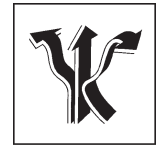
МІЖРЕГІОНАЛЬНА
АКАДЕМІЯ УПРАВЛІННЯ ПЕРСОНАЛОМ




# МЕТОДИЧНІ МАТЕРІАЛИ ЩОДО ЗАБЕЗПЕЧЕННЯ САМОСТІЙНОЇ РОБОТИ СТУДЕНТІВ

**з дисципліни**

# "ТЕХНОЛОГІЯ ПРОГРАМУВАННЯ ТА СТВОРЕННЯ ПРОГРАМНИХ ПРОДУКТІВ"

**(для бакалаврів)**

Київ 2008

Підготовлено доцентом кафедри прикладної математики та програмування
*Н. М. Москальковою*

Затверджено на засіданні кафедри інформатики та інформаційних технлогій (протокол № 13 від 03.08.07)

*Схвалено Вченою радою Міжрегіональної Академії управління персоналом*

**Москалькова Н. М.** Методичні матеріали щодо забезпечення самостійної роботи студентів з дисципліни "Технологія програмування та створення програмних продуктів" (для бакалаврів). — К.: МАУП, 2008. — 30 с.

Методичні матеріали містять пояснювальну записку, питання для самостійного вивчення студентами та самоконтролю, теми рефератів, тестові завдання, а також список літератури.

Призначена для методичного забезпечення самостійної роботи з дисципліни "Технологія програмування та створення програмних продуктів" студентів денної форми навчання, які здобувають освіту за спеціальністю "Програмне забезпечення автоматизованих систем".

## *ПОЯСНЮВАЛЬНА ЗАПИСКА*

Сучасна реформа вищої освіти — це насамперед перехід від парадигми навчання до парадигми освіти, самоосвіти. Тому при реформуванні вищої школи, введенні кредитно-модульної технології навчання значно зростає роль самостійної роботи студентів. Самостійна робота є основним засобом опанування навчального матеріалу у позааудиторний час. Студент, який хоче якомога краще оволодіти професією, має добре розуміти: на занятті викладач подає основи знань, навчає, як учити, виокремлює ті ключові істини дисципліни, які пробуджують у молодої людини потяг до поглиблення й удосконалення усіх знань. Лише постійне самостійне навчання дає можливість якомога ближче підійти до вершини знань певної галузі, оволодіти такою сумою знань і вмінь, які дали б змогу заявити про себе як про професіонала.

Самостійна робота студентів є надзвичайно важливою складовою підготовки спеціалістів з напряму "Комп'ютерні науки", зокрема зі спеціальності "Програмне забезпечення автоматизованих систем". Теоретичний матеріал з програмування потребує багаторазового підкріплення практичними прикладами. Студенти мають набути навичок самостійно розробляти програмне забезпечення на всіх етапах (проектування, створення, тестування тощо). Це потребує систематичного виконання практичних завдань протягом семестру та підготовки до кожного практичного заняття.

Самостійна робота повинна бути спланована, організаційно і методично спрямована як особиста творча праця студента без безпосередньої взаємодії з викладачем. Згідно з державними стандартами навчальний матеріал навчальної дисципліни, передбачений робочим навчальним планом для засвоєння студентом у процесі самостійної роботи, виноситься на підсумковий контроль поряд з навчальним матеріалом, який опрацьовувався при проведенні навчальних занять. Самостійна робота студента над засвоєнням навчального матеріалу з конкретної дисципліни може виконуватися у бібліотеці вищого навчального закладу, навчальних кабінетах, комп'ютерних класах (лабораторіях), а також в домашніх умовах. Навчальний час для самостійної роботи регламентується робочим навчальним планом і згідно з Болонською декларацією повинен становити не менше 50 % загального обсягу навчального часу, відведеного для вивчення конкретної дисципліни. У необхідних випадках ця робота проводиться відповідно до

заздалегідь складеного графіка, що гарантує можливість індивідуального доступу студента до потрібних дидактичних засобів. Графік доводиться до відома студентів на початку поточного семестру. При організації самостійної роботи з використанням складного обладнання чи устаткування, складних систем доступу до інформації (наприклад, комп'ютерних баз даних, систем автоматизованого проектування тощо) передбачається можливість отримання необхідної консультації або допомоги з боку фахівця.

Самостійна навчальна діяльність студента може здійснюватись через:

- запам'ятовування певної інформації за рахунок уважного слухання і конспектування лекцій; активної роботи під час практичних занять;
- роботу над конспектами лекцій, планами практичних занять;
- опрацювання літературних джерел (конспектування самостійно вивченого матеріалу тощо);
- роботу з каталогами звичайних і електронних бібліотек, інформаційно-пошуковими сервісами Internet;
- вивчення навчального матеріалу за паперовими та електронними підручниками, навчальними посібниками, практикумами тощо;
- опрацювання матеріалу за першоджерелами, науковою і спеціальною літературою;
- підготовку доповідей, рефератів, написання курсових робіт, пошукову і науково-дослідну діяльність;
- самотестування.

*Самостійна робота студента під час лекції.* Лекційний матеріал призначається для спрямування студентів у раціональному напрямі щодо вивчення навчальної дисципліни і акцентуванні уваги на найскладніших, вузлових питаннях навчальної дисципліни. Належне ведення конспекту сприяє збереженню необхідної інформації та дає змогу в подальшому аналізувати її. За умови подання лекційного матеріалу в усній формі одночасно засвоюється до 20 % інформації. Викладання інформатики в комп'ютерних класах або в аудиторіях, обладнаних мультимедійним обладнанням (наприклад, мультимедійним проектором або сенсорним екраном), водночас з демонстрацією прийомів роботи з користувальницьким інтерфейсом програми дозволяє підвищити рівень засвоєння лекційного матеріалу до 50–60 %.

*Робота над конспектами лекцій, планами практичних занять.* При підготовці до практичних занять студент має спиратися на конспект лекції. При його опрацюванні слід зіставити законспектований матеріал з планом практичного заняття, що міститься у методичних матеріалах для практичних занять або у навчально-методичному комплексі. Якщо у конспекті бракує матеріалів з окремих питань або деякі з них розкриті недостатньо чи винесені на самостійне опрацювання, студент повинен звернутися до рекомендованих підручників, навчальних посібників і методичних матеріалів. Підготовку для практичного заняття краще здійснювати з використанням ПЕОМ зі встановленим на ньому відповідним програмним забезпеченням. За цієї можливості слід використовувати інтерактивні довідкові системи програм *MS Office* та інформаційно-пошукові системи *Internet*.

Працювати із підручниками, навчальними посібниками, методичними вказівками, практикумами, науковою і спеціальною літературою, незалежно від типу носія (паперового чи електронного), необхідно таким чином, щоб отримати максимум теоретичних знань і навичок. При роботі з джерелами насамперед слід ознайомитися з їх змістом, щоб визначити, чи необхідно їх опрацьовувати і чи мають вони відношення до навчального курсу, що вивчається, і тільки після цього визначити послідовність їх опрацювання, щоб відібрати необхідний для вивчення матеріал (глави, розділи тощо). В разі роботи з інтерактивними електронними джерелами слід використовувати можливості навігації за документами, що надаються сучасними програмами, призначеними для читання електронних документів відповідних форматів (*MS Word, Adobe Reader, Adobe Acrobat* та ін.), і особливо переваги гіпертекстової технології подачі навчального матеріалу, а саме — за допомогою гіперпосилань знаходити відповіді на поставлені питання. При опрацюванні матеріалу необхідно з'ясувати суть питання, що вивчається, не уникаючи при цьому визначення суті незрозумілих чи незнайомих слів, термінів. Саме інтерактивні гіпертекстові електронні джерела (довідки у складі програмних продуктів, електронні посібники та словники) дозволяють конкретизувати терміни та визначення якнайшвидше. Працюючи з електронними джерелами необхідно аналізувати прочитане, порівнюючи з матеріалами лекції, робити логічні висновки, позначати незрозумілі положення з метою їх подальшого з'ясування на практичному занятті. Бажано відпрацювати зручну для себе певну систему позначень (позначки на полях конспекту, підкреслення маркерами різних ко-

льорів, доповнення конспекту альтернативними формулюваннями та посиланнями на інші джерела тощо) та фіксації опрацьованого матеріалу. Сучасні текстові редактори (в першу чергу *MS Word*) дають можливість створити електронний конспект з примітками, виносками, коментарями та здійснити його роздрук. Для самостійного поглибленого вивчення навчального матеріалу студенту слід звертатися до наукової та спеціальної літератури, яка може бути і не зазначеною в навчально-методичному комплексі. Використання самостійно отриманих відомостей як у навчанні, так і на практиці є, безперечно, цінним здобутком діяльності студента на шляху формування свого професійного потенціалу.

Знання з інформатики відносяться до базової підготовки сучасної людини. Вони є основою для подальшого засвоєння спеціалізованого програмного забезпечення за фаховою освітою і застосовуватимуться в будь-якій сфері діяльності. З позицій випереджаючої освіти навчання тільки за конспектом лекцій і основною літературою, вказаною у навчальній програмі, є недостатнім. У більшості випадків належна підготовка потребує вмінь швидко знаходити та опрацьовувати необхідний матеріал за першоджерелами, науковою і спеціальною літературою та коректно цитувати його. Перелік такої літератури, як правило, наводиться у навчально-методичному комплексі навчальної дисципліни. Тому завдання студента зводиться до самостійного знаходження цих матеріалів шляхом пошуку у паперових або електронних фондах бібліотек, а також у різноманітних файлових архівах, базах даних та базах знань, доступ до яких здійснюється за допомогою відповідних сервісів *Internet* (в основному — *Word Wide Web, FTP* та *UseNet newsgroups*).

Для пошуку документа використовуються різні його ознаки, насамперед реквізити (УДК. Автор(и). Заголовок опису. Основний заголовок: відомості, що відносяться до заголовка/Відомості про відповідальність. Відомості про видання (в тому числі URL-адреса Web-документа або Ftp-файла). Місце видання, дата видання. Обсяг.). УДК — це універсальна десяткова класифікація будь-яких офіційних видань в усьому світі. Відповідні довідники видаються багатьма мовами і постійно оновлюються. В Україні у 2006 р. Книжковою палатою України ім. І. Федорова видано "Універсальну десяткову класифікацію. Зміни та доповнення. Випуск 4" у паперовому варіанті. Довідкова база УДК постійно нарощується за рахунок електронних видань. Знання УДК дозволяє швидко знайти необхідне джерело за

систематичним бібліотечним каталогом. Наприклад, УДК видань з інформаційних технологій починається з 004.

Коли код УДК невідомий, то необхідно звернутися до алфавітного каталогу бібліотеки і за назвою джерела або прізвищем та ініціалами автора знайти відповідний бібліотечний шифр джерела.

Якщо ж студент здійснює наукове дослідження, готує наукову доповідь або виступ на конференції і йому не відомі реквізити джерела або саме джерело, то слід звернутися до систематичного бібліотечного каталогу. Завдання студента полягає у пошуку необхідної галузі (підгалузі), що охоплює розшукувану інформацію, а потім у межах цієї галузі (підгалузі) — картки з необхідним джерелом і бібліотечним шифром. У подальшому студент повинен оформити бібліотечне замовлення на літературу встановленого зразка, до якого внести шифр знайденого джерела та усі необхідні реквізити. Робота з електронними фондами в цьому варіанті значно ефективніша, оскільки у розвинених бібліотеках облік літератури ведеться в середовищах систем управління базами даних, за допомогою яких пошук потрібної інформації здійснюється найефективніше.

Сервіси мережі *Internet* надають унікальні можливості для знаходження літературних джерел у географічно віддалених фондах та архівах, а також шляхом участі у мережних конференціях, де можна отримати відповіді та поради щодо питань з розшукуваної інформації. Для доступу до *Internet*-ресурсів необхідно знати їх мережну адресу. Оскільки *Internet* постійно оновлюється і розвивається, в ньому немає єдиного каталога, змісту або наочного покажчика ресурсів. Проте в *Internet* існують різні інформаційно-пошукові системи, що допомагають користувачам знайти те, що їм потрібно. Це в першу чергу тематичні каталоги і так звані пошукові машини. Тематичні (наочні) каталоги — це інформаційно-довідкові системи, підготовлені вручну редакторами цих систем на основі інформації, зібраної на серверах *Internet*. Інформація в цих системах розподіляється за тематичними розділами відповідно до певної ієрархії. Верхній рівень розділів містить загальні категорії (наприклад, “Інтернет”, “Бізнес”, “Мистецтво”, “Освіта” тощо), а нижній посилання на конкретні *Web*-сторінки або інші інформаційні ресурси. Для швидкого переходу до потрібного розділу тематичного каталогу можна скористатися вбудованою системою автоматичного пошуку за ключовими словами. Для цього в рядок запиту слід ввести ключове слово (поєднання слів), клацнути ***Пошук***, і система повідомить, чи є відповідний розділ в її каталозі і за-

пропонує перейти в нього, одразу, оминаючи проміжні розділи. Рекомендуємо використовувати каталоги: http://www.yahoo.com, http://www.portal.edu.ru, http://www.ipl.org

Пошукові системи є складними інформаційно-довідковими системами, що автоматично генеруються на основі даних, які збираються мережними програмами-роботами в базах *Internet*, на запит користувача дають відповідь через посилання на різні *Internet*-ресурси. Запит здійснюється за певною процедурою (на певній мові), яка може різнитися в різних системах, проте у спрощеному вигляді зводиться до того, що користувач вводить у спеціальному полі (або в кількох полях) ключові слова, та/або словосполучення, що найточніше відповідають суті проблеми.

Серед загальних положень мов запитів виокремимо такі:

- ключові слова можна вводити у відповідне поле пошукової системи поодинці, послідовно звужуючи пошук, або ж вводити відразу кілька слів, розділяючи їх пробілами чи комами. Регістр не має значення;
- режим пошуку "AND" ("І") означає, що буде знайдено тільки ті дані, де зустрічається кожне з ключових слів;
- при використанні режиму "OR" ("АБО") результатом пошуку будуть усі дані, де зустрічається хоч би одне ключове слово;
- використовуйте знаки "+" і "–" перед ключовим словом. Щоб виключити документи, де зустрічається певне слово, поставте перед ним мінус. І навпаки, щоб певне слово обов'язково було присутнє в документі, поставте перед ним плюс. Зверніть увагу на те, що між знаком і словом не повинно бути пробілу;
- якщо Ви хочете виключити яке-небудь слово з пошуку, поставте перед ним знак "–". Наприклад: "+таблиці –Excell";
- за замовчуванням програма шукає всі дані, де зустрічається введене вами слово. Наприклад, при запиті "редактор" буде знайдено слова "редактор", "текстовий", "графічний", "газети", "головний" і багато інших. Знак оклику перед або після ключового слова означає, що буде знайдено тільки слова точно відповідні запиту (наприклад, "текстовий! редактор!").

Також корисно запам'ятати і під час пошуку вдаватися до таких прийомів:

- якщо для пошуку потрібно ввести словосполучення, беріть його в лапки;

- якщо Ви пишете все слово малими буквами, буде знайдено всі варіанти його написання; якщо введено хоч би одну букву прописну, то система шукатиме тільки такі варіанти;
- якщо Ви хочете знайти не текст, а яке-небудь зображення, то можна користуватися словом image. Наприклад, image:sea дасть список сторінок із зображенням моря;
- якщо слово, яке Ви шукаєте, зустрічається в різних контекстах, можна виключити слова, які зустрічаються в непотрібному контексті. Наприклад, вказати аргумент пошуку + Celeron + Price + UA – USA;
- перевіряйте орфографію. Якщо пошук не дав результатів, можливо, при введенні Ви припустилися помилки;
- використовуйте синоніми. Якщо список знайдених сторінок дуже малий або не містить корисних сторінок, спробуйте змінити слово. Наприклад, замість "реферати", можливо, більше підійде "курсові роботи" або "твори";
- якщо один із знайдених документів ближче до теми, то клацніть ***Знайти схожі документи***. Це посилання розташовано під короткими описами знайдених документів. Система проаналізує сторінку і знайде документи, схожі на ті, що Ви вказали.

Подібних систем в *Internet* значно більше від тематичних каталогів. Серед пошукових систем існують як широкотематичні, так і вузькоспеціалізовані. Найбільш відомі з них: http://www.google.com, http://www.altavista.com, http://www.askjeeves.com, http://www.lycos.com, http://www.sciseek.com, http://www.msn.com, http://meta.ua, http://www.rambler.ru, http://www.yandex.ru, http://www.aport.ru, http://www.metabot.ru, http://newsgroups.langenberg.com, uk.wikipedia.org, www.bukinist.agava.ru.

Матеріали щодо методів підвищення ефективності пошуку інформації в *Internet* містяться у статтях: http://www.yandex.ru/info/search.html, http://www.searchengines.ru/, http://www.zodchiy.ru/links/search/, http://www.citforum.ru/internet/search/index.shtml, http://websearch.report.ru/, http://www.kokoc.com/search-engines/index.html, http://www.zhurnal.ru/search-r.shtml.

Самостійна робота має такі складові і форми їх оцінювання:

- підготовка та власне аудиторна робота на практичних і лабораторних заняттях, результати її оцінюються під час поточного контролю;

- виконання самостійних робіт у вигляді усних доповідей, підготовці есе, рефератів з конкретних проблем та складання письмових звітів на електронних або паперових носіях;
- опрацювання програмного матеріалу зі змістового модуля та оцінка її результатів під час проміжного контролю;
- виконання письмової контрольної роботи або тестування;
- звіт про проходження практики;
- звіт про науково-дослідну роботу, результати якої можуть бути використані для написання випускної роботи і за рішенням кафедри опубліковані.

**Метою** вивчення дисципліни "Технологія програмування та створення програмних продуктів" є узагальнення та розширення знань студентів з технологій проектування програмних продуктів та програмування, забезпечення їх необхідним апаратом для ґрунтовного вивчення дисциплін кваліфікаційних рівнів спеціаліста та магістра, формування у майбутніх спеціалістів уявлення про технології структурного та об'єктно-орієнтованого програмування та проектування програмних продуктів і їх застосування для розробки і доробки програмного та інформаційного забезпечення. В основу курсу покладено методи об'єктно-орієнтованого і компонентного програмування та CASE-технологія проектування програмного забезпечення.

Особлива увага приділяється методам проектування програмних систем, проектуванню інтерфейсу користувача, а також питанням якості програмного забезпечення. В курсі розглядаються основні етапи створення програмного продукту, в тому числі аналіз вимог, складання технічного завдання, проектування, кодування, тестування програмних продуктів, а також їх супроводження після встановлення у замовника.

Дисципліна "Технологія програмування та створення програмних продуктів" базується на таких дисциплінах: "Основи програмування та алгоритмічні мови", "Об'єктно-орієнтоване програмування", "Організація баз даних і знань".

Після вивчення дисципліни студент повинен

*знати*:

- методику створення структурних проектів та програм;
- методику створення об'єктно-орієнтованих проектів та програм;
- методику створення компонентних проектів та програм;
- особливості програмування в операційній системі Windows;
- етапи проектування програмних продуктів;

- CASE-технологію проектування програмного забезпечення;

*набути вмінь та навичок*:

- створювати структурні проекти та програми;
- створювати об'єктно-орієнтовані проекти та програми;
- створювати компоненти та застосовувати їх для написання програм;
- виконувати основні етапи проектування програмних продуктів;
- проектувати програмне забезпечення засобами CASE-технології;
- використовувати технологічні засоби створення програмного забезпечення.

## *ЗМІСТ САМОСТІЙНОЇ РОБОТИ*

**Змістовий модуль І. Застосування сучасних об'єктно-орієнтованих середовищ та продуктів при розробці програмних систем**

### *Тема 1. Вступ. Історичний і соціальний контекст програмування. Загальні принципи розробки програмних продуктів*

1. Історичний і соціальний аспект програмування. Програма як формалізоване описання процесу обробки даних. Технологія програмування і інформатизація суспільства.
2. Джерела помилок в програмних продуктах. Поняття правильної програми. Шляхи боротьби з помилками в програмних продуктах.
3. Проблеми проектування складних програмних продуктів. Поняття технології програмування. Життєвий цикл програмного забезпечення.
4. Етапи життєвого циклу програмного забезпечення (стандарт ISO/IEC 12207). Моделі життєвого циклу. Процеси життєвого циклу. Якість програмних продуктів. Методи боротьби із складністю програмних продуктів.

### *Тема 2. Парадигми програмування*

1. Основні концепції типізованих мов програмування: вирази, лексеми, константи, типи змінних, оператори, сфера дій декларацій,

правила видимості змінних, локальні та глобальні змінні, процедури та функції, структуровані типи даних, трансляція та інтерпретація мов програмування.
2. Низхідне проектування програм.
3. Структуроване програмування: лінійне, процедурне, модульне.
4. Абстракція даних. Об'єктно-орієнтоване, компонентне програмування.

*Література* [1; 4–6; 14; 15; 17–19; 24–26; 30; 33–35]

**Теми рефератів**

1. Написати бібліотеку функцій обробки матриць (множення матриці на матрицю, множення матриці на вектор, транспонування матриці, знаходження оберненої матриці тощо).
2. Написати бібліотеку функцій обробки комплексних чисел (виконання основних арифметичних операцій з комплексними числами).
3. Написати бібліотеку функцій обробки раціональних чисел (виконання основних арифметичних операцій з раціональними числами).
4. Скласти опис класу для об'єктів-векторів, що задаються координатами кінців у тривимірному просторі. Реалізувати методи додавання та віднімання векторів, обчислення скалярного добутку, довжини векторів, косинуса кута між векторами. Написати програму, що демонструє роботу з цим класом.
5. Скласти опис класу для об'єктів прямокутників зі сторонами, що паралельні осі координат. Реалізувати методи переміщення прямокутників на площині, зміну розмірів, побудову найменшого прямокутника, що містить два заданих прямокутника, та прямокутника, що є спільною частиною (перетином) двох прямокутників. Написати програму, що демонструє роботу з цим класом.
6. Скласти опис класу поліномів однієї змінної, що задаються ступенем полінома та масивом коефіцієнтів.
7. Реалізувати методи обчислення значення полінома для заданого аргумента, додавання, віднімання та множення поліномів, виведення на екран поліному. Написати програму, що демонструє роботу з цим класом.
8. Скласти опис класу, що забезпечує представлення матриці довільного розміру з можливістю зміни кількості рядків, стовпців,

виведення на екран під матриці довільного розміру та всієї матриці. Написати програму, що демонструє роботу з цим класом.

9. Скласти опис класу “домашня бібліотека”, що забезпечує можливість представлення довільної кількості книг, пошуку книги за будь-якою ознакою (автора, року випуску), додавання книги у бібліотеку, видалення книг з неї, сортування книг за різними полями. Написати програму, що демонструє роботу з цим класом.
10. Скласти опис класу “адресна книга”, що забезпечує можливість представлення довільної записів, пошуку запису за будь-якою ознакою (прізвищем, датою народження, номером телефону), додавання запису у книгу, видалення запису з неї, сортування записів за різними полями. Написати програму, що демонструє роботу з цим класом.
11. Скласти опис класу “студентська група”, що забезпечує можливість представлення довільної кількості студентів у групі, пошуку студента за будь-якою ознакою (прізвищем, датою народження, номером телефону), додавання студента у групу, видалення студента з неї, сортування студентів за різними полями. Написати програму, що демонструє роботу з цим класом.
12. Скласти опис класу “множина”, що забезпечує можливість виконання основних операцій з множинами (додавання і видалення елементу, перетин, об’єднання та різниця множин). Написати програму, що демонструє роботу з цим класом.
13. Скласти опис класу, що реалізує стек. Написати програму, що використовує цей клас для відшукання шляху у лабіринті.
14. Скласти опис класу “предметний покажчик”. Кожен компонент покажчика містить слово та номери сторінок, на яких це слово зустрічається. Реалізувати методи формування покажчика з клавіатури та із файла, виведення покажчика на екран та у файл, виведення номерів сторінок для заданого слова, видалення елементів покажчика. Написати програму, що демонструє роботу з цим класом.
15. Скласти опис класу, що реалізує бінарне дерево. Реалізувати методи послідовного доступу до всіх елементів, додавання нових елементів, видалення існуючих, пошуку елементу за ключем. Написати програму, що демонструє роботу з цим класом.

### Питання для самоконтролю та співбесіди

1. Охарактеризуйте поняття структурного програмування.

2. Що таке модульне програмування?
3. Що означає проектування згори вниз та знизу вгору?
4. Основні поняття об'єктно-орієнтованого програмування.
5. Основні поняття компонентного програмування
6. У чому полягає проблема раціональної структурної побудови програмних продуктів?
7. У чому полягає проблема вибору технології розробки програмного продукту?
8. Проблема стандартизації програмних продуктів
9. Як здійснюється розробка технічного завдання?
10. Як здійснюють комплексне динамічне відлагодження програмних продуктів?

**Тестові завдання**

***1. Найправильнішим варіантом визначення поліморфізму в контексті об'єктно-орієнтованого програмування є такий:***

- заміннність об'єктів з однаковим інтерфейсом;
- загальний принцип, згідно з яким знання про більш загальну категорію застосовується до більш конкретної категорії;
- принцип, що дозволяє приховувати реалізацію об'єкта від користувача;
- принцип, який дозволяє описати новий клас на основі такого, що вже існує.

***2. Інкапсуляція — це:***

- принцип, що дозволяє приховувати реалізацію об'єкта від користувача;
- принцип, який дозволяє описати новий клас на основі такого, що вже існує;
- принцип, що характеризує властивість деяких об'єктів набувати форми залежно від обставин;
- загальний принцип, згідно з яким знання про більш загальну категорію застосовується до більш конкретної категорії.

***3. Проектування на основі алгоритмічної декомпозиції називається:***

- структурним проектуванням;
- проектуванням потоків даних;
- об'єктно-орієнтованим проектуванням;
- модульним проектуванням.

***4. Не рекомендують змінювати в процесі розробки програмного продукту для підтримки динамічного компонування:***

- поля;
- інтерфейси;
- класи;
- методи.

***5. Поведінка об'єкта визначається за допомогою:***

- полів;
- властивостей;
- методів;
- функцій.

***6. З наведених висловлювань найбільш правильним є:***

- реалізація класу містить визначення операцій, що оголошені в інтерфейсі класу;
- інтерфейс реалізує методи, що оголошені в класі;
- інтерфейс визначає засоби обробки властивостей класу;
- реалізація класу містить типи даних, що оголошені в інтерфейсі класу.

***7. У процесі виконання програми можуть бути створені та знищені:***

- класи;
- об'єкти;
- інтерфейси;
- методи.

***8. Незалежність від мови програмування забезпечують:***

- бібліотеки;
- модулі;
- об'єкти;
- компоненти.

***9. Кореневим COM-інтерфейсом є такий інтерфейс:***

- IDispatch;
- IUnknown;
- ICLassFactory;
- IAccessor.

***10. Зазначте метод, який дозволяє отримати покажчик на інтерфейс:***

- QueryInterface;
- Addref;
- Release;
- Invoke.

## Змістовий модуль ІІ. Типи програмних продуктів та особливості їх розробки

### *Тема 3. Об'єктно-орієнтований підхід і діаграми класів в UML*

1. Об'єктно-орієнтований підхід. Клас. Інтерфейс. Типи і класи реалізації. Параметризовані класи (шаблони). Об'єкт. Взаємозв'язки об'єктів, класів.
2. Введення до UML. Статичні, динамічні моделі. Зображення взаємодії, зображення у вигляді кінцевого автомата, зображення діяльності, фізичне зображення. Статичне зображення моделі: класифікатори, відношення, асоціації, узагальнення, успадкування, реалізація, обмеження.
3. Діаграми класів в UML.

### *Тема 4. Модель компонентних об'єктів та компонентна технологія програмування*

1. Поняття COM. Технологія клієнт/сервер.
2. Розробка класів інтерфейсів. Інтерфейси IUnknown, IClass Factory, IDispatch. Технологія створення exe- та dll — серверів. Контейнери та елементи управління ActiveX.

*Література* [1; 2; 4; 6; 7; 9; 14–16; 19; 20; 31; 33; 36]

### Теми рефератів

1. Побудувати систему класів для описання геометричних фігур: кола, квадрата, прямокутника. Реалізувати методи для створення об'єктів, переміщення на площині, зміни розмірів та обертання на заданий кут. Написати програму, що демонструє роботу з цими класами.
2. Побудувати систему класів для описання многокутників: трикутників, чотирикутників тощо. Реалізувати методи для створення об'єктів, переміщення на площині, зміни розмірів та обертання на заданий кут. Написати програму, що демонструє роботу з цими класами.
3. Побудувати систему класів для описання різновидів трикутників: рівностороннього, рівнобедреного, прямокутного. Реалізувати методи для створення об'єктів, переміщення на площині, зміни розмірів та обертання на заданий кут. Написати програму, що демонструє роботу з цими класами.

4. Побудувати систему класів для описання різновидів чотирикутників: ромба, квадрата, паралелограма. Реалізувати методи для створення об'єктів, переміщення на площині, зміни розмірів та обертання на заданий кут. Написати програму, що демонструє роботу з цими класами.
5. Побудувати систему класів для описання банківських рахунків (чекового рахунка, ощадного рахунка, рахунка для нарахування заробітної плати). Реалізувати методи для відкриття рахунка, зняття коштів, внесення коштів, переведення коштів на інший рахунок, нарахування банківських відсотків. Написати програму, що демонструє роботу з цими класами.
6. Розробити компонент: поле введення для додатних числових значень. Реалізувати перевірку коректності введених значень. Написати програму, що демонструє роботу з цим компонентом.
7. Розробити компонент: поле введення для малих значень латиницею. Реалізувати перевірку коректності введених значень. Написати програму, що демонструє роботу з цим компонентом.
8. Розробити компонент: поле введення для малих значень кирилицею. Реалізувати перевірку коректності введених значень. Написати програму, що демонструє роботу з цим компонентом.
9. Розробити компонент: поле введення для дати. Реалізувати перевірку коректності введених значень. Написати програму, що демонструє роботу з цим компонентом.
10. Розробити компонент: поле введення за деякою маскою. Реалізувати перевірку коректності введених значень. Написати програму, що демонструє роботу з цим компонентом.
11. Розробити компонент, що дозволяє реалізувати введення графічних зображень шляхом "креслення олівцем". Написати програму, що демонструє роботу з цим компонентом.
12. Розробити компонент, що дозволяє реалізувати введення графічних зображень шляхом додавання графічних примітивів та визначення їх параметрів. Написати програму, що демонструє роботу з цим компонентом.
13. Розробити компонент, що дозволяє змінювати та відображати у формі різні примітиви фігур (Shape) і змінювати їх колір. Написати програму, що демонструє роботу з цим компонентом.
14. Розробити компонент, в якому користувач може додавати та видаляти елементи списку. Написати програму, що демонструє роботу з цим компонентом.

15. Розробити компонент, в якому користувач може обирати певний поточний каталог. Написати програму, що демонструє роботу з цим компонентом.

### Питання для самоконтролю та співбесіди

1. Для чого потрібно розподіляти систему, що проектується, на окремі компоненти?
2. Які основні прикмети мови моделювання UML?
3. Побудуйте діаграму варіантів використання для моделі банкомату.
4. Яка послідовність дій рекомендована при розробці діаграми варіантів?
5. Побудуйте діаграму кооперації для вибору адреси клієнту з метою відправлення електронного листа.
6. Побудуйте діаграму варіантів використання для системи продажу товарів за каталогом.
7. Переваги компонентно-орієнтованого програмування?
8. Що таке компонент?
9. Для чого призначений інтерфейс IUnknown?
10. Для чого призначений інтерфейс IClassFactory?

### Тестові завдання

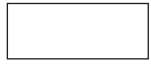
***1. Мова UML призначена для:***

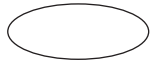
- об'єднання нотацій ERD, IDEF0, DFD;
- уніфікації нотацій ERD, IDEF0, DFD;
- генерації програмного коду на основі моделей нотацій ERD, IDEF0, DFD;
- моделювання бізнес-процесів.

***2. З наведених діаграм до канонічних у мові UML належать:***

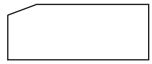
- діаграма варіантів використання;
- діаграма DFD;
- діаграма послідовності;
- діаграма таблиць.

***3. Пакет у нотації мови UML зображують так:***

- 
- 

- 
- 

***4. Зазначте, яке зображення використовується для позначення варіанта використання на діаграмі варіантів використання:***

- 
- 
- 
- 

***5. Відношення включення на діаграмі варіантів використання відображається так:***

- 

- 

- 
- 

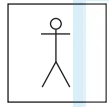
***6. Відношення включення зв’язує:***

- актора з окремим варіантом використання;
- два варіанта використання;
- два актора;
- діаграми варіантів використання для відображення вкладеності цих діаграм.

***7. Асоціація на діаграмі варіантів використання пов’язує:***

- окремого актора з варіантом використання;
- окремих акторів між собою;
- окремі варіанти використання;
- діаграми варіантів використання.

***8. Бізнес-актора на діаграмі варіантів відображають так:***

- 
- 
- 
- 

***9. Діаграма варіантів використання — це діаграма, на якій відображаються:***

- відношення між акторами та варіантами використання;
- функції системи, що моделюються;
- функціональні вимоги до програмного продукту;
- проектні обмеження та вимоги управління програмним продуктом.

***10. Відношення розширення пов'язує:***

- окремого актора з окремим варіантом використання;
- окремих акторів між собою;
- два варіанта використання;
- діаграми варіантів використання.

***11. Сценарії на діаграмі варіантів використання відображаються у формі:***

- приміток;
- додаткових текстових документів;
- базових варіантів використання;
- асоціацій зі стереотипом.

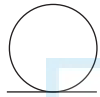
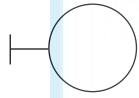
***12. Процес розробки діаграми варіантів використання починається з визначення:***

- головних та другорядних акторів;
- базових класів системи, що моделюється;
- відношення включення та розширення;
- вимог до програмного продукту.

***13. Квантор видимості “захищений” (protected) на діаграмі класів відображається так:***

- +
- #
- –
- ~

***14. Клас-сутність (entity class) на діаграмі класів відображається так:***

- 
- 
- 
- 

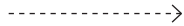
***15. На діаграмі класів є правильними такі записи операцій:***

- + відобразити (форма : Еліпс);
- # setAddress (inout Сотрудник) = String;
- поповнити рахунок клієнта (номер рахунка : Integer);
- # Message ( ) :=‘Ошибка деления на ноль’.

***16. На діаграмі класів є правильними такі записи кратності кінця асоціації:***

- 1..*;
- 1: n;
- 0..1;
- (0 : 1).

***17. Відношення агрегації (aggregation) на діаграмі класів відображається так:***

- 
- 
- 
- 

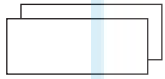
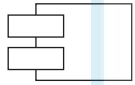
***18. Зазначте, яке обмеження означає, що в даному відношенні специфіковані усі класи-нащадки та інших класів нащадків у цього класу бути не може:***

- {disjoint};
- {overlapping};
- {complete};
- {incomplete}.

***19. Зазначте, яке обмеження означає, що в даному відношенні узагальнення окремі екземпляри класів-нащадків можуть належати одночасно до кількох класів:***

- {disjoint};
- {overlapping};
- {complete};
- {incomplete}.

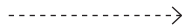
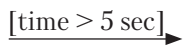
***20. Мультиоб'єкт (multiobject) на діаграмі кооперації відображається так:***

- 
- 
- 
- 

***21. Зазначте стереотип, що позначає посилання іншому об'єкту певного сигналу, який асинхронно ініціюється одним об'єктом та приймається іншим:***

- <<call>>;
- <<return>>;
- <<create>>;
- <<send>>;
- <<destroy>>.

***22. Повернення з виклику процедури на діаграмі послідовності відображається так:***

- 
- 

- 
- 

**Змістовий модуль ІІІ. Прогресивні технології розробки програмних продуктів**

***Тема 5. Зовнішній опис програмного продукту***

1. Поняття та призначення зовнішнього опису програмного продукту. Визначення вимог користувача до програмного продукту. Специфікація якості програмного продукту. Функціональна специфікація програмного продукту.
2. Методи специфікації семантики функцій: метод таблиць рішень, операційна семантика, денотаційна семантика, аксіоматична семантика. Мови специфікацій.

***Тема 6. Поняття архітектури програмного продукту. Низхідний підхід до розробки програмного продукту***

1. Класи архітектур програмних продуктів. Метод декомпозиції програмних продуктів. HIPO діаграми програмних продуктів. Розробка структури програми. Методи розробки структури програми та контроль її правильності.
2. Розробка програмного модуля. Структурне програмування. Покрокова деталізація. Контроль програмного модуля. Метод головного програміста. Програмні засоби тестування.

***Тема 7. Інформаційні моделі і системи. Управління програмними проектами***

1. Принципи організації і схема проектування; роль і місце інструментальних засобів; системи керування базами даних: структура, маніпуляція, цілісність даних. Інструментальні засоби проектування концептуальної моделі даних.
2. Генерація застосувань: принципи генерації застосувань, інструментальні засоби генерації застосувань.
3. Управління програмними проектами: управління групою розробників; планування графіка проекту; методи оцінювання програм-

ного продукту, аналіз ризиків; забезпечення якості, управління конфігурацією програмного продукту.

### *Тема 8. Забезпечення функціональності та надійності програмного продукту*

1. Функціональність та надійність програмного продукту. Забезпечення точності, автономності, стійкості, захищеності програмного продукту. Забезпечення ефективності, мобільності, якості.

### *Тема 9. Документування програмних засобів*

1. Види документації, що створюється та використовується під час розробки програмних продуктів: технічне завдання, технічний, робочий проекти. Документація користувача. Документація з супроводження програмного продукту.

### *Тема 10. CASE-технологія розробки програмних продуктів*

1. Поняття CASE-технології та її можливості. Концептуальні основи CASE-технології.
2. Характеристика сучасних CASE-засобів. Класифікація CASE-засобів. Огляд сучасних CASE-засобів та сфера їх застосування.
3. Проектування діаграм потоків даних, контекстних діаграм, діаграм "сутність-зв'язок", діаграм переходів станів, структурних карт.

*Література* [1; 3; 4; 6; 8; 10–15; 21–23; 25–29; 32; 34]

### Теми рефератів

1. Процес розробки системи класів для описання геометричних фігур: кола, квадрата, прямокутника за допомогою певного CASE-засобу.
2. Процес розробки системи класів для описання многокутників: трикутників, чотирикутників тощо за допомогою певного CASE-засобу.
3. Процес розробки системи класів для описання різновидів трикутників: рівностороннього, рівнобедреного, прямокутного за допомогою певного CASE-засобу.
4. Процес розробки системи класів для описання різновидів чотирикутників: ромба, квадрата, паралелограма за допомогою певного CASE-засобу.

5. Процес розробки системи класів для описання банківських рахунків (чекового рахунка, ощадного рахунка, рахунок для нарахування заробітної плати) за допомогою певного CASE-засобу.
6. Документування компонента: поле введення для додатних числових значень.
7. Документування компонента: поле введення для малих значень латиницею.
8. Документування компонента: поле введення для малих значень кирилицею.
9. Документування компонента: поле введення для дати.
10. Документування компонента: поле введення за певною маскою.

**Питання для самоконтролю та співбесіди**

1. Як визначається надійність програмних продуктів?
2. Які переваги дає об'єктно-орієнтоване програмування порівняно з таким, що не використовує об'єкти?
3. Поняття життєвого циклу програмного продукту
4. Що таке зовнішній опис програмного продукту?
5. Як здійснюється супроводження програмного продукту?
6. Які існують методи контролю якості програмного продукту?
7. Навіщо потрібні інтегральні середовища розробки?
8. Дати визначення програмного модуля?
9. У чому полягає процес керування розробкою програмного продукту?
10. Що таке діаграма стану класу?
11. Опишіть роль і місце мови HTML при розробці Web-орієнтованої системи.
12. Опишіть роль і місце технології ASP.Net при розробці Web-орієнтованої системи.
13. Які є проблеми захисту інформації у Web-орієнтованих системах і як вони вирішуються?
14. Що таке CASE-засоби для розробки програмних систем?
15. Які є сучасні CASE-засоби для розробки програмних систем і на яких технологіях вони базуються?
16. Що таке реінжиніринг бізнес-процесів?
17. Охарактеризуйте основні функції BPwin.
18. Як представляється функціональна модель діяльності в методології IDEF0?
19. Що таке CASE-технології?

20. Назвіть переваги та недоліки CASE-технології.
21. За якими критеріями класифікують CASE-засоби?
22. Які компоненти мають входити до повного комплексу CASE-засобів, що забезпечують підтримку життєвого циклу програмного продукту?
23. Здійсніть порівняльний аналіз традиційної технології розробки та розробки засобами CASE-технології.
24. Назвіть основні об'єкти діаграм функціональної моделі за методологією IDEF0.
25. Що означають роботи в діаграмах функціональної моделі, як вони відображаються за методологією IDEF0?
26. Призначення стрілок в діаграмах функціональної моделі за методологією IDEF0.
27. Типи стрілок в діаграмах функціональної моделі за методологією IDEF0.
28. Для чого призначено словник стрілок у діаграмах функціональної моделі за методологією IDEF0.
29. Що описують за допомогою DFD-діаграм?
30. Назвіть особливості DFD-діаграм

**Тестові завдання**

***1. До CASE-засобів належать:***

- ERwin+BPwin;
- Rational Application Developer;
- CASE.Аналітик;
- Borland Delphi.

***2. Rational Rose. Нотація IDEF0 реалізована в такій системі:***

- AllFusion Process ModelerT;
- IBM Rational Rose 2003T;
- MS Project;
- Rational Application Developer.

***3. Нотацію мови UML підтримують такі з названих засобів:***

- AllFusion Process Modeler;
- IBM Rational Rose;
- ArgoUML;
- Rational Application Developer.

***4. Діаграми IDEF0 призначені для:***

- розробки статичної моделі предметної сфери інформаційної системи;

- розробки конструкторської документації;
- моделювання бізнес-процесів;
- документування програмного продукту.

***5. До компонентів CASE-засобів не належать:***

- засоби генерації програмного коду на основі моделей нотацій ERD, DFD;
- репозиторій;
- засоби документування;
- база даних.

***6. Згідно з класифікацією вимог FURPS+ до програмних систем висуваються такі категорії вимог:***

- вимоги на апаратне забезпечення;
- функціональні вимоги;
- ергономічні вимоги;
- вимоги продуктивності.

***7. Зазначте додаткові вимоги, які висуваються до програмних систем згідно з класифікацією вимог FURPS+:***

- ергономічні вимоги;
- вимоги до графічного інтерфейсу користувача;
- фізичні вимоги;
- вимоги безпеки.

***8. Вершиною деревоподібної структури діаграм у IDEF0 є діаграма такого типу:***

- контекстна діаграма;
- діаграма декомпозиції;
- діаграма дерева вузлів;
- діаграма експозиції.

***9. ERwin використовується для:***

- проектування схеми бази даних та генерації її описання мовою цільової СУБД;
- моделювання бізнес-процесів;
- тестування програмних продуктів;
- реінжинірингу існуючих баз даних.

***10. Для опису документообігу використовуються:***

- діаграми потоків даних;
- контекстні діаграми;
- діаграми послідовності;
- діаграми таблиць.

## *СПИСОК ЛІТЕРАТУРИ*

*Основна*


1. *Буч Г.* Объектно-ориентированный анализ и проектирование: с примерами приложений на C++. — М.: Бином; Невский диалект, 1998.
2. *Буч Г., Рамбо Д., Джекобсон А.* Язык UML. Руководство пользователя: Пер. с англ. — М.: ДМК, 2000.
3. *Вендров А. М.* CASE-технологии. Современные методы и средства проектирования информационных систем. — М., 1997. — 346 с.
4. *Дробушевич Л. Ф.* Технология программирования: Моделирование программных систем в Rational Rose: Учеб.-метод. пособие. — Минск.: БГУ, 2002.
5. *Жоголев Е. А.* Введение в технологию программирования. — М.: "ДИАЛОГ-МГУ", 1994.
6. *Зиглер К.* Методы проектирования программных систем. — М.: Мир, 1985.
7. *Ларман Крэг.* Применение UML и шаблонов проектирования: Пер. с англ. — 2-е изд. — М.: Вильямс, 2002.
8. *Маклаков С. В.* BPwin и ERwin. CASE-средства разработки информационных систем. — М.: ДИАЛОГ-МИФИ, 1999. — 256 с.
9. *Роджерсон Д.* Основы COM. — М.: Рус. ред. , 1997. — 370 с.
10. *Федотова Д. Э., Семенов Ю. Д., Чижик К. Н.* CASE-технологии: Практикум. — М., 2005. — 160 с.


*Додаткова*


11. *Агафонов В. Н.* Спецификация программ: понятийные средства и их организация. — Новосибирск: Наука, 1987.
12. *Бейзер Б.* Тестирование черного ящика. Технология функционального тестирования. — СПб.: Питер, 2004.
13. *Боэм Б., Браун Дж., Каспар Х.* и др. Характеристики качества программного обеспечения. — М.: Мир, 1981.
14. *Брауде Э.* Технология разработки программного обеспечения. — СПб.: Питер, 2004.
15. *Брукс Ф.П.*, мл. Как проектируются и создаются программные комплексы / Пер. с англ. А. П. Ершова. — М.: Наука, 1979.
16. *Буч Г., Рамбо Д., Джекобсон А.* UML. Руководство пользователя. — М.: ДМК, 2000.
17. *Дал У., Дейстра Э., Хоор К.* Структурное программирование. — М., 1975. — 245 с.

18. *Дейкстра Э.* Заметки по структурному программированию // У. Дал, Э. Дейкстра, К. Хоор. Структурное программирование. — М.: Мир, 1975. — С. 7–97.
19. *Дробушевич Л. Ф.* Технология программирования: Моделирование программных систем: Метод. указания и задания к лабораторным работам. — Минск.: БГУ, 2003.
20. *Дэвид Чеппел.* Технологии ActiveX и OLE: Пер. с англ. — М.: Рус. ред., 1997.
21. *Калянов Г. Н.* CASE структурный системный анализ. — М.: ЛОРИ, 1996.
22. *Канер Сэм и др.* Тестирование программного обеспечения: Пер. с англ. — К.: Изд-во "ДиаСофт", 2000.
23. *Кантор М.* Управление программными проектами: Пер. с англ. — М.: Вильямс, 2002.
24. *Кауфман В. Ш.* Языки программирования. Концепции и принципы. — М.: Радио и связь, 1993.
25. *Кинг Д.* Создание эффективного ПО. — М.: Мир, 1991.
26. *Константайн Л.* Конструирование программ. — М.: Мир, 1991.
27. *Липаев В. В.* Качество программного обеспечения. — М.: Финансы и статистика, 1983.
28. *Майерс Г.* Искусство тестирования программ. — М.: Финансы и статистика, 1982.
29. *Майерс Г.* Надежность программного обеспечения. — М.: Мир, 1980.
30. *Павловская Т. А., Щупак Ю. А.* C/C++. Структурное программирование: Практикум. — СПб., 2003. — 240 с.
31. *Рофейл Э, Шохауд Я.* COM и COM+. — К.: Век+, 2000. — 560 с.
32. *Требования* и спецификации в разработке программ: Пер. с англ. — М.: Мир, 1984.
33. *Турский В.* Методология программирования. — М.: Мир, 1981.
34. *Фокс Дж.* Программное обеспечение и его разработка. — М.: Мир, 1985.
35. *Хьюз Дж., Мичтом Дж.* Структурный подход к программированию. — М.: Мир, 1980.
36. *Шлеер С., Меллор С.* Объектно-ориентированный анализ: моделирование мира в состояниях. — К.: ДИАЛЕКТИКА, 1994.

## *ЗМІСТ*



Відповідальний за випуск *А. Д. Вегеренко*
Редактор *О. М. Коваленко*
Комп'ютерне верстання *О. А. Залужна*

Зам. № ВКЦ-3568
Міжрегіональна Академія управління персоналом (МАУП)
03039 Київ-39, вул. Фрометівська, 2, МАУП