МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

Робоча програма

навчальної дисципліни

"ОСНОВИ ПРОГРАМУВАННЯ ТА АЛГОРИТМІЧНІ МОВИ"

для студентів напряму підготовки "Комп'ютерні науки"

усіх форм навчання

Харків. Вид. ХНЕУ, 2008

Затверджено на засіданні кафедри информаційних систем.
Протокол №1 від 28.08.2007 р.

Р78 Робоча програма навчальної дисципліни "Основи програмування та алгоритмічні мови" для студентів напряму підготовки "Комп'ютерні науки" усіх форм навчання / Укл. М. Ю. Лосєв, Ю. Е. Парфьонов, В. М. Федорченко, О. В Щербакова. — Харків: Вид. ХНЕУ, 2008. — 44 с. (Укр. мов.)

Наведено програму навчальної дисципліни "Основи програмування і алгоритмічні мови".

Подано кваліфікаційні вимоги до студентів в галузі інформаційних систем в промисловості і у банківській сфері; тематичний план навчальної дисципліни; зміст дисципліни за модулями та темами; плани лекцій; плани практичних та лабораторних занять; самостійна робота студента; індивідуально-консультативна робота; методики активізації процесу навчання; система поточного та підсумкового контролю знань студентів.

Рекомендовано для студентів спеціальності напрямку підготовки "Комп'ютерні науки".

Вступ

Навчальну дисципліну "Основи програмування і алгоритмічні мови" віднесено до групи освітньо-професійних дисциплін підготовки бакалаврів за спеціалізаціями – "Інформаційні управляючі системи і технології", "Комп'ютерний еколого-економічний моніторинг".

Сьогоднішні умови господарювання вимагають від фахівців з економічного управління всебічного використання новітніх інформаційних технологій. Широкі можливості комп'ютеризованих засобів у питаннях збору, обробки та видачі необхідної інформації здатні значно підвищити якість економічних розрахунків, зробити більш ефективним процес обґрунтування економічних рішень.

Навчальна дисципліна "Основи програмування та алгоритмічні мови" є інструментальною основою для виконання аналітичної частини подальших спецкурсів, а також курсових і дипломних робіт. Вона забезпечує наступні дисципліни: "Об'єктно-орієнтоване програмування", "Системне програмування і операційні системи", "Організація баз даних і знань", "Основи автоматизованого проектування складних об'єктів і систем".

Мета навчальної дисципліни – викладення основних понять алгоритмізації і техніки застосування у програмуванні базових алгоритмічних структур (організація програм) і базових структур даних (організація даних).

Для досягнення мети поставлені такі ***основні завдання***:

вивчення основних етапів процесу проектування програмного забезпечення і визначення принципів процедурного програмування щодо розробки програм мовою C++ ;

вивчення типових підходів до розробки і аналізу найбільш розповсюджених алгоритмів рішення економіко-математичних задач;

здійснення аналізу можливостей сучасних інструментальних середовищ розробки програм (на прикладі середовища Visual C++.NET);

визначення концепцій і вивчення основних принципів організації програм у середовищі ОС Windows.

Предметом навчальної дисципліни є теорія і практика застосування у програмуванні базових алгоритмічних структур і базових структур даних на базі сучасних технологій розробки програмного забезпечення.

Структура робочої програми навчальної дисципліни "Основи програмування та алгоритми мови" наведена в табл.1.

Таблиця 1

Структура програми навчальної дисципліни

Навчальна дисципліна: підготовка бакалаврів	Напрям, спеціальність, освітньо-кваліфікаційний рівень	Характеристика навчальної дисципліни
Кількість кредитів відповідних ECTS – 9, у тому числі: змістовних модулів – 2, самостійна робота; індивідуальне науково-дослідне завдання (ІНДЗ)	Галузь знань 0501 "Інформатика та обчислова техніка"	Обов'язкова. Рік підготовки: 1. Семестр: 1,2
Кількість годин за змістовними модулями: модуль 1 – 162, модуль 2 – 162 Всього –324 години	Напрям підготовки "Комп'ютерні науки"	Лекції (теоретична підготовка) – 72 години. Практичні заняття – 36 годин Лабораторні заняття – 72 години. Самостійна робота – 90 годин. ІНДЗ – 54 годин
кількість тижнів викладення навчальної дисципліни: 36. Кількість годин за тиждень 5	Освітньо-кваліфікаційний рівень: бакалавр	Вид контролю: залік / іспит

У процесі навчання студенти отримують необхідні знання під час проведення аудиторних занять: лекційних, практичних та лабораторних. Велике значення в процесі вивчення та закріплення знань має самостійна робота студентів і виконання ІНДЗ. Усі ці види занять розроблені відповідно до кредитно-модульної системи організації навчального процесу.

1. Кваліфікаційні вимоги до студентів у галузі інформаційних управляючих систем і технологій

Дисципліна "Основи програмування і алгоритмічні мови" є базовою для бакалаврів напряму підготовки "Комп'ютерні науки".

Необхідна навчальна база перед початком вивчення дисципліни: для успішного вивчення дисципліни необхідні базові знання, отримані студентами в об'ємі шкільної програми, а також поточні знання при паралельному освоєнні ними дисциплін "Схемотехніка ЕОМ", "Введення в спеціальність", "ЕОМ і мікропроцесорні системи", "Вища математика", "Основи дискретної математики".

У результаті вивчення навчальної дисципліни студенти повинні:

знати:

основні етапи процесу проектування програмного забезпечення;

типові алгоритмічні конструкції;

принципи процедурного і структурованого програмування;

особливості застосування сучасних базових інструментальних програмних засобів, призначених для вирішення економічних задач;

базові типи даних;

похідні типи даних: переліки, покажчики, посилання, масиви, структури, об'єднання;

оператори управління програмою;

команди передпроцесорної обробки;

правила роботи з функціями;

систему уведення-виведення С++;

основні принципи роботи з файлами;

правила роботи із шаблонами;

принципи розробки Windows-додатків;

основу побудови програм на керованому С++;

вміти:

складати програми мовою С++, *забезпечуючи*:

рішення задач з курсу вищої математики (чисельне диференціювання і інтеграція, рішення рівнянь і т.д.);

створення і обробку структур, масивів структур;

найпростішу обробку файлів;

використовування функцій;

використовування основних елементів призначеного для користувача інтерфейсу ОС Windows;

використовування сучасного інструментального програмного забезпечення;

користуватися раніше складеними програмами і здійснювати супровід програм, вносити зміни в програму, виконувати відладку програм за допомогою вбудованих інструментальних засобів.

2. Тематичний план навчальної дисципліни

З самого початку вивчення дисципліни кожен студент має бути ознайомлений як з програмою дисципліни і формами організації навчання, так і зі структурою, змістом та обсягом кожного з її навчальних модулів, а також з усіма видами контролю та методикою оцінювання знань. Тематичний план дисципліни складається з двох змістових модулів, кожний з яких об'єднує в собі відносно окремий самостійний блок дисципліни, який логічно пов'язує кілька навчальних елементів дисципліни за змістом та взаємозв'язками (табл.2).

Таблиця 2

Структура залікового кредиту навчальної дисципліни

Тема	Кількість годин			
	Лекції	Практичні/ лабораторні заняття	Індивідуальна робота	Самостійна робота
Модуль 1. Введення в розробку і кодування алгоритмів				
Тема 1. Алгоритм як основне поняття програмування. Лексичні основи мов високого рівня	4	2/6	4	5
Тема 2. Алгоритмічна мова С++. Основні типи даних	4	2/6	4	8
Тема 3. Програмування обчислювальних процесів. Оператори управління програмою	4	4/6	4	8
Тема 4. Функції	6	4/6	4	8
Тема 5. Похідні типи даних. Рядки	12	6/12+4*	4	8
Тема 6. Введення в систему вводу-виводу С++	6	2*/4*	7	8
Усього за модулем	36	18/36	27	45
Модуль 2. Принципи розробки Windows-додатків				
Тема 7. Передпроцесорна обробка	4	2/4	4	8
Тема 8. Структури та об'єднання	8	4/4	4	8
Тема 9. Шаблони	6	2/4	4	5
Тема 10 Програмування в середовищі Windows	12	6/10	6	12
Тема 11. Програмування на керованому С++	6	2/6	9	12
Усього за модулем	36	18/36	27	45
Усього	72	36/72	54	90

*** – виконується у весняному семестрі**

3. Зміст навчальної дисципліни за модулями та темами

Модуль 1. Введення в розробку і кодування алгоритмів

Тема 1. Алгоритм як основне поняття програмування. Лексичні основи мов високого рівнч

Поняття алгоритму. Властивості алгоритму. Типові алгоритмічні конструкції. Розробка алгоритму методом покрокового уточнення. Алгоритмічні конструкції: послідовність, вибір, повторення. Способи завдання алгоритму. Критерії оцінки алгоритмів.

Початкові відомості про технологію програмування: процедурне, структуроване і об'єктно-орієнтоване програмування. Мови програмування: процедурні, апликативні, системи правил, об'єктно-орієнтовні. Стандартизація мов та середовища проектування. Транслятори. Редактори. Компонувальники. Відладчики. Керуючі структури: оператори, вирази та підпрограми. Огляд сучасних інтегрованих систем програмування. Інтегроване середовище системи програмування Visual C++.NET. Платформа DOT.NET. Етапи розробки та впровадження програм. Вимоги до програмного коду.

Тема 2. Алгоритмічна мова C++. Основні типи даних

Стандарт ANSI. Структура C++ програми.

Лексичні елементи мови C++: алфавіт, коментарі, ідентифікатори, службові слова, дані, вираз, операнд, змінна, операція. Домовленості про імена.

Поняття типу даних. Класифікація і представлення даних. Базові типи даних: логічний, символьний, цілий, речовинний. Перетворення типів: неявні перетворення, явні перетворення.

Пріоритети операції. Зведена таблиця пріоритетності і асоціативності операцій.

Операції. Унарні операції: унарний мінус, унарний плюс, порозрядне інвертування, логічне заперечення, інкремент, декремент, операція обчислення розміру (sizeof). Бінарні операції: аддитивні, мультиплікативні, зсувів, порозрядні, операції відносин, логічні, привласнення, операція "кома".

Пріоритети операції. Зведена таблиця пріоритетності і асоціативності операцій.

Стандартні математичні функції.

Константні величини: цілі, речовинні, перечислювальні, символьні (літерні), рядкові (рядки або літерні рядки). Правила визначення компілятором констант. Визначення констант за допомогою ключового слова const. Константи переліків.

Тема 3. Програмування обчислювальних процесів. Оператори управління програмою

Загальні відомості про систему вводу-виводу даних.

Вирази, символи пропусків, блоки і комплексні вирази. Операнд, змінна. Оператор привласнення. Оголошення та ініціалізація змінних.

Типи операторів. Найпростіший оператор, оператор-оголошення, оператор-визначення, оператор-вираз. Управляючі оператори: оператори проходження; оператори вибору (єдиний вибір – if, подвійний вибір – if / else, множинний вибір – switch, умовна операція); оператори повторення (оператор while, оператор do-while, оператор for). Вкладені цикли. Управляючі оператори в циклах: оператор break, оператор continue, оператор goto. Рекомендації по вибору циклів.

Тема 4. Функції

Загальні відомості про функції. Структура функції. Значення, параметри і аргументи, що повертаються. Оголошення функції. Прототипи функцій. Визначення функції. Виконання функції.

Локальні і глобальні змінні. Правило видимості змінних. Приведення типів аргументів функцій. Правила автоматичного (неявного) приведення типів. Явні перетворення типів. Правила роботи з функціями. Класи пам’яті.

Список параметрів функції. Параметри за умовчанням. Способи передачі параметрів. Способи повернення значення. Функції, що підставляються.

Створення власних заголовних файлів.

Перевантаження функцій. Рекурсія.

Робота функцій. Розбиття пам’яті. Стек і функції.

Модифікатори функцій.

Тема 5. Похідні типи даних. Рядки

Масиви. Оголошення масивів. Ініціалізація масивів. Обробка одновимірних масивів даних економічного характеру. Алгоритми сортування масивів. Багатовимірні масиви. Ініціалізація багатовимірного масиву. Типові приклади обробки матриць.

Масиви як параметри функцій.

Рядки як масиви символів. Операції з рядками. Тип даних string. Ввід-вивід рядків.

Поняття покажчика, посилання. Покажчики і масиви. Адресна арифметика. Посилання. Приклади використовування покажчиків і посилань.

Покажчики на функції. Посилання. Параметри функцій як посилання.

Організація пам’яті в сучасних процесорах і покажчики мови С++. Моделі пам’яті. Статичні і динамічні змінні. Оператори new і delete. Динамічні масиви. Динамічні масиви як параметри функцій. Зв’язні списки. Створення однозв’язного списку. Прохід однозв’язного списку. Включення нового елемента в існуючий список. Виключення елемента із списку.

Тема 6. Введення в систему вводу-виводу С++

Базові положення системи вводу-виводу С++. Потоки і буфери. Стандартні об’єкти вводу-виводу.

Ввід даних за допомогою глобального об’єкта cin.

Вивід рядків. Введення одного символу. Використовування функції get(): без параметрів, з параметрами. Ввід рядків із стандартного пристрою введення. Використовування функції getline(). Вивід даних за допомогою глобального об’єкта cout. Очищення буфера виводу – flush(). Використовування функцій put() і write(). Ввід-вивід даних, що форматується. Маніпулятори вводу-виводу. Функції width(), precision(), fill(). Маніпулятори, що визначаються користувачем.

Використовування файлів для вводу-виводу даних. Створення файла. Створення потоку. Відкриття потоку. "Приєднання" файла до потоку. Обміни з файлом за допомогою потоку. "Від’єднання" потоку від файлу. Закриття файла. Знищення файла.

Модуль 2. Принципи розробки windows-додатків

Тема 7. Передпроцесорна обробка

Основи апарату макросів. Директиви препроцесора.

Директива препроцесора #include і файли, що включаються. Директива препроцесора #define: оголошення констант і макросів. Умовна компіляція. Використовування ключового слова typedef із структурами. Відмінність директиви #define від оператора typedef. Оператор defined.

Тема 8. Структури та об'єднання

Структури. Структури з бітовими полями. Вкладені структури. Доступ до елементів структур. Операції з структурами. Структури як параметри функцій.

Масиви структур. Покажчики на структури. Передача з посилання масивів структур.

Об'єднання. Операції з об'єднаннями. Переліки.

Функції роботи з датою та часом.

Тема 9. Шаблони

Основи апарату шаблонів. Шаблони функцій. Перевантаження шаблонів функцій. Шаблони функцій сортування.

Стандартна бібліотека шаблонів (STL). Призначення та склад STL. Контейнери. Робота з векторами, списками, стеками, чергами.

Тема 10. Програмування в середовищі Windows

Компоненти та завдання операційної системи (ОС). Допоміжні модулі операційної системи. Характеристики ОС. Інтерфейс прикладного програмування (API). Утиліти, системні програми, які оброблюють бібліотеки процедур.

Особливості ОС Windows. Графічні елементи вікна Windows. Елементи призначеного для користувача інтерфейсу Windows. Поняття повідомлення (формат повідомлення). Джерела отримання повідомлень в Windows-програмах. Цикл обробки повідомлень. Графічна схема типової Windows-програми.

Каркасний Windows-додаток. Клас вікон. Головна функція. Створення вікна. Створення вікон з використанням існуючих класів. Визначення структури WNDCLASS. Реєстрація класу вікон. Обробка повідомлень у віконній функції. Функції підтримки вікон. Приєднання даних класу вікна до вікна. Зміна зовнішнього вигляду вікна.

Головна функція додатка. Типи даних, які використовуються в Windows. Цикл обробки повідомлень. Джерела повідомлень. Рентабельні функції. Функції перехвату повідомлень. Черги повідомлень.

Органи управління. Редактори тексту. Смуги прокрутки.

Ресурси Windows-додатків. Структура файлів ресурсу. Підключення ресурсів до виконавчого файлу. Таблиці рядків. Ресурси, які визначаються користувачем. Створення меню. Головне меню додатка. Спливаюче (контекстне) меню додатка. Повідомлення меню. Системне меню додатка. Меню без використання ресурсів. Додавання меню до вікна. Повідомлення меню.

Акселератори. Опис акселераторів у файлах ресурсів. Підключення таблиць акселераторів.

Діалогові вікна та їх елементи. Модальні та немодальні діалогові вікна. Клавіатурний інтерфейс діалогового вікна. Динамічні діалогові вікна. Шаблони діалогового вікна.

Елементи управління діалогових вікон. Створення діалогових вікон. Класи передвизначених вікон. Списки прості та комбіновані. Нотифікаційні повідомлення та їх обробка. Стандартні діалогові вікна. Стандартні діалогові панелі. Панелі для відкриття або збереження файлів. Панелі вибору кольору. Панелі для вибору шрифтів.

Структура багатовіконного інтерфейсу. Головне вікно. Складання багатовіконного додатка. Дочірні вікна. Порядок взаємодії між вікнами. Додаткова інформація про обробку повідомлень в головному вікні.

Визволення забраних ресурсів. Особливості віконної процедури. Меню в багатовіконному додатку. Режими огляду та сортування.

Загальні відомості про контексти пристроїв. Концепція GDI. Інтерфейс графічних пристроїв. Ввід об’єктів до контексту пристроїв. Растрові зображення. Функції роботи з растровими зображеннями. Кольори та бітові образи. Формат монохромного бітового образу. Формат кольорового бітового образу. Кольорові профілі. Перетворення кольорів.

Піктограми. Створення піктограм. Функції роботи з піктограмами.

Алгоритми рисування в Windows. Інструменти рисування. Перо, кість, палітра та області для рисування. Області виводу. Визначення області недійсної, відсікання та повідомлення WM_PAINT.

Режими рисування. Рисування базових графічних елементів.

Тема 11. Програмування на керованому C++

Поняття складки (Assembly). Відображення C++ на специфікацію CLS. Типи даних C++ і CLR. Директива #using та оператор using. Стандартний ввод-вивід. Керовані та некеровані типи. Типова безпека. Управління прибиранням сміття. Створення керованого коду, прапор компіляції /clr.

4. Плани лекцій

Лекційні заняття проводяться в межах єдиної технології викладання комп'ютерних дисциплін в ХНЭУ, що практикується на кафедрі Комп'ютерних систем і технологій. Лекції читаються в аудиторіях, що мають достатню кількість місць, зручні столи, нормальну акустику, дошку і можливість підключення ТСН. На лекції виноситься найбільш складний матеріал, і матеріал, недостатньо повно відображений в підручниках.

Змістовний модуль 1. Введення в розробку і кодування алгоритмів

Тема 1. Алгоритм як основне поняття програмування. Лекційні основи мов високого рівня

Лекція 1. **Вступ. Алгоритм як основне поняття програмування**

1.1. Вступ до дисципліни.

1.2. Етапи розробки та виконання програм на ЕОМ.

1.3. Поняття алгоритму, властивості алгоритму.

1.4. Типові алгоритмічні конструкції.

1.5. Розробка алгоритму методом покрокового уточнення.

Література: основна [1; 3; 8]; додаткова [13; 17].

Лекція 2. **Основи розробки програм**

2.1. Основні поняття платформи Microsoft .Net.

2.2. Культура програмування.

2.3. Лексичні основи мов високого рівня.

Література: основна [1; 3; 8]; додаткова [13; 17].

Тема 2. Алгоритмічна мова С++. Основні типи даних

Лекція 3. **Введення в С++**

3.1. Лексичні елементи мови С++. Операнд, змінна. Оператор привласнення. Оголошення та ініціалізація змінних.

3.2. Поняття типу даних. Класифікація і представлення даних. Базові типи даних. Перетворення типів.

3.3. Константні величини. Операції. Пріоритети операцій.

3.4. Структура програми С++. Класи пам'яті.

Література: основна [1; 3; 8]; додаткова [13; 17].

Тема 3. Програмування обчислювальних процесів. Оператори управління програмою

Лекція 4. **Програмування обчислювальних процесів**

4.1. Загальні відомості про систему вводу-виводу даних.

4.2. Оператор привласнення. Стандартні функції.

4.3. Умовний оператор.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

Лекція 5. **Програмування обчислювальних процесів (продовження)**

5.1. Оператор вибору варіанту.

5.2. Оператор while.

5.3. Оператор do-while.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

Лекція 6. **Програмування обчислювальних процесів (продовження)**

6.1. Оператор вибору варіанту.

6.2. Циклічні оператори for.

6.3. Оператори переходу (виходу).

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

Тема 4. Функції

***Лекція 7*. Функції**

7.1. Загальні відомості про функції. Заголовні файли.

7.2. Оголошення функції. Структура функції.

7.3. Параметри функції. Значення, що повертаються. Локальні і глобальні змінні.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

***Лекція 8*. Функції (продовження)**

8.1. Стек і функції. Функції, що підставляються.

8.2. Перевантаження функцій.

8.3. Рекурсія.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

***Лекція 9*. Контрольна модульна робота**

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

Тема 5. Похідні типи даних

***Лекція 10*. Похідні типи даних**

10.1. Масиви. Алгоритми обробки масивів. Оголошення масивів. Ініціалізація масивів.

10.2. Обробка одновимірних масивів даних економічного характеру.

10.3. Алгоритми сортування масивів.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

***Лекція 11*. Похідні типи даних (продовження)**

11.1. Багатовимірні масиви. Ініціалізація багатовимірного масиву.

11.2. Типові приклади обробки матриць.

11.3. Функції і масиви.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

Лекція 12*. Похідні типи даних. *Покажчики

12.1. Поняття покажчика, посилання. Адресна арифметика.

12.2. Організація пам’яті. Статичні і динамічні змінні. Оператори new і delete.

12.3. Покажчики і масиви.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

***Лекція 13*. Похідні типи даних. *Покажчики* (продовження)**

13.1. Покажчики і функції.

13.2. Динамічні масиви як параметри функцій.

13.3. Зв'язні списки.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

***Лекція 14*. Похідні типи даних. *Рядки* (продовження)**

14.1. Рядки і масиві символів.

14.2. Операції з рядками.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

***Лекція 15*. Похідні типи даних. *Рядки* (закінчення)**

15.1. Тип даних string.

15.2. Обробка об'єктів типа string.

15.3. Ввод-вивід рядків.

Література: основна [1; 3; 8]; додаткова [12; 13; 16 – 18].

Тема 6. Введення в система вводу-виводу C++

***Лекція 16*. Система вводу-виводу C++**

16.1. Базові положення системи вводу-виводу C++.

16.2. Ввід даних за допомогою глобального об'єкту cin.

16.3. Вивід даних за допомогою глобального об'єкту cout.

Література: основна [1; 2; 4; 6; 8]; додаткова [12; 13; 16 – 18].

***Лекція 17*. Система вводу-виводу C++**

17.1. Використовування файлів для вводу-виводу даних.

17.2. Файли з довільним доступом.

Література: основна [1; 2; 4; 6; 8]; додаткова [12; 13; 16 – 18].

***Лекція 18*. Контрольна модульна робота**

Література: основна [1; 2; 4; 6; 8]; додаткова [12; 13; 16 – 18].

Змістовний модуль 2. Принципи розробки Windows-додатків

Тема 7. Передпроцесорна обробка

***Лекція 19.* Передпроцесорна обробка**

19.1. Директиви передпроцесора.

19.2. Умовна компіляція.

Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Лекція 20. **Передпроцесорна обробка. Макроси (продовження)**
20.1. Основи апарату макросів.
20.2. Передвизначені макроси.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Тема 8. Структури та об’єднання

Лекція 21. **Структури**
21.1 Структури.
21.2 Структури з бітовими полями.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Лекція 22. **Структури (продовження)**
22.1. Масиви структур.
22.2. Покажчики на структури.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Лекція 23. **Структури (продовження)**
23.1. Структури як параметри функцій.
23.2. Передача за посиланням масивів структур.
23.3. Функції роботи з датою та часом.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Лекція 24. **Структури (продовження)**
24.1. Об’єднання.
24.2. Переліки.
24.3. Операції з об’єднаннями.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Тема 9. Шаблони

Лекція 25. **Шаблони функцій**
25.1. Основи апарату шаблонів.
25.2. Перевантаження шаблонів функцій.
25.3. Шаблони функцій сортування.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Лекція 26. **Стандартна бібліотека шаблонів (STL)**
26.1. Призначення та склад STL.
26.2. Контейнери. Операції з списками та чергами.
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Лекція 27. Контрольна модульна робота
Література: основна [1 – 3; 6; 8]; додаткова [12; 13; 15 – 18].

Тема 10. Програмування в середовищі Windows

Лекція 28**. Особливості програмування в середовищі Windows**
28.1. Особливості ОС Windows.
28.2. Графічні елементи вікна Windows.
28.3. Елементи призначеного для користувача інтерфейсу Windows.
28.4. Поняття повідомлення.
Література: основна [6; 7]; додаткова [10; 11; 14; 18].

Лекція 29**. Архітектура Windows-додатка**
29.1. Поняття API Windows – функції.
29.2. Типи даних в Windows-додатках. Типи вікон.
29.3. Ініціалізація додатка.
29.4. Цикл обробки повідомлень.
Література: основна [6; 7]; додаткова [10; 11; 14; 18].

Лекція 30**. Механізми взаємодії додатка з користувачем**
30.1. Файл ресурсів.
30.2. Меню додатка.
30.3. Повідомлення меню.
30.4. Акселератори.
Література: основна [6; 7]; додаткова [10; 11; 14; 18].

Лекція 31**. Діалогові вікна та їх елементи**
31.1. Типи діалогових вікон.
31.2. Функція діалогового вікна.
31.3. Оператори шаблону діалогового вікна.
31.4. Елементи управління діалогового вікна.
Література: основна [6; 7]; додаткова [10; 11; 14; 18].

***Лекція 32*. Багатовіконний інтерфейс**
32.1. Створення MDI-додатка.
32.2. Вікна MDI-додатка.
Література: основна [6; 7]; додаткова [10; 11; 14; 18].

***Лекція 33*. Інтерфейс графічних пристроїв (GDI)**
33.1 Загальні відомості про контексти пристроїв.
33.2 Контекст дисплея.
33.3 Контекст принтера.
33.4 Інформаційний контекст.
33.5 Контекст у пам'яті.
Література: основна [6; 7]; додаткова [10; 11; 14; 18].

Тема 11. Програмування на керованому C++

***Лекція 34*. Програмування на керованому C++**
34.1 C++ та платформа .NET.
34.2 Типи даних C++ і CLR.
34.3 Стандартний ввод-вивід.
Література: основна [6; 9]; додаткова [17; 18].

***Лекція 35*. Програмування на керованому C++ (продовження)**
35.1 Управління прибиранням сміття.
35.2 Типова безпека.
35.3 Типи-значення і абстрактні типи. Приведення типів.
Література: основна [6; 9]; додаткова [17; 18].

***Лекція 38*. Контрольна модульна робота**
Література: основна [1; 2; 6 – 9]; додаткова [10; 11; 13 – 18].

5. Плани практичних і лабораторних занять

5.1. План практичних занять

Практичне заняття – форма навчального заняття, спрямована на закріплення студентом теоретичних знань, отриманих як на лекційних заняттях, так і в процесі самостійного вивчення матеріалу, а також під час виконання ІНДЗ.

На практичному занятті викладач організовує детальний розгляд студентами окремих теоретичних положень навчальної дисципліни та формує вміння та навички їх практичного застосування шляхом індивідуального виконання студентом відповідно до сформульованих завдань.

Проведення практичного заняття ґрунтується на попередньо підготовленому методичному матеріалі (тестах) для виявлення ступеня оволодіння студентами необхідними теоретичними положеннями, наборі завдань різної складності для розв'язання їх студентами на занятті. Практичні заняття також проводяться з метою підготовки матеріалів до лабораторних робіт, які потребують розробки програм, або попередніх розрахунків.

Практичне заняття включає проведення попереднього контролю знань, вмінь і навичок студентів, постановку загальної проблеми викладачем та її обговорення за участю студентів, розв'язування завдань з їх обговоренням, розв'язування контрольних завдань, їх перевірку, оцінювання.

У процесі проведення практичного заняття викладач організовує слідуючі види методичної роботи з студентами:

поточний контроль у вигляді письмових тестів;

дискусію навколо попередньо визначених тем рефератів, до яких студенти готуються заздалегідь;

вирішення поточних запропонованих завдань різного рівня складності;

перевірку домашніх завдань щодо розробки програм та алгоритмів;

захист індивідуальних науково-дослідних робіт окремих студентів;

Практичне заняття проводиться в аудиторіях з однією академічною групою.

На кожному практичному занятті викладач оцінює підготовлені студентами виступи, активність у дискусії, уміння формулювати і відстоювати свою позицію тощо. Підсумкові оцінки за кожне практичне заняття вносяться у відповідний журнал. Отримані студентом оцінки за практичні заняття враховуються при виставленні поточної модульної (практичний модульний контроль) оцінки з даної навчальної дисципліни. Перелік тем практичних занять наведено у табл. 3.

Таблиця 3

Перелік тем практичних занять

Назва теми	Теми практичних занять	Кількість годин	Література
1	2	3	4
Модуль 1. Введення в розробку і кодування алгоритмів			
Тема 1. Алгоритм як основне поняття програмування. Лексичні основи мов високого рівня	1. Розробка графічних схем алгоритмів методом покрокової деталізації	2	Основна: [1; 3; 8]. Додаткова [13; 17]
Тема 2. Алгоритмічна мова С++. Основні типи даних	2. Розробка розгалужених програм на мові С++	2	Основна: [1; 3; 8]. Додаткова [13; 17]
Тема 3. Програмування обчислювальних процесів. Оператори управління програмою	3. Розробка розгалужених та циклічних програм на мові С++. 4. Розробка циклічних програм на мові С++	2 2	Основна: [1; 3; 8]. Додаткова: [12; 13; 16 – 18]
Тема 4. Функції	5. Розробка програм з використанням функцій на мові С++. 6. Рекурсія	2 2	Основна: [1; 3; 4; 8]. Додаткова: [12; 13; 16 – 18]
Тема 5. Похідні типи даних. Рядки	7. Розробка програм обробки одномірних масивів з використанням функцій на мові С++. 8. Розробка програм обробки двовиірних масивів з використанням функцій на мові С++. 9. Розробка програм обробки масивів з використанням покажчиків	2 2 2	Основна: [1; 3; 4; 8]. Додаткова: [12; 13; 16 – 18]
Тема 6. Введення в систему вводу-виводу С++	10. Розробка програм обробки рядків, що зберігаються на зовнішніх носіях	2	Основна: [1; 3; 4; 6; 8]. Додаткова: [12; 13; 15 – 18]
Модуль 2. Принципи розробки Windows-додатків			
Тема 7. Передпроцесорна обробка	11. Розробка програм обробки рядків, з використанням макросів	2	Основна: [1 – 3; 6; 8]. Додаткова: [12; 13; 15 – 18]
Тема 8. Структури та об’єднання	12. Розробка програм обробки структур. 13. Розробка програм обробки масивів структур	2 2	Основна: [1 – 3; 6; 8]. Додаткова: [12; 13; 15 – 18]

Закінчення табл. 3

Тема 9. Шаблони	14. Розробка програм роботи зі списком з використанням шаблонів	2	Основна: [1 – 3; 6; 8]. Додаткова: [12; 13; 15 – 18]
Тема 10. Програмування в середовищі Windows	15. Розробка каркасного Windows-додатка. 16. Розробка Windows-додатка з використанням меню користувача. 17. Розробка Windows-додатка з використанням вікон діалогу та інтерфейсу графічних пристроїв	2 2 2	Основна: [6; 7]. Додаткова: [10; 11; 14; 15; 18]
Тема 11. Програмування на керованому C++	18. Розробка програм на керованому C++	2	Основна: [6, 9]. Додаткова: [17; 18].

5.2. План лабораторних занять

Лабораторне заняття – форма навчального заняття, спрямована на закріплення та вдосконалення студентом теоретичних знань, отриманих як на лекційних, практичних заняттях, так і в процесі самостійного вивчення матеріалу. Під час лабораторного заняття студенти під керівництвом викладача особисто набувають практичних навичок у роботі з обчислювальною технікою, оволодівають методикою створення програмних продуктів у програмному середовищі.

На лабораторному занятті студенти під керівництвом викладача проводять експерименти чи досліди в навчальних лабораторіях з використанням комп'ютерної техніки.

Основною метою лабораторного заняття є практичне підтвердження окремих теоретичних положень та набуття практичних вмінь з виконання обчислювальних експериментів. Головна особливість цих занять полягає в тому, що вони об'єднують теорію з практикою, забезпечують їх єдність. Сукупність лабораторних занять з дисципліни є лабораторним практикумом, що сплановане за єдиним задумом. Лабораторні заняття плануються після проведення лекцій. А при необхідності розробки програм, проектування баз даних або підготовки складних розрахунків і початкових даних перед лабораторними заняттями проводяться практичні.

Напередодні проведення кожного лабораторного заняття (звичайно після відповідної лекції, практичного заняття) студентам видається завдання, що містить:

тему і мету заняття;

список питань для підготовки (це можуть бути контрольні питання з теми, що вивчається, заповнення роздаточних матеріалів індивідуальними даними, розробка програм, таблиць і т.д.);

послідовність підлягаючих виконанню на занятті дій;

вимоги до змісту звіту.

Студент повинен вивчити навчальний матеріал, завдання, підготувати необхідні для роботи на занятті матеріали і знати відповіді на контрольні питання. У ході підготовки може бути складений звіт, що дозволить заощадити час на занятті.

Лабораторні заняття проводяться в аудиторіях ОЦ, академічна група ділиться на підгрупи.

Усі лабораторні заняття з дисципліни проводяться фронтально, кожний студент працює за окремим комп'ютером.

На початку заняття, після оголошення теми, цільової установки і коротких вказівок з особливостей роботи викладачем проводиться контроль підготовленості студентів, шляхом перевірки відповідей на контрольні питання, рідше у формі усної бесіди з теми заняття. Для контролю може використовуватися і тестування. Обов'язково перевіряється наявність матеріалів для виконання роботи (програм, роздаточного матеріалу з відпрацьованими індивідуальними питаннями, початкових даних для вирішення задач, складання звіту і т. п.).

За відсутності матеріалів, необхідних для виконання роботи, і знань, які не дозволяють виконати роботу, студент до роботи не допускається, і йому пропонується виконати необхідну підготовку. Сама робота повинна виконуватися в додатковий час.

У ході заняття студенти самостійно виконують передбачені завданням дії, заносячи результати в звіт. На це відводиться до 85 – 90% часу заняття. Викладач здійснює контроль за роботою і надає допомогу при виникненні труднощів, звертає увагу на складні, ключові моменти. Причому основна увага приділяється не вказівці на конкретну помилку, а методиці пошуку причин виникнення цих помилок.

Складання звіту – це відповідальний етап лабораторного заняття. При його складанні студенти розвивають навички аналізу, узагальнення і творчого осмислення результатів роботи, а також навички розробки документації. Необхідно прагнути того, щоб студенти оформляли звіт про виконану роботу і подавали його викладачу до кінця заняття. Цьому сприяє наявність наперед підготовленого звіту, в якому послідовно заносяться всі необхідні дані і зроблені висновки. Звіт може бути представлений в традиційному вигляді або у вигляді електронного документа.

За наслідками контролю готовності студентів до роботи, об'єму і правильності її виконання, повноти і якості оформлення звіту і його захисту викладач виставляє оцінку. Звіти, які не подані під час заняття, захищаються в додатковий час. В окремих випадках оцінка може виставлятися за групу взаємопов'язаних робіт.

При оцінці лабораторної роботи викладач враховує правильність та розуміння роботи розроблених програмних продуктів, вміння працювати у програмному середовищі. Оцінки за кожну лабораторну роботу вносяться у відповідний журнал.

Студент, що пропустив лабораторне заняття або не допущений до нього зобов'язаний виконати відповідну роботу під час самостійної підготовки і відзвітувати. Повторна здача робіт, які не були прийняті, проводиться під час додаткових занять.

Оцінки, отримані студентом за окремі лабораторні заняття, враховуються при виставленні поточної модульної оцінки з навчальної дисципліни.

У процесі лабораторного заняття викладач організовує слідуючі види методичної роботи з студентами:

вирішення поточних запропонованих індивідуальних завдань на лабораторну роботу;

перевірку завдань щодо розробки програм та алгоритмів;

захист лабораторних робіт окремих студентів.

Перелік тем лабораторних занять наведено у табл.4.

Таблиця 4

Перелік тем лабораторних занять

Назва теми	Теми лабораторних занять	Кількість годин	Література
1	2	3	4
Модуль 1. Введення в розробку і кодування алгоритмів			
Тема 1. Алгоритм як основне поняття програмування. Лексичні основи мов високого рівня	1. Знайомство з інтегрованим середовищем розробки програм MS Visual C++ .NET. Підготовка та розв'язання на ПЕОМ задач лінійного характеру	6	Основна: [1; 3; 8]. Додаткова: [13; 17]
Тема 2. Алгоритмічна мова С++. Основні типи даних	2. Підготовка і розв'язання на ПЕОМ задач з розгалуженням	6	Основна: [1; 3; 8]. Додаткова: [13; 17]
Тема 3. Програмування обчислювальних процесів. Оператори управління програмою	3. Підготовка та розв'язання на ПЕОМ задач з використанням циклів	6	Основна: [1; 3; 8]. Додаткова: [12; 13; 16 – 18]

Закінчення табл. 4

1	2	3	4
Тема 4. Функції	4. Підготовка та розв'язання на ПЕОМ задач із використанням функцій	6	Основна: [1; 3; 4; 8]. Додаткова; [12; 13; 16 – 18]
Тема 5. Похідні типи даних. Рядки	5. Підготовка та розв'язання на ПЕОМ задач обробки одновимірних масивів	6	Основна: [2 – 4; 7]. Додаткова [11; 13; 16 – 18]
	6. Підготовка та розв'язання на ПЕОМ задач обробки двовимірних масивів	6	
Тема 6. Введення в систему вводу-виводу С++	7. Підготовка і розв'язання на ПЕОМ задач обробки масивів з використанням покажчиків 8. Підготовка і розв'язання на ПЕОМ задач з використанням рядків і макросів	4	Основна: [1 – 3; 6; 8]. Додаткова: [12; 13; 15 – 18]
Модуль 2. Принципи розробки Windows-додатків			
Тема 7. Передпроцесорна обробка	9. Підготовка та розв'язання на ПЕОМ задач з використанням рядків і макросів	4	Основна: [1; 2; 4; 6; 8]. Додаткова: [12; 13; 15 – 18]
Тема 8. Структури та об'єднання	10. Підготовка та рішення на ПК задач обробки масивів структур	4	Основна: [1; 2; 4; 6; 8]. Додаткова: [12; 13; 15 – 18]
Тема 9. Шаблони	11. Підготовка та рішення на ПК задач обробки масивів структур з використанням контейнерів	4	Основна: [1; 2; 4; 6; 8]. Додаткова: [12; 13; 15 – 18]
Тема 10. Програмування в середовищі Windows	12. Розробка каркасного Windows-додатка.	4	Основна: [6; 7]. Додаткова: [10; 11; 14; 18]
	13. Розробка Windows-додатка з використанням меню користувача та вікон діалогу	6	
Тема 11. Програмування на керованому С++	14. Розробка програм на керованому С++	6	Основна: [6; 9]. Додаткова: [16; 18]

6. Індивідуальне навчально-дослідне завдання

Індивідуальні навчально-дослідні завдання (ІНДЗ) виконуються самостійно при консультуванні викладачем протягом вивчення дисципліни у відповідності до графіка навчального процесу за рахунок часу відведеного на самостійну роботу.

ІНДЗ виконуються з метою закріплення, поглиблення і узагальнення знань, одержаних студентами за час навчання та придбання практичних навичок їх застосування при вирішенні практичних завдань.

Індивідуальне навчально-дослідне завдання припускає наявність наступних елементів наукового дослідження: практичної значущості; комплексного системного підходу до вирішення завдань дослідження; теоретичного використання передової сучасної методології і наукових розробок; наявність елементів творчості.

У процесі виконання ІНДЗ, разом з теоретичними знаннями і практичними навиками за фахом, студент повинен продемонструвати здібності до науково-дослідної роботи і уміння творчо мислити, навчитися вирішувати науково-прикладні актуальні задачі.

6.1 Тематика ІНДЗ:

1. "Алгоритмізація циклічних обчислювальних роцесів".

Мета роботи: набуття практичних навичок розробки алгоритмів

Основні завдання:

1) розробка графічної схеми алгоритму розв’язання практичної задачі.

2. "Обробка матриць".

Мета роботи: набуття практичних навичок розробки алгоритмів і програм обробки масивів даних

Основні завдання:

1) розробка графічної схеми алгоритму розв’язання практичної задачі;

2) розробка та відлагодження програми розв’язання практичної задачі.

3. "Створення та обробка однозв’язного списку".

Мета роботи: набуття практичних навичок розробки алгоритмів і програм обробки динамічних структур даних

Основні завдання:

1) розробка графічної схеми алгоритму розв’язання практичної задачі;

2) розробка та відлагодження програми розв’язання практичної задачі.

4. "Створення Windows-додатка".

Мета роботи: набуття практичних навичок розробки Windows-додатків.

Основні завдання:

1) розробка сценарію взаємодії користувача та додатка;

2) розробка та відлагодження програми розв'язання практичної задачі.

ІНДЗ повинне містити наступні розділи.

Титульна сторінка повинна містити назву університету; назву кафедри; назву навчальної дисципліни; тему ІНДЗ з вказівкою бази дослідження; прізвище, ініціали студента, навчальна дисципліна, номер академічної групи; дату подання ІНДЗ викладачеві на перевірку (день, місяць, рік).

Зміст повинен відтворювати назви розділів, параграфів тощо, які розкривають тему ІНДЗ, із зазначенням номерів сторінок, на яких вони розміщені.

Вступ. У "Вступі" студентом розкривається сутність і стан наукового завдання та її значущість, засоби її вирішення, вхідні та вихідні дані для розробки теми ІНДЗ.

Основна частина складається з 3-х розділів.

Перший розділ повинен містити графічну схему алгоритму розв'язання задачі та опис інтерфейсу користувача, в цьому розділі студент повинен визначити:

1) основні та допоміжні (якщо необхідно) схеми алгоритмів;

2) сценарій взаємодії користувача та програми.

Другий розділ повинен містити код програми на мові С++ з відповідними коментарями.

Третій розділ повинен містити результати тестування програми результати роботи програми з реальними вихідними даними.

Висновки. У висновках викладають аналіз отриманих результів. Далі формулюють пропозиції щодо практичного використання результатів ІНДЗ.

Список літератури потрібно розміщувати в алфавітному порядку прізвищ перших авторів або заголовків. Відомості про джерела, які включені до списку, необхідно давати згідно з вимогами державного стандарту з обов'язковим наведенням праць.

Додатки. У додатки можуть бути включені матеріали, що є копією документів, звітів, або розрахункові таблиці, узагальнюючі схеми чи діаграми. При наявності до кількох додатків оформлюється окрема сторінка "ДОДАТКИ", номер якої є останнім, що відноситься до обсягу ІНДЗ.

Вимоги до оформлення. ІНДЗ слід оформляти розбірливим почерком, чорнилом (пастою) одного кольору. Дозволяється друкувати машинописним способом або за допомогою комп'ютера на одній стороні аркуша білого паперу формату А4 (210х297 мм), – міжрядковий інтервал 1,5 згідно з вимогами Державного стандарту України ДСТУ 3008-95 "Документація. Звіти в сфері науки і техніки. Структура та правила оформлення" (тридцять рядків на сторінку). Мінімальна висота шрифту основного тексту має бути не меншою за 8 мм. Цифри та букви необхідно писати чітко, висотою не менш 3,5 мм.

Обсяг ІНДЗ повинен становити 5 – 10 друкованих сторінок. Текст ІНДЗ необхідно писати (друкувати), залишаючи береги таких розмірів:

лівий – не менше 30 мм;

правий – не менше 10 мм;

верхній – не менше 20 мм;

нижній – не менше 20 мм.

Текст основної частини ІНДЗ поділяють на розділи і підрозділи. Заголовки структурних частин ІНДЗ "ЗМІСТ", "ВСТУП", "РОЗДІЛ ...", "ВИСНОВКИ", "СПИСОК ЛІТЕРАТУРИ", "ДОДАТКИ" пишуть (друкують) великими літерами симетрично до тексту.

Заголовки підрозділів пишуть (друкують) маленькими літерами (крім першої великої) з абзацу. Крапку в кінці заголовка не ставлять. У кінці заголовка, написаного (надрукованого) в підбір до тексту, ставиться крапка.

Усі структурні складові основної частини ІНДЗ починаються з нових сторінок, відокремлюються від наступного тексту одним пустим рядком. Крапка після назви розділу або підрозділу не ставиться.

Нумерацію сторінок, розділів, підрозділів, рисунків, таблиць подають арабськими цифрами без знаку №.

Структурні частини роботи "ЗМІСТ", "ВСТУП", "ВИСНОВКИ", "СПИСОК ЛІТЕРАТУРИ" не нумерують. Номер розділу ставлять після слова "РОЗДІЛ" на тому ж рядку, після номера крапку не ставлять, а потім з нового рядка пишуть (друкують) заголовок розділу.

Підрозділи нумерують у межах кожного розділу. Номер підрозділу складається з номера розділу та порядкового номера підрозділу, між якими ставлять крапку. В кінці номера підрозділу повинна стояти крапка, наприклад: "2.3." (третій підрозділ другого розділу). Потім у тому ж рядку йде заголовок підрозділу.

Ілюстрації та таблиці необхідно подавати безпосередньо після тексту, де вони згадані (посилання за зразком – "подано на рис. 3.1", "дивись у табл. 3.2" або "... (рис. 3.2)") вперше, або на наступній сторінці. Ілюстрації та таблиці, які розміщенні на окремих сторінках, включають до загальної нумерації сторінок.

Ілюстрації позначають словом "Рис." та нумерують послідовно в межах розділу, за виключенням ілюстрацій, поданих у додатках.

Номер ілюстрації повинен складатися з номера розділу та порядкового номера ілюстрації, між якими ставиться крапка. Наприклад: "Рис. 1.2" (другий рисунок першого розділу). За умови наявності у роботі тільки однієї ілюстрації, цей рисунок нумерується як "Рис. 1". Номер ілюстрації, її назва та пояснювальні підписи розміщують послідовно під ілюстрацією.

Таблиці нумерують послідовно (за винятком таблиць, поданих у додатках) в межах розділу. В правому верхньому куті над відповідним заголовком таблиці розміщують напис "Таблиця" із зазначенням її номера. Номер таблиці повинен складатися з номера розділу та порядкового номеру таблиці, між якими ставиться крапка, наприклад: "Таблиця 1.2" (друга таблиця першого розділу).

При переносі частини таблиці на інший аркуш (сторінку) слово "Таблиця" та її номер вказують один раз справа над першою частиною таблиці, над іншими частинами пишуть слова "Продовження табл." та вказують номер таблиці, наприклад: "Продовження табл. 1.2".

7. Самостійна робота студентів

Необхідним елементом успішного засвоєння навчального матеріалу дисципліни є самостійна робота студентів з вітчизняною та закордонною спеціальною літературою. Самостійна робота є основним засобом оволодіння навчальним матеріалом у час, вільний від обов’язкових навчальних занять. Самостійна робота студентів передбачає поглиблене вивчення тем з використанням рекомендованої літератури, пошук інформації в Інтернеті, а також додаткову роботу в комп’ютерних класах для виконання індивідуальних завдань. Основні види самостійної роботи, які запропоновані студентам:

1. Вивчення лекційного матеріалу.
2. Робота з вивчення рекомендованої літератури.
3. Вивчення основних термінів та понять з галузі обчислювальної техніки і програмування.
4. Підготовка до лабораторних і практичних занять.
5. Робота над ІНДЗ.
6. Робота над рефератом.

Тематика рефератів по курсу

1. Історія розвитку мови програмування С++.
2. Типізація даних в С++: вбудовані, похідні типи і класи.
3. Призначення і використання покажчиків в С++.
4. Масиви та їх реалізація в С++.
5. Операції, які використовуються в мові програмування С++. Пріоритети виконання операцій.
6. Конструкції мови програмування С++, що управляють.
7. Робота з рядками в С++.
8. Типові математичні функції мови С++.
9. Стандартні потоки вводу-виводу.
10. Файлові потоки вводу-виводу.
11. Динамічний розподіл і звільнення пам’яті.
12. Загальна характеристика динамічних структур даних.

8. Контрольні запитання для самодіагностики

1. Структура типової програми мов С++.
2. Бібліотечні файли, їх організація та використання.
3. Опишіть можливості застосування інтегрованого середовища Visual C++ .NET для розробки Windows – додатків.
4. Дайте класифікацію типів даних, способи опису їх в програмах, приклади використання.
5. Перерахуйте структури мови С++, що управляють, і вкажіть особливості їх застосування.
6. Формат і робота умовного оператора в повній і скороченій формах. Наведіть приклади.
7. Формат і робота оператора **switch**. Наведіть приклади.
8. Формат і робота оператора циклу **while** (цикл з передумовою). Наведіть приклади.
9. Формат і робота оператора циклу **do/while** (цикл з післяумовою). Наведіть приклади.
10. Формат і робота оператора циклу **for**. Наведіть приклади.
11. Як записати два і більше виразів в умові **while**?
12. Чи відрізняється умова циклу в циклі **while** від умови циклу **do/while**?

13. Дайте визначення поняття "масив". До яких типів даних відносяться масиви в C++?

14. Наведіть приклад циклу **for** від 0 до 20. Коли виконується кожний з трьох виразів?

15. Де в заголовку **for** може використовуватися кома?

16. Чи можна опускати крапки з комою в циклі **for**?

17. Як працює оператор **break**?

18. Як працює оператор **continue**?

19. Як нумеруються елементи масиву a [5]?

20. Чи може розмірність масиву визначатися змінною?

21. Як ініціалізувати масив? Чи можна при цьому опускати розмірність?

22. Що буде, якщо в списку ініціалізації менше елементів, ніж розмірність масиву? А якщо більше?

23. Чи можна ініціалізувати символьний масив рядком символів?

24. Чи можна привласнити масив масиву?

25. Визначити поняття покажчика в C++. Чим відрізняються покажчики від посилань?

26. Що відбудеться, якщо буде вихід за межу масиву?

27. Що значить запис a[1] для масиву a[5][5]?

28. Що тут визначене: int *i1, i2?

29. Як визначається операція отримання адреси в C++?

30. Що таке покажчик типу void*? Чи можна його розіменувати?

31. Що позначає окремо ім’я a для масиву a[10]?

32. На що вказує ptr + 10, якщо ptr вказує на перший елемент масиву?

33. Чим відрізняється робота з масивом від роботи з покажчиком?

34. Як розмістити в C++ об’єкт у вільній пам’яті?

35. Як звільнити пам’ять об’єкта?

36. Опишіть структури даних – масиви. Поясніть організацію багатовимірних масивів. Для чого використовуються покажчики і посилання?

37. Програмні модулі C++ (функції). Визначення функцій, виклик, прототипи.

38. Створіть функцію, яка приймає три аргументи: ім’я масиву елементів типу int, розмір масиву і значення типу int. Функція повинна встановити для кожного елементу масиву значення типу int.

39. Як здійснюється організація файлової системи в C++?

40. Дайте визначення поняття "потік" в C++. Назвіть і охарактеризуйте стандартні потоки вводу-виводу.

41. Опишіть ієрархію потокових класів в C++.

42. Назвіть засоби форматування вводу-виводу.

43. Вкажіть потоки файлового вводу-виводу.

44. Які операції використовуються для запису даних у файл і для читання з файлу?

45. У чому відмінність консольного застосування від графічного?

46. Призначення і склад головної функції WinMain( ).

47. Призначення і склад структур WNDCLASS і MSG.

48. Перерахуйте характеристики класу вікна.

49. Опишіть алгоритм створення і показу вікна.

50. Призначення і принцип реалізації циклу обробки повідомлень.

51. Призначення і склад віконної функції.

52. Опишіть алгоритм створення кнопок в Windows-вікні.

53. Укажіть принципи обробки повідомлень WM_CHAR, WM_LBUTTONDOWN, WM_RBUTONDOWN і повідомлень групи WM_COMMAND.

54. Укажіть призначення функцій MessageBox( ), GetWindowText( ), SetWindowText( ), SendMessage( ).

55. Як створювати діалогові вікна за допомогою редактора ресурсів?

56. Вкажіть призначення типових елементів управління діалогових вікон (Button, Edit Control, Static Text, Radio Button, Check Box) і формати їх опису у файлах ресурсів.

57. Перерахуйте призначення функцій DialogBox( ), GetDlgItemText( ), GetDlgItem( ), SetWindowText( ).

58. Сформулюйте концепцію побудови інтерфейсу графічних пристроїв (GDI).

59. У чому суть контекстного представлення пристроїв?

60. Призначення бібліотеки STL?

61. У чому особливості обробки події WM_PAINT.

62. Перерахуйте основні директиви компілятора.

63. Наведіть приклади функцій що працюють зі списками.

64. Вкажіть призначення файлів шаблонного проекту простого Windows-додатка.

65. Опишіть ресурси, що використовуються в шаблонному проекті і механізми їх модифікації.

9. Індивідуально-консультативна робота

Індивідуально-консультативна робота здійснюється за графіком індивідуально-консультативної роботи у формі: індивідуальних занять, консультацій, перевірки виконання індивідуальних завдань, перевірки та захисту завдань, що винесені на поточний контроль тощо.

Формами організації індивідуально-консультативної роботи є:

а) за засвоєнням теоретичного матеріалу:

консультації: індивідуальні (запитання – відповідь);

групові (розгляд типових прикладів);

б) за засвоєнням практичного матеріалу:

консультації індивідуальні і групові;

в) для комплексної оцінки засвоєння програмного матеріалу:

індивідуальне здавання виконаних робіт, звітів;

підготовка реферату.

10. Методики активізації процесу навчання

При викладанні дисципліни передбачено застосування активних і інтерактивних методів навчання – ділових ігор, рольових ігор, тренінгів, семінарів у активній формі, розгляд кейсів, модерації. Основні відмінності активних та інтерактивних методів навчання від традиційних визначаються не тільки методикою і технікою викладання, але і високою ефективністю навчального процесу, який виявляється у:

високій мотивації студентів;

закріпленні теоретичних знань на практиці;

підвищенні самосвідомості студентів;

виробленні здатності ухвалювати самостійні рішення;

виробленні здібності до колективних рішень;

виробленні здібності до соціальної інтеграції;

придбанні навичок вирішення конфліктів;

розвитку здібності до компромісів.

Розподіл форм та методів активізації процесу навчання за темами навчальної дисципліни наведено у табл. 5.

Таблиця 5

Розподіл форм та методів активізації процесу навчання за темами навчальної дисципліни

Тема	Практичне застосування навчальних технологій
Тема 1. Алгоритм як основне поняття програмування. Лексичні основи мов високого рівня	*Міні-лекція, семінар-дискусія, мозкова атака* з питання "Розробка графічних схем алгоритмів". *Презентація* результатів роботи в малих групах
Тема 2. Алгоритмічна мова С++. Основні типи даних	*Міні-лекція, семінар-дискусія* з питань "Способи завдання логічних функцій у мові С++". *Ділова гра* з питання "Представлення даних у пам'яті ЕОМ"
Тема 3. Програмування обчислювальних процесів. Оператори управління програмою	*Проблемна лекція* з питання "Організація циклічних процесів". *Кейс-метод* з питання "Розширення можливостей циклічних програм"
Тема 4. Функції	*Проблемна лекція* з питання "Використання процедурного програмування". *Презентація* результатів роботи в малих групах
Тема 5. Похідні типи даних. Рядки	*Міні-лекція, семінар-дискусія* з питання "оптимальна обробка символьної інформації"
Тема 6. Введення в систему вводу-виводу С++	*Проблемна лекція* з питання "Шляхи удосконалення методів зберігання даних". *Ознайомлювальні ігри* "Оцінка властивостей двійкових файлів"
Тема 7. Передпроцесорна обробка	*Міні-лекція, семінар-дискусія* з питань з питання "Універсальні властивості макросів". *Презентація* результатів роботи в малих групах
Тема 8. Структури та об'єднання	*Міні-лекція, семінар-дискусія* з питання "Засоби зберігання та обробки табличних даних". *Презентація* результатів роботи в малих групах
Тема 9. Шаблони	*Модерація* з питання "Що робити з чергами?"
Тема 10. Програмування в середовищі Windows	*Початкова гра* за темою "Події в сучасних ОС"
Тема 11. Програмування на керованому С++	*Міні-лекція, семінар-дискусія* з питань з питання "Безпечне програмування". *Презентація* результатів роботи в малих групах

Проблемні лекції – спрямовані на розвиток логічного мислення студентів і характеризуються тим, що коло питань теми обмежується двома-трьома ключовими моментами, увага студентів концентрується на матеріалі, що не знайшов відображення в підручниках, використовується досвід закордонних навчальних закладів з роздачею студентам під час лекцій друкованого матеріалу та виділенням головних висновків з питань, що розглядаються. При читанні лекцій студентам даються питання для самостійного розмірковування, проте лектор сам відповідає на них,

не чекаючи відповідей студентів. Система питань в ході лекції відіграє активізуючу роль, примушує студентів сконцентруватися і почати активно мислити в пошуках правильної відповіді. Використовуються під час викладання лекцій за темами 3, 4, 6.

Міні-лекції – передбачають виклад навчального матеріалу за короткий проміжок часу й характеризуються значною ємністю, складністю логічних побудов, образів, доказів та узагальнень. Міні-лекції проводяться, як правило, як частина заняття-дослідження і використовуються під час практичних, лабораторних занять та консульнацій.

Робота в малих групах – використовується з метою активізації роботи студентів при проведенні семінарських і практичних занять. Це так звані групи психологічного комфорту, де кожен учасник відіграє свою особливу роль і певними своїми якостями доповнює інших. Використання цієї технології дає змогу структурувати практично-семінарські заняття за формою і змістом, створює можливості для участі кожного студента в роботі за темою заняття, забезпечує формування особистісних якостей та досвіду соціального спілкування. Використовується під час лабораторних занять.

Семінари-дискусії – передбачають обмін думками і поглядами учасників з приводу даної теми, а також розвивають мислення, допомагають формувати погляди і переконання, виробляють вміння формулювати думки й висловлювати їх, вчать оцінювати пропозиції інших людей, критично підходити до власних поглядів. Використовуються під час захисту лабораторних робіт та рефератів.

Мозкові атаки – це метод розв’язання невідкладних завдань за дуже обмежений час. Сутність його полягає в тому, щоб висловити як найбільшу кількість ідей за невеликий проміжок часу, обговорити і здійснити їх селекцію. Використовуються під час викладання лекцій, практичних занять.

Ознайомлювальні або початкові ігри – частіше за все використовуються на початку занять для створення робочої атмосфери, "настройки" учасників на групову роботу.

Презентації – виступи перед аудиторією – використовуються для представлення певних досягнень, результатів роботи групи, звіту про виконання індивідуальних завдань, інструктажу, демонстрації нових товарів і послуг.

Кейс-метод – метод аналізу конкретних ситуацій, який дає змогу наблизити процес навчання до реальної практичної діяльності спеціалістів і передбачає розгляд виробничих, управлінських та інших ситуацій, складних конфліктних випадків, проблемних ситуацій, інцидентів у процесі вивчення навчального матеріалу. Використовуються під час практичних занять.

Модерація – це метод, який допомагає групам розглядати теми, проблеми, задачі зосереджуючись на змісті цілеспрямовано і ефективно при самостійній участі кожного у вільній колегіальній атмосфері. Модерація як спосіб проведення обговорення швидко призводить до конкретних результатів, дає можливість всім присутнім брати участь в процесі вироблення рішень, відчуваючи при цьому свою повну відповідальність за результат. Використовується під час практичних занять.

Банки візуального супроводження – сприяють активізації творчого сприйняття змісту дисципліни за допомогою наочності. Використовуються під час викладання лекцій.

11. Система поточного та підсумкового контролю знань студентів

У процесі навчання студенти отримають необхідні знання під час лекційних занять, виконуючи практичні завдання на лабораторних та практичних заняттях щодо розробки алгоритмів та програм.

Оцінювання знань, вмінь та навичок студентів враховує види занять, які згідно з програмою навчальної дисципліни "Основи програмування і алгоритмічні мови" передбачають лекційні, лабораторні та практичні заняття, а також самостійну роботу та виконання індивідуальних завдань.

Перевірка та оцінювання знань студентів може проводитись кількома методами:

1. Оцінювання знань студента під час практичних занять.
2. Виконання індивідуальних навчально-дослідних завдань.
3. Написання реферату.
4. Виконання та захист лабораторних робіт.
5. Проведення поточно-модульного контролю.
6. Проведення заліку.
7. Проведення підсумкового письмового іспиту.

Оцінювання знань студента під час практичних, лабораторних занять та виконання індивідуальних завдань проводиться за 12-бальною шкалою за такими критеріями:

1) ступінь засвоєння фактичного теоретичного матеріалу навчальної дисципліни;

2) ознайомлення з рекомендованою літературою, а також із сучасною літературою з питань, що розглядаються;

3) уміння поєднувати теорію з практикою при розв’язанні задач, проведенні розрахунків при виконанні індивідуальних завдань, та завдань, винесених на розгляд в аудиторії;

4) якість розробки та роботи програмних продуктів;

5) логіка, структура, стиль викладу матеріалу в письмових роботах, звітах і при виступах в аудиторії, вміння обгрунтовувати свою позицію, здійснювати узагальнення інформації та робити висновки.

Оцінка "відмінно" (10 – 12 балів) ставиться за умови відповідності індивідуального завдання студента, або його усної відповіді усім п'ятьом зазначеним критеріям. Відсутність тієї або іншої складової знижує оцінку на відповідну кількість балів.

При оцінюванні індивідуальних завдань увага також приділяється якості, самостійності та своєчасності здачі виконаних завдань викладачу (згідно з графіком навчального процесу). Якщо якась із вимог не буде виконана, то оцінка, на розсуд викладача, буде знижена.

Оцінювання знань студента під час виконання завдань для самостійної роботи проводиться за 12-бальною шкалою.

Реферат є додатковою частиною самостійної роботи студента над навчальною дисципліною "Основи програмування і алгоритмічні мови". Мета реферату – поглиблення теоретичних знань, набутих студентами в процесі вивчення дисципліни.

Написання реферату має сприяти глибшому засвоєнню студентами дисципліни "Основи програмування і алгоритмічні мови", спонукає грунтовно вивчати теретичну базу з питань програмування мовою С++, спеціальні наукові видання вітчизняних і закордонних авторів, у яких розглядаються питання розробки сучасних програмних продуктів.

Першим етапом написання реферату є вибір теми. Студенти обирають тему реферату за власним розсудом, але відповідно до тематики рефератів, визначеної кафедрою. За погодженням з викладачем студент може підготувати реферат на іншу тему, якої немає у цьому переліку.

Після вибору теми студент повинен розробити й викласти в письмовій формі його план. План теми слід розробляти після ознайомлення з літературою, яка висвітлюють ті чи інші питання і проблеми з теми дослідження. Це дасть змогу студенту детальніше уявити собі структуру реферату, послідовно викласти його зміст, повніше висвітлити коло питань, які мають бути вирішені.

План має включати лише ті питання, які безпосередньо стосуються теми і дають змогу повно і глибоко розкрити її.

Писати реферат слід на білому папері стандартного формату А4. Аркуші можна зшивати будь-яким способом, але так, щоб вони не розсипалися.

Титульний аркуш реферату повинен мати такий зміст: назва університету; назва кафедри; назва навчальної дисципліни; тема реферату; прізвище, ініціали студента, навчальна дисципліна, номер академічної групи; дата подання реферату викладачеві на перевірку (день, місяць рік).

За титульним аркушем слідує детальний план реферату, в якому потрібно виділити вступ, два – три підрозділи основного змісту, висновки та список використаної літератури та додатки.

Складні таблиці, які не вміщуються в тексті, а також інші допоміжні матеріали включаються в додатки до роботи. При цьому в тексті на них робляться відповідні посилання.

Усі аркуші слід пронумерувати – порядковий номер ставиться в правому верхньому куточку сторінки, при цьому нумерація починає ставитись на першому аркуші після вступу.

У кінці реферату дається повний список використаної літератури. Його необхідно скласти у певному порядку: спочатку наводяться нормативні документи, статистичні довідники, загальна та спеціальна література за алфавітом.

Реферат має бути виконано і подано на кафедру не пізніше зазначеної в навчальному плані дати.

Реферат оцінюється за критеріями:

самостійності виконання;

логічності та деталізації плану;

повноти й глибини розкриття теми;

наявності ілюстрації (таблиці, рисунки, схеми тощо);

кількості використаної літератури (не менше десяти);

якості оформлення.

Оцінка за реферат є складовою підсумкової оцінки з заліку з даної навчальної дисципліни.

Проведення поточно-модульного контролю. Поточно-модульний контроль здійснюється два рази за семестр (два рази у рамках вивчення змістовного модулю) та оцінюється за трьома складовими: практичний модульний контроль, теоретичний модульний контроль і модульний контроль виконання ІНДЗ. Оцінка за практичну складову модульного контролю виставляється за результатами оцінювання знань студента під час захисту лабораторних робіт та проміжного тестового контролю згідно з графіком навчального процесу.

Теоретичний модульний контроль здійснюється у письмовій формі, у вигляді контрольних опитувань на кожному практичному занятті. Оцінка за теоретичну складову модульного контролю виставляється за результатами контрольних опитувань і поточних оцінок студента на практичних заняттях.

Оцінка за ІНДЗ виставляється за результатами оцінювання знань студента при захисті індивідуальних завданнь і враховує своєчасність їх виконання.

Для підведення підсумків роботи студентів із першого змістовного модуля виставляється підсумкова залікова оцінка яка, складається з оцінок за поточно-модульний контроль і оцінки за реферат.

Проведення підсумкового письмового іспиту. Умовою допуску до іспиту є позитивні оцінки з поточно-модульного контролю. Підсумковий контроль знань студентів здійснюється у письмовій формі за екзаменаційними білетами за 12-бальною шкалою.

Екзаменаційні білети включають такі завдання:

1) теоретичне запитання;

2) практичні завдання (ситуації) різного ступеня складності (стереотипне, діагностичне, евристичне).

Зразок екзаменаційного завдання

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний економічний університет

Напрям підготовки "Комп'ютерні науки" 2 Семестр

Навчальна дисципліна "Основи програмування і алгоритмічні мови"

Екзаменаційний білет №_

Завдання 1.

Опишіть структуру типової програми на мові С++.

Завдання 2.

Написати програму, що здійснює наступну обробку масиву.

У одновимірному масиві, що складається з *n* речовинних елементів, обчислити:

1) суму негативних елементів масиву;

2) добуток елементів масиву, розташованих між максимальним і мінімальним елементами;

3) упорядкувати елементи масиву за збільшенням;

Реалізацію кожного пункту завдання представити у вигляді функції і відповідної їй схеми алгоритму.

Завдання 3.

Створити динамічну структуру з ім'ям student простого однозв'язного списку, кожен елемент якої містить наступні поля: name – прізвище студента, vosrast – його вік і gryp – номер групи. Формування нового елементу повинне починатися з відповідного запиту з клавіатури.

Роздрукувати вміст структури.
Зберегти вміст структури у файлі на магнітному диску.

Затверджено на засіданні кафедри інформаційних систем
Протокол № __ від __.__.200_р.

Зав. кафедри (підпис) Екзаменатор (підпис)

Кожне завдання екзаменаційного білета оцінюється окремо. Загальна оцінка дорівнює середній арифметичній із суми оцінок кожного завдання. Якщо одна з оцінок "незадовільно", то загальна оцінка не може бути вищою за "задовільно".

Для оцінки рівня відповідей студентів на теоретичні запитання та вирішення практичних завдань використовуються такі критерії:

оцінка **"відмінно"**

(12 балів) ставиться за глибоке засвоєння програмного матеріалу, застосування для відповіді не тільки рекомендованої, а й додаткової літератури та творчого підходу; чітке володіння понятійним апаратом, методами, методиками та інструментами розробки програмних продуктів, вміння використовувати їх для виконання конкретних практичних завдань. Відповідь на теоретичне питання білету має бути вірною та повною, оформлення відповіді – акуратним, логічним та послідовним;

(11 балів) ставиться за глибоке засвоєння програмного матеріалу, засвоєння рекомендованої літератури; чітке володіння понятійним апаратом, методами, методиками та інструментами розробки програмних продуктів, вміння використовувати їх для виконання конкретних практичних завдань. Відповідь на теоретичне питання білету має бути вірною та повною, оформлення відповіді – акуратним, логічним та послідовним;

оцінка **"дуже добре"** (10 балів) ставиться за повне засвоєння програмного матеріалу та рекомендованої літератури; чітке володіння понятійним апаратом, методами, методиками та інструментами розробки програмних продуктів, вміння використовувати їх для виконання кон-кретних практичних завдань. Відповідь на теоретичне питання білету має бути вірною та повною, оформлення відповіді – акуратним, логічним та послідовним. Припускаються незначні випадкові погрішності, які не надають суттєвого впливу на повноту та змістовність відповіді;

оцінка **"добре"**

(9 балів) ставиться за повне засвоєння програмного матеріалу та наявне вміння орієнтуватися в ньому, усвідомлене застосування знань для розв’язання практичних завдань; за умови виконання всіх вимог, які

передбачено для оцінки "відмінно", при наявності незначних помилок (тобто методичний підхід до вирішення завдання є вірним, але припущені неточності у кодах) або не зовсім повних висновків за одержаними результатами вирішення завдання. Оформлення виконаного завдання (код програми) має бути охайним (відповідати вимогам до коду);

(8 балів) ставиться за повне засвоєння програмного матеріалу та наявне вміння орієнтуватися в ньому, усвідомлене застосування знань для розв'язання практичних завдань. Практичні завдання виконуються в цілому правильно з використанням типового алгоритму, але при їх виконанні студент припускається окремих помилок. Оформлення виконаного завдання має бути охайним;

оцінка **"задовільно"** (7 балів) ставиться, якщо студент при виконанні практичних завдань ефективно застосовує основні знання навчального матеріалу, що передбачені навчальною програмою. Оцінка "задовільно" ставиться за умови, якщо завдання в основному виконане та мету завдання досягнуто, а студент при відповіді продемонстрував розуміння основних положень матеріалу навчальної дисципліни;

оцінка **"достатньо"**

(6 балів) ставиться за недостатнє вміння застосовувати теоретичні знання для розв'язання практичних завдань; за умови, якщо завдання в основному виконане та мету завдання досягнуто, а студент при відповіді продемонстрував розуміння основних положень матеріалу навчальної дисципліни;

(5 балів) ставиться за часткове вміння застосовувати теоретичні знання для розв'язання практичних завдань; за умови, якщо завдання частково виконане, а студент при відповіді продемонстрував розуміння основних положень матеріалу навчальної дисципліни;

(4 бали) ставиться у випадках, якщо студент при виконанні практичних завдань без достатнього розуміння застосовує навчальний матеріал, припускається суттєвих помилок, стикається з труднощами при застосуванні програмних засобів;

оцінка **"незадовільно"**

(3 бали) ставиться студенту за неопанування значної частини програмного матеріалу, який не може правильно виконати практичні завдання, стикається зі значними труднощами при застосуванні програмних засобів;

(2 бали) ставиться студенту, що не опанував програмний матеріал, не може правильно виконати практичні завдання, стикається зі значними труднощами при застосуванні програмних засобів;

(1 бал) ставиться за невиконання завдання загалом.

Для підведення підсумків роботи студентів з навчальної дисципліни "Основи програмування і алгоритмічні мови**"** виставляється загальна оцінка, яка враховує оцінки за кожним видом контролю (чотири оцінки поточно-модульного контролю за роботу протягом двох семестрів та оцінки за результатами заліку та іспиту).

Підсумкова оцінка з дисципліни згідно з Методикою переведення показників успішності знань студентів Університету в систему оцінювання за шкалою ECTS конвертується в підсумкову оцінку за шкалою ECTS (табл. 6).

Таблиця 6

Переведення показників успішності знань студентів ХНЕУ в систему оцінювання за шкалою ECTS

Відсоток студентів, які зазвичай досягають відповідної оцінки	Оцінка за шкалою ECTS		Оцінка за бальною шкалою, що використовується в ХНЕУ	Оцінка за національною шкалою
10	відмінне виконання	A	12 – 11	відмінно
25	вище середнього рівня	B	10	
30	взагалі робота правильна, але з певною кількістю помилок	C	9 – 7	добре
25	непогано, але зі значною кількістю недоліків	D	6	задовільно
10	виконання задовольняє мінімальні критерії	E	5 – 4	
–	потрібне повторне перескладання	FX	3	незадовільно
–	повторне вивчення дисципліни	F	2 – 1	

12. Рекомендована література

12.1 Основна

1. Браткевич В. В Основы программирования и алгоритмические языки (язык программирования С++): Конспект лекций Ч. 1 / В. В Браткевич, Ю. В. Перколаб, Л. И. Лукашева. – Харьков: Изд. ХГЭУ, 2001. – 68 с.

2. Браткевич В. В. Основы программирования и алгоритмические языки (язык программирования С++): Конспект лекций Ч. 2 / В. В. Браткевич, Ю. В. Перколаб, Л. И. Лукашева. – Харьков: Изд. ХГЭУ, 2002. – 152 с.

3. Методические рекомендации к лабораторным работам по курсу "Основы программирования и алгоритмические языки" для студентов специальности 7.080401 дневной формы обучения. Ч. 1. / Сост.: Ю. В. Перколаб, В. В. Браткевич, Л. И. Лукашева. – Харьков: Изд. ХГЭУ, 2001. – 68 с.

4. Методические рекомендации к лабораторным работам по курсу "Основы программирования и алгоритмические языки" для студентов специальности 7.080401 дневной формы обучения. Ч. 2. / Сост.: Ю. В. Перколаб, В. В. Браткевич, Л. И. Лукашева. – Харьков: Изд. ХГЭУ, 2002. – 68 с.

5. Методичні рекомендації до виконання робіт з курсу "Основи програмування і алгоритмічні мови" (робота з файлами) для студентів спеціальностей 7.080401, 7.080407 усіх форм навчання. Ч. 3 / Укл. Ю. В. Перколаб, В. В. Браткевич, І. О. Бондар. – Харків. Вид. ХНЕУ, 2005. – 64 с.

6. Пирогов В. Ю. Программирование на Visual C++.NET. – СПб.: БХВ-Петербург, 2003. – 800 с.

7. Румянцев П. В. Азбука программирования в Win32 API. – М.: Горячая линия – Телеком 2001. – 312 с.

8. Харви Дейтел. Как программировать на C++ / Харви Дейтел, Пол Дейтел. [Пер. с англ. – М.: ЗАО Изд. "БИНОМ", 2001 г. – 1024 с.

9. Шеферд Д. Программирование на Microsoft Visual C++.NET / Пер. с англ. – М.: Издательско-торговый дом "Русская редакция", 2003. – 928 с.

12.2. Додаткова

10. Верма Р. Д. Справочник по функциям Win32 API. – М.: Горячая линия – Телеком, 2002. – 488 с.

11. Методические рекомендации к выполнению лабораторных работ по курсу "Системное программирование" для студентов всех форм обучения по специальности 7.080401. Ч. 1 / Сост. Д. Ю. Голубничий, В. Ф. Третьяк.

12. Павловская Т. А. C/C++. Программирование на языке высокого уровня. – СПб.: Питер, 2006. – 461 с.

13. Подбельский В. В. Язык C++: Учеб. пособие. – 4-е изд. – М.: Финансы и статистика, 1999. – 560 с.

14. Саймон Р. Windows 2000 API. Энциклопедия программиста. / Пер. с англ. – К.: ООО "ДиасофтЮП", 2002. – 1088 с.

15. Страуструп Б. Язык программирования C++. / Пер. с англ. – 3-е изд. – СПб.; М.: "Невский диалект" – Издательство БИНОМ", 1999 – 991 с.

12.3. Ресурси мережі Internet

16. Опис стандарту C++ANSI / ISO // http://reality.sqi.com/austern/std-c++/fag.htnl

17. Підручник для початківців що вивчають Microsoft Visual C++ // http://chesworth.com/pv/languages/c/visual_cpptutuorial.htm

18. Список питань, що часто задаються (з відповідями) // http://www.math.uio.no/nett/fag/C-fag/fag.html

Зміст

НАВЧАЛЬНЕ ВИДАННЯ

Робоча програма
навчальної дисципліни

"ОСНОВИ ПРОГРАМУВАННЯ ТА АЛГОРИТМІЧНІ МОВИ"

для студентів напряму підготовки "Комп'ютерні науки"
усіх форм навчання

Укладачі: **Федорченко Володимир Михайлович**
Лосєв Міхаїл Юрійович
Парфьонов Олександр Всеволодович
Щербаков Олександр Всеволодович

Відповідальний за випуск **Прізвище І. П.**

Редактор **Прізвище І. П.**

Коректор **Бриль В. О.**

План 2008 р. Поз. №215.
Підп. до друку Формат 60 × 90 1/16. Папір MultiCopy. Друк Riso.
Ум.-друк. арк. 2,75. Обл.-вид. арк. 3,16. Тираж прим. Зам. №

Свідоцтво про внесення до Державного реєстру суб'єктів видавничої справи ***Дк №481 від 13.06.2001 р.***

Видавець і виготівник — видавництво ХНЕУ, 61001, м. Харків, пр. Леніна, 9а

Робоча програма
навчальної дисципліни

"ОСНОВИ ПРОГРАМУВАННЯ ТА АЛГОРИТМІЧНІ МОВИ"

для студентів напряму підготовки "Комп'ютерні науки"
усіх форм навчання