку керування. Створення потоків. Механізми високорівневої координації потоків: семафори, замки, mutex'и, події, умови. Таймер, його використання в програмуванні потоків. Модулі threading та Queue.
2. Механізми низькорівневого програмування потоків. Модуль thread. Приклади мультипроцесних задач.

Література [1–4]

Студент повинен знати:

- зміст поняття потоку, механізми ініціювання потоків та координації їх спільної роботи як на високому, так і низькому рівні (ресурси модулів threading, Queue та thread).

Студент повинен вміти:

- розпаралелювати потоки обчислень в довготривалих та мультипроцесних задачах;
- синхронізувати виконання потоків, використовуючи таймер та інші механізми координації їх одночасної роботи.
- дотримуватись ясного стилю програмування розпаралелених завдань.

Тема 6. Функціональне програмування

Основні питання, які необхідно опрацювати і засвоїти.

1. Поняття функціонального програмування як композиції послідовного виконання функцій. Основні функції зазначеного стилю програмування. Область переважного використання.
2. Обробка послідовностей. Функції xrange(), map(), filter(), sum(), reduce(), zip() , iter(), enumerate(), sorted().

3. Модуль itertool, його функції (chain(), repeat(), count(), cycle(), imap(), starmap(), ifilter(), takewhile(), dropwhile(), izip(), groupby(), tee()) та їх застосування.
4. Власні ітератори та прості генератори. Генераторні вирази. Каррінг.

Література [1–4]

Студент повинен знати:

- зміст функціонального програмування, відповідний набір стандартних функцій, переваги та недоліки зазначеного стилю програмування, рекомендації щодо їх використання.

Студент повинен вміти:

- ефективно використовувати знаряддя функціонального програмування.

Тема 7. Математика в Python

Основні питання, які необхідно опрацювати і засвоїти.

1. Засоби ефективних математичних обчислень. Загальна характеристика модуля Numpy. Доступ до ресурсів модуля.
2. Масиви (вектори та матриці) модуля Numpy та їх властивості. Конструктор масивів array(). Типи елементів масивів. Способи ініціювання масивів. Параметри масивів, методи їх ініціювання та редагування.
3. Арифметичні операції над масивами. Швидкодіючі операції "за місцем в пам'яті". Логічні операції. Властивість розповсюдження операцій. Зрізи.
4. Засоби перетворення масивів: матриця — вектор і навпаки. Матричні операції над масивами: перемножування матриць та їх транспонування.
5. Клас універсальних функцій над масивами. Спеціальні методи класу (accumulate(), reduce() і reduceat()). Функції доступу до окремих груп та елементів масивів.
6. Операції лінійної алгебри (модуль LinearAlgebra). Обчислення визначників, розв'язків лінійних рівнянь.
7. Генерація масивів випадкових чисел (модуль RandomArray). Закони розподілення генерованих послідовностей.
8. Пакет плоскої та просторової графіки matplotlib та його реалізація в Numpy (pylab).

Література [5–6]

Студент повинен знати:

- засоби ініціювання масивів, векторних та матричних структур з використанням модуля Numpy і методи їх обробки;
- засоби відображення графічної інформації pylab.

Студент повинен вміти:

- використовувати швидкодіючі матричні операції та операції над великими числовими масивами для програмування завдань з обчислювальної математики.
- використовувати ресурси модуля pylab для графічного відображення числової інформації.

Тема 8. Програмування графічного інтерфейсу користувача

Основні питання, які необхідно опрацювати і засвоїти.

1. Подійно-орієнтовне програмування застосувань під Windows. Головне вікно застосування. Події та процедури обслуговування подій. Неперервний цикл чекання подій.
2. Події, що пов'язані безпосередньо зі зміною стану вікон (FocusIn, FocusOut, Visibility, Reparent, Activate, Deactivate, Destroy) та положенням курсору по відношенню до вікон (Enter, Leave). Події від маніпуляцій з клавішами миші (ButtonPress, ButtonReleas, Motion, MouseWheel) та клавіатури (KeyPress, KeyRelease)
3. Елементи віконного інтерфейсу (віджети). Етикетки, однорядкові та багаторядкові поля введення тексту та графічних образів, канва для малювання, звичайні кнопки, радіокнопки та кнопки вибору, звичайні меню та спливаючі меню, списки вибору, шкали, смуги прокручування та контейнери.
4. Засоби розміщення елементів інтерфейсу в полі головного вікна застосування (менеджери віджетів). Безпосередня прив'язка віджетів до координат відповідних точок контейнера (place). Упаковування віджетів в контейнер (pack). Побудова сітки для розміщення віджетів (grid).
5. Засоби оформлення зовнішнього вигляду користувацького інтерфейсу: колір фону та переднього плану текстових написів, тип і розміри текстових шрифтів, рельєф та облямовуючі бордюри.

Література [8]

Студент повинен знати:

- бібліотеку класів віконного інтерфейсу Tkinter.

Студент повинен вміти:

- користуватись ресурсами класу Tkinter для програмування користувацького інтерфейсу.

Тема 9. Засоби програмування баз даних

Основні питання, які необхідно опрацювати і засвоїти.

1. Поняття реляційної СУБД. Таблиця, рядок, стовпчик, кортеж. Інтерфейс Python-скрипта з базою даних. Об'єкт-з'єднання, об'єкт-курсор, об'єкти типів даних СУБД.
2. Конструктор об'єктів з'єднання (connect). Параметри конструктора connect: джерело даних, ім'я користувача, пароль, адреса хосту бази даних, ім'я бази даних.
3. Керуючі константи інтерфйсу бази даних: apilevel, threadsafet, paramstyl, "format", "pyformat", "qmark", "numeric", "named".
4. Виняткові повідомлення: Warning (попередження), Error (помилка), InterfaceError (помилка інтерфейсу), DatabaseError (помилка звернення до бази даних), DataError (помилка в обробці даних), OperationalError (збій з'єднання з базою даних), IntegrityError (порушення цілісності бази даних), InternalError (внутрішня помилка бази даних), ProgrammingError (помилка в запиті до бази даних), NotSupportedError (не реалізована підтримка запиту).
5. Поняття трансакції. Методи об'єкта-з'єднання: закрита з'єднання з базою даних (close), здійснення транзакції (commit), відновлення попереднього стану, бази даних (rollback). Видача курсор-об'єкта (cursor).
6. Поняття курсору бази даних. Атрибути курсору: arraysize (кількість записів, що повертаються методом fetchmany), callproc (результат, дії fetch-методу), close (закриття, объект-курсору), description (name, type_code, display_size, internal_size, precision, scale, null_ok, параметри стовпчика результатів запиту), execute(operation, parameters) (виконання запиту до бази даних), executemany (operation, seq_of_parameters) (виконання серії запитів за шаблоном), fetchall (всі записи за запитом), fetchmany ([size]) (серія наступних записів заданої кількості), nextset (перехід до початку наступної порції даних запиту), rowcount (кількість записів останнього запиту), setinputsizes (sizes) (резерв

пам'яті для операцій з базами даних), setoutputsize(size[, column]) (розмір буфера для параметра з номером column).

7. Типи даних СУБД: STRING (рядок та символ), BINARY (бінарний об'єкт), NUMBER (число), DATATIME (дата та час), ROWID (ідентифікатор запису), NONE (null-значення), та їх конструктори: Date (рік, місяць, день), Time (години, хвилини, секунди), Timestamp (рік, місяць, день, години, хвилини, секунди), DateFromTicks(secs) (дата у вигляді числа секунд від початку епохи — 1 січня 1970 року), TimeFromTicks(secs), TimestampFromTicks(secs) (еквівалент попереднього), Binary(string) (великий бінарний об'єкт на базі рядкового типу string).
8. Робота в СУБД SQLite. З'єднання з базою даних (connect), створення одного чи декількох курсорів (cursor), формування та виконання SQL-запиту (execute), отримання результатів запиту (fetchone), завершення транзакції або відкат (commit(), rollback()).), закриття з'єднання (close).
9. Поняття мови запитів (SQL). Створення таблиці (CREATE TABLE). Відбір даних із таблиці (SELECT<>FROM<>WHERE <>LIKE<>). Вставка даних в таблицю (INSERT). Оновлення записів (UPDATE). Вилучення записів (DELETE). Опорожнення таблиці (DROP). Математичні операції над даними. Об'єднання таблиць (JOIN).
10. Робота в середовищі інтерпретатора СУБД SQLite.

Література [4; 7]

Студент повинен знати:

- організацію інтерфейсу Python, яка має дотримуватись при вбудовуванні нової бази даних в середовище програмування;
- реалізацію інтерфейсу до реляційної бази даних SQLite, мову запитів SQL до бази даних SQLite.

Студент повинен вміти:

- оцінювати особливості конкретної реалізації інтерфейсу до бази даних (поглиблені знання для розробника інтерфейсів до СУБД);
- створювати, поповнювати та редагувати бази даних застосувань в СУБД SQLite, вести пошук, витягувати та відображати дані із бази на відеотерміналі за допомогою мови запитів СУБД SQL.

СПИСОК ЛІТЕРАТУРИ

Основна

1. *Бизли Д.* "Язык программирования Python", Диасофт, 2000.
2. *Россум, ван, Г., Дрейк, Ф. Л. Дж., Откидач Д. С. и др.* Язык программирования Python (недрукований матеріал, див. http://www.python.ru/files/book-ods.pdf)
3. *Пилгрим, М.* Вглубь языка Python (http://ru.diveintopython.org/)
4. *Сузи, Р. А.* Python. Наиболее полное руководство (+CD). — СПб.: БХВ-Санкт-Петербург, 2002., див. також (http://www.intuit.ru/department/pl/python/)
5. Інтернет ресурси математики в Python:
   http:/scipy.org;
   http:/numpy.scipy.org
6. Інтернет ресурси плаской та просторової графіки:
   http://matplotlib.sourceforge.net/
   http://dislin.de
7. Інтернет ресурси СУБД SQLITE:
   http://www.python.org/topics/database/
   http://sqlite.org/sqlite-3_3_15.zip
   http://linux.fopf.mipt.ru/files/scripting/php/ru/ref.sqlite.html
   http://docs.python.org/lib/module-sqlite3.html
   http://initd.org/pub/software/pysqlite/doc/usage-guide.html
   http://sqlcourse.com/
8. Tkinter: http://effbot.org/tkinterbook/)

Додаткова

1. *Андре Лесса.* Python: Руководство разработчика. — ДиаСофтЮП, 2001.
2. *Лейнингем И.* Освой самостоятельно Python за 24 часа. — М., Вильямс, 2001.
3. *Лутц М.* Программирование на Python. — СПб.: Символ-Плюс, 2002.
4. *Пітон*, домашня сторінка розробника http://www.python.org/
5. *Пітон*, вікипедія, (http://ru.wikipedia.org/wiki/Python)
6. *Пітон*, стандартні бібліотеки, англ., (ресурси Інтернет) (http://effbot.org/zone/librarybook-index.htm)
7. *Чаплыгин, А. Н. и др.* Учимся программировать вместе с Питоном. (http://pythonbook.it-arts.ru/)
8. *Ferg, Stephen.* Event-Driven Programmin. Introduction, Tutorial, History (http://Tutorial_EventDrivenProgramming.sourceforge.net)
9. *Grayson, John E.* Python and Tkinter Programming. Manning Publications Company, 1999.
10. *Frederik Lundh.* Python Standard Library. — O'Reilly & Associates, 2001.

ЗМІСТ

Відповідальний за випуск *А. Д. Вегеренко*
Редактор *О. М. Коваленко*
Комп’ютерне верстання *Н. М. Музиченко*

Зам. № ВКЦ-3531

Міжрегіональна Академія управління персоналом (МАУП)
03039 Київ-39, вул. Фрометівська, 2, МАУП