Міністерство освіти і науки України

Тернопільський національний технічний університет імені Івана ПУЛЮЯ

кафедра програмної інженерії

# М Е Т О Д И Ч Н І  В К А З І В К И

щодо самостійної роботи студентів
та модульного контролю знань

з дисципліни

# «ОСНОВИ ПРОГРАМУВАННЯ»

для студентів першого рівня вищої освіти за
спеціальністю No 121 Інженерія програмного забезпечення

Тернопіль 2020

Методичні вказівки щодо самостійної роботи студентів та модульного контролю знань з дисципліни “Основи програмування” для студентів першого рівня вищої освіти за спеціальністю No 121 Інженерія програмного забезпечення / Уклад.: М.Петрик, О.Петрик - Тернопіль: ТНТУ 2020 - 24 с.

Призначені для полегшення засвоєння дисципліни “Основи програмування” і контролю знань студентів. Складається з урахуванням модульної системи навчання, рекомендацій до самостійної роботи і індивідуальних завдань, тем практичних та лабораторних занять, тестів, екзаменаційних питань, типових завдань до модульних контролів

Укладачі: О.Петрик, М.Петрик

Розглянуто на засіданні кафедри програмної інженерії,

протокол № 2 від 15.09.2020р.

ЗМІСТ

# ВСТУП

Щоб навчити студентів основам алгоритмізації, необхідне, насамперед, прагнення самого студента досягти успіху). Навчити програмувати можна тільки студента, який сам хоче навчитись.

Метою викладання дисципліни є надання майбутнім фахівцям основ науково – теоретичних знань та практичних навичок з програмування сучасними алгоритмічними мовами високого рівня. В системі підготовки фахівця дисципліна займає особливе місце, оскільки засвоєння курсу складає перший найважливіший крок, абсолютно необхідний для успішного засвоєння подальших фахових дисциплін.

Від студентів не вимагається попереднього досвіду в галузі програмування та інформатики.

Завдання навчальної дисципліни

За результатами вивчення дисципліни студент повинен продемонструвати такі результати навчання:

– володіння поняттями мови програмування, середовища програмування; елементів мови програмування - тип даних, класифікація типів даних, базові структури даних; основні алгоритмічні конструкції та їх реалізація засобами мови програмування C/C++;

– знання простих алгоритмів пошуку і сортування даних;

– навики написання та налагодження програм в процедурно-орієнтованому стилі;

– володіння технологією розроблення програмного забезпечення відповідно до вимог і обмежень замовника.

– вміння ефективно працювати як автономно, так і у складі команди, відповідально ставитись до виконуваної роботи та досягти поставленої мети, приймати обґрунтовані рішення та оцінювати їх наслідки.

– користуватися стандартною бібліотекою мови, можливостями середовища програмування при написанні та налагодженні програм.

Вивчення навчальної дисципліни передбачає формування та розвиток у студентів компетентностей:

загальних:

– базові знання фундаментальних наук в обсязі, необхідному для освоєння загально-професійних дисциплін.

– Здатність виявляти, ставити та вирішувати проблеми, приймати обґрунтовані рішення.

– Здатність до пошуку, оброблення та аналізу інформації з різних джерел, проведення досліджень на відповідному рівні.

– Здатність оцінювати та забезпечувати якість виконуваних робіт.

– Здатність працювати як індивідуально, так і в команді, мотивувати людей та рухатися до спільної мети.

– Здатність аргументовано переконувати колег у правильності пропонованого рішення, вміти донести до інших свою позицію.

– Здатність використовувати можливості мережевих програмних систем.

– Базові уявлення про сучасні стандарти та процеси управління якістю програмного забезпечення.

– Дотримання професійної етики програмної інженерії.

фахових:

– володіння основами конструювання програмного забезпечення.

– Здатність аналізувати, проектувати та прототипувати людино-машинний інтерфейс.

– Здатність здійснювати аналіз вимог, розробляти специфікацію програмних вимог, виконувати їхню верифікацію та атестацію.

– Здатність розв'язувати математичні, фізичні та економічні задачі шляхом створення відповідних застосувань.

– Здатність розробляти алгоритми та структури даних для програмних продуктів.

– Здатність створення технічної документації до програмного проекту.

Завданням лекційних занять є ознайомлення студентів з основними принципами програмування, базовими елементами мови програмування С/С++, засобами їх реалізації, особливостями використання операторів мови.

Завдання лабораторних занять полягає у вивченні студентами основних прийомів програмування.

Знання, вміння та навички, отримані студентами під час вивчення даної дисципліни будуть необхідними для їхньої професійної діяльності, а також використовуватимуться при написанні курсових та дипломних проектів з програмної інженерії.

У цих методичних вказівках для полегшення засвоєння матеріалу приведені контрольні питання для самоперевірки.

# 1 СТРУКТУРА НАВЧАЛЬНОЇ ДИСЦИПЛІНИ ЗА МОДУЛЬНОЮ СИСТЕМОЮ

## 1.1 Розподіл годин

Для вивчення навчальної дисципліни “Основи програмування” відводиться: 240 год (8 кредитів) для денної форми навчання і 270 год (9 кредитів) – для заочної.

Навчальна дисципліна об’єднує в собі три складові навчання: теорію, практику і контроль засвоєння матеріалу. Структура навчальної дисципліни наведена в таблиці 1.

Курс наповнений стислими теоретичними відомостями, систематизованими завданнями до лабораторних робіт, перевіреними прикладами програмної реалізації та наборами контрольних тестів.

Таблиця 1 - Структура навчальної дисципліни

| Показник | Всього годин | |
|---|---|---|
| | Денна форма навчання | Заочна (дистанційна) форма навчання |
| Кількість кредитів/годин | 8 | 9 |
| Аудиторні заняття, год. | 128 | 32 |
| Самостійна робота, год. | 112 | 238 |
| Аудиторні заняття: | | |
| - лекції, год. | 64 | 16 |
| - лабораторні заняття, год. | 64 | 16 |
| Самостійна робота: | | |
| - підготовка до лабораторних занять | 32 | 32 |
| - опрацювання окремих розділів програми, які не виносяться на лекції | 32 | 160 |
| - підготовка та складання заліків, екзаменів, тестування | 48 | 46 |
| Екзамен | 1 | 1 |

Частка годин самостійної роботи студента:

денна форма навчання - 47 %;

заочна (дистанційна) форма навчання -88%.

1.2 Тематика лекційних занять

Робоча програма з дисципліни містить 2 змістових модулі. Перший змістовий модуль «Організація програм» включає 9 тем, другий змістовий модуль «Структура даних» - 6 тем.

Погодинний розподіл і тематику лекційних занять курсу наведено у таблиці 2.

Таблиця 2 – Лекційні заняття

| № | Тема заняття та короткий зміст | Кількість годин | |
|---|---|---|---|
| | | ДФН | ЗФН |
| 1. | Тема1. Вступ до дисципліни. Основні поняття та визначення<br><br>Призначення курсу. Структура курсу. Література. Вимоги і критерії оцінювання. Основні терміни. Принципи програмного керування та адресації. Поняття алгоритму та основні алгоритмічні структури. Властивості та способи опису алгоритму. Алгоритмічна структура розгалуження. Алгоритмічна структура повторення. Алгоритмічні | 4 | |
| 2. | Тема2. Елементи мови<br><br>Історія виникнення і розвитку мови C/C++. Поняття програми. Структура програми мовою C++: поняття препроцесора, функції. Структура функцій. Алфавіт, слова. Правила побудови ідентифікаторів. Типи даних. Область видимості. Поняття виразу. Арифметичні операції. Класифікація операцій. Пріоритет операцій. Оператори присвоєння. Інкремент і декремент. Оператор sizeof. | 4 | 1 |
| 3. | Тема3. Організація введення-виведення в C/C++<br><br>Введення-виведення символів і рядків. Форматне введення-виведення з допомогою функцій бібліотеки stdio. Поняття потокового введення-виведення з допомогою об'єктів cin, cout класу ios. Деякі маніпулятори форматування потоку виведення. | 4 | 1 |
| 4. | Тема4. Базові структури алгоритмів і їх реалізація засобами мови C/C++<br><br>Оператори: порожній оператор і блок. Оператори порівняння. Оператор безумовного переходу. Умовний оператор, умовний вираз. Оператор переключення. Оператори циклу. Операція приєднання. Оператор продовження. Оператор повернення значень. Оператор переривання. Особливості написання програм з використанням конструкцій переходу. Особливості написання програм з використанням конструкцій повторень. Поєднання циклів і розгалужень. Вкладені цикли. | 4 | 1 |

| | | | |
|---|---|---|---|
| 5. | Тема5. Процедурно-орієнтоване програмування. Введення у функції<br><br>Цілі структурного програмування. Принципи структурної методології. Стандарти структурного програмування. Стандартні бібліотечні функції. Функції користувача: оголошення і визначення функції. Тип void. Область дії змінних. Класи пам'яті. Виклик функції. | 4 | |
| 6. | Тема6. Функції<br><br>Перевантаження функцій. Значення параметрів по замовчуванню. Функції зі змінним числом параметрів. Вбудовані функції. Оператори розподілу динамічної пам'яті. Оператор дозволу області видимості (розширення області дії). Проектування функцій програмної системи. Створення користувацьких функцій. Поняття рекурсії і основні визначення. Форми рекурсивних процедур. | 4 | 2 |
| 7. | Тема7. Вказівники. Посилання<br><br>Поняття вказівника. Оголошення. Арифметика з вказівниками. Використання вказівників у функціях. Поняття даних типу посилання. Основні відмінності вказівників і посилань. Параметри-посилання. Функції, що повертають значення типу посилання. Застосування динамічних даних у функціях. | 4 | 1 |
| 8. | Тема8. Методології розробки програм, поняття модульного програмування<br><br>Поняття низхідного і висхідного проектування. Створення, компіляція та підключення власних модулів. Реалізація меню програми. | 2 | |
| 9. | Тема9. Директиви препроцесора<br><br>Поняття директив препроцесора. Директиви включення. Директиви макровизначення. Директиви умовної компіляції. Оператор defined. Директива #ERROR. Директива #pragma. Попередньо визначені макроси. Моделі пам'яті, дальні та ближні вказівники, макроси для роботи з вказівниками. | 2 | 1 |
| 10. | Тема10. Організація даних та алгоритми їх обробки. Масиви<br><br>Оголошення і ініціалізація цифрових масивів. Доступ до елементів масивів. Вказівники масивів. Масиви як параметри функцій. Масиви вказівників. Використання масивів у функціях. Матриці. | 6 | 2 |
| 11. | Тема11. Символьні масиви<br><br>Оголошення і ініціалізація рядків. Вбудовані функції обробки рядків. Передача рядків у функції. | 4 | 2 |

| | | | |
|---|---|---|---|
| 12. | Тема12. Типи даних користувача<br><br>Структури, оголошення і ініціалізація. Вказівники на структуру. Доступ до елементів структури. Структури як параметри функцій. Масиви структур. Перейменування типів. Типи, що задаються переліком. Об'єднання. Бітові поля. Особливості застосування структур при написанні програм. | 6 | 2 |
| 13. | Тема13. Файлові структури даних<br><br>Файли: структура логічного і фізичного файлів, класифікація файлів за внутрішньою організацією. Оголошення даних файлового типу. Функції відкриття і закриття файлу. Функції запису до файлу; створення нового файлу; функції читання з файлу. Реалізація основних кроків роботи з файлами засобами C/C++. | 4 | 2 |
| 14. | Тема14. Способи організації доступу до файлів.<br><br>Поняття послідовного і прямого доступів до файлу. Функції для реалізації довільного доступу до файлів. Обробка текстових і бінарних файлів. | 6 | |
| 15. | Тема15. Алгоритмізація типових обчислювальних задач<br><br>Алгоритми сортування. Алгоритми пошуку. Теорія чисел: прості числа: пошук і підрахунок, подільність чисел, арифметика остач. Комбінаторика. Бітові операції і їх застосування до рішення задач. | 6 | 1 |
| Усього годин | | 64 | 16 |

1.3 Тематика лабораторних занять

Тематика лабораторних робіт наведена в таблиці 3.

Завдання до лабораторних робіт підібрані індивідуально, за кількістю студентів, виходячи з їхнього початкового рівня підготовки в галузі програмування, реалізація яких, надіюсь, сприяє формуванню логічного мислення, та культури програмування, формує творчий стиль програміста та закладає наукові основи оволодіння сучасними технологіями програмування.

Таблиця 4 – Лабораторні заняття

| № | Тема заняття | Кількість годин | |
|---|---|---|---|
| | | ДФН | ЗФН |
| 1 | Створення найпростіших діалогових програм. Реалізація лінійного алгоритму | 6 | - |
| 2 | Умовні оператори | 6 | - |

| | | | |
|---|---|---|---|
| 3 | Цикли | 6 | 2 |
| 4 | Застосування динамічних змінних до рішення задач | 6 | 2 |
| 5 | Реалізація функцій користувача засобами мови С/С++<br>Перевантажені, вбудовані функції. Рекурсія | 8 | 2 |
| 6 | Одновимірні, двовимірні масиви | 6 | 2 |
| 7 | Стрічки | 6 | 2 |
| 8 | Структури та об’єднання | 6 | 2 |
| 9 | Файли | 8 | 2 |
| 10 | Реалізація типових обчислювальних задач | 6 | 2 |
| Усього годин | | 64 | 16 |

1.4. Теми для самостійного вивчення

Самостійна робота студента є основним видом засвоєння навчального матеріалу у вільний від аудиторних занять час.

Метою самостійної роботи є вироблення студентами навичок і вміння працювати з літературою, віднаходити головні аспекти проблем, що потребують стійкого засвоєння.

Предметом самостійної роботи студентів є опрацювання ними як окремих тем програми курсу в цілому, так і деяких розділів тем, самостійне розв’язування конкретних задач з метою практичного поглиблення здобутих знань.

Перевірка рівня засвоєння матеріалу самостійно опрацьованих тем здійснюється при проведенні поточного тестового контролю з окремих тем згідно програми та під час написання домашніх і лабораторних робіт.

Таблиця 4 – Тематика самостійної роботи з курсу

| № | Найменування робіт | Кількість Годин | |
|---|---|---|---|
| | | ДФН | ЗФН |
| 1 | Опрацювання лекційного матеріалу | 32 | 8 |
| 2 | Розв’язування індивідуальних завдань лабораторних робіт. | 20 | 20 |
| 3 | Оформлення звітів, підготовка лабораторних робіт до захисту. | 12 | 12 |
| 4 | Підготовка до складання екзамену, тестування. | 16 | 38 |
| 5 | Опрацювання окремих розділів програми, які не виносяться на | 0 | 0 |

| | | | |
|---|---|---|---|
| | лекції: | | |
| 5.1 | Стандартизація мови С. Консольні додатки. Використання IDE: опції панелі інструментів, проекти і рішення. Настроювання опцій середовища. | 2 | 12 |
| 5.2 | Порозрядні логічні операції. Операції зсуву вліво і вправо. | 2 | 20 |
| 5.3 | Функції стандартної бібліотеки. Генерація випадкових чисел. Класи пам'яті. Функції дати, часу і локалізації. Робота з датами, операції з бітами і байтами (підрахунок, зсув, дзеркальне відображення бітів). | 4 | 20 |
| 5.4 | Рекурсія в порівнянні з ітерацією. Непряма рекурсія. Рекурсія і час життя даних. | 2 | 14 |
| 5.5 | Оголошення вказівників на функції. Використання модифікатора const з вказівниками. Порівняльний аналіз вказівників і посилань. | 2 | 16 |
| 5.6 | Параметри функції main. | 2 | 6 |
| 5.7 | Директива #pragma. Попередньо визначені макроси. Моделі пам’яті для роботи з вказівниками, дальні та ближні вказівники. | 2 | 8 |
| 5.8 | Динамічне виділення пам’яті. Функції динамічного розподілу пам’яті malloc, calloc, free, оператори new і delete | 4 | 16 |
| 5.9 | Деякі функції бібліотеки <string>. Пошук символів. Пошук підрядків. Інверсія рядків. | 4 | 16 |
| 5.10 | Вказівники на структури. Об'єднання і операції з ними. Бітові поля. | 2 | 16 |
| 5.11 | Функції позиціювання у файлі. | 2 | 16 |
| 5.12 | Основи аналізу алгоритмів. Масиви змінного розміру. | 4 | 0 |
| | Разом | 112 | 238 |

1.5 Контрольні питання для самоперевірки

Питання для самоперевірки складені за матеріалами всієї дисципліни “Основи програмування” і є для студентів допоміжним засобом вивчення пропонованого курсу. Нижче приводяться складені питання щодо дисципліни.

1. Етапи розв’язування задач на ЕОМ.
2. Оперативна пам'ять процесора. Регістри процесора. Представлення інформації в пам’яті ПК: представлення чисел з фіксованою і плаваючою точкою.
3. Історія виникнення мови С/С++.

4. Елементи мови С: алфавіт, ідентифікатори, константи, коментарі.
5. Структура програми мовою С/С++; поняття препроцесора, функції.
6. Загальний формат оголошення типів даних, ініціалізація даних; область видимості.
7. Область дії змінних. Класи пам’яті: auto, extern, register, static.
8. Функція scanf, загальний формат, форматування введення. Приклад.
9. Функція printf, загальний формат, форматування виведення. Приклад.
10. Функції виведення символів і рядків: загальний формат, особливості застосування.
11. Функції введення символів і рядків: загальний формат, особливості застосування.
12. Реалізація потокового введення-виведення.
13. Основні операції: арифметичні операції, операції присвоєння, операції порівняння. Пріоритет операцій.
14. Логічні операції, операція слідування (кома), умовна операція ?:, операція sizeof(). Їх призначення, загальний формат, дія, приклад.
15. Вирази і їх запис мовою С/С++. Приклади.
16. Оператори: порожній оператор і блок.
17. Умовний оператор, умовний вираз. Загальний формат, дія оператора, приклад використання.
18. Оператор переключення. Загальний формат, дія оператора, особливості застосування. Приклад.
19. Оператори циклу, їх види, особливості застосування.
20. Оператор циклу з лічильником. Загальний формат, дія оператора, приклад використання.
21. Оператор циклу з передумовою. Загальний формат, дія оператора, приклад використання.
22. Оператор циклу з післяумовою. Загальний формат, дія оператора, приклад використання.
23. Оператор повернення значень.
24. Поняття перевантаження функцій. Способи перевантаження функцій. Приклади.
25. Вбудовані функції. Загальний формат. Приклад.
26. Функції із змінним числом параметрів. Загальний формат, приклад використання.
27. Функції користувача: оголошення і визначення функції. Тип void. Виклик функції.
28. Функції, що не повертають значення. Передача параметрів у функцію.
29. Основні відомості про вказівники: поняття вказівника, оголошення, приклад.

30. Базові типи даних.

31. Операції з вказівниками, приклади.

32. Арифметика з вказівниками. Проблеми, пов’язані з вказівниками.

33. Використання вказівників у функціях в якості параметрів і результату.

34. Поняття даних типу посилання. Основні відмінності вказівників і посилань.

35. Порівняльна характеристика вказівників і посилань.

36. Параметри-посилання. Функції, що повертають значення типу посилання.

37. Поняття складених типів даних і їх класифікація.

38. Поняття масиву, властивості масивів. Розміщення масивів у пам’яті комп’ютера. Назва масиву, індексація масиву, ідентифікатори і значення елементів масиву.

39. Оголошення, способи ініціалізації цифрових масивів.

40. Доступ до елементів масивів. Вказівники масивів. Доступ до елемента масиву з допомогою індексації вказівника, зміщення вказівника.

41. Масиви як параметри функцій. Масиви вказівників.

42. Динамічний масив, оператори виділення і вивільнення пам’яті під масив.

43. Типові задачі на роботу з масивами.

44. Робота з одновимірними масивами.

45. Операції з матрицями: обчислення суми, добутку, максимуму, мінімуму кожного стовпця (рядка), виділення головної і бічної діагоналей, перестановка заданих рядків (стовпців).

46. Використання масивів у функціях (як параметр і як тип функції).

47. Оголошення, способи ініціалізації рядків. Вбудовані функції обробки рядків: функції введення, виведення рядків, визначення довжини рядка, порівняння, копіювання, конкатенації рядків, розбиття рядка на лексеми, пошук символу, підрядка у рядку . Передача рядків у функції. Рядок як результат виконання функції.

48. Структури, оголошення і ініціалізація. Елемент структури, змінна типу структури. Вказівники на структуру. Доступ до елементів структури. Операції над структурами. Копіювання структури, динамічне виділення пам’яті під структуру. Вкладені структури, масиви структур.

49. Структури як параметри функцій. Повернення значень типу структура.

50. Перейменування типів. Типи, що задаються переліком.

51. Об’єднання. Бітові поля.

52. Файли: структура логічного і фізичного файлів, класифікація файлів, способи організації доступу до файлів.

53. Оголошення даних файлового типу. Функції відкриття і закриття файлу. Функції запису до файлу; створення нового файлу; функції читання з файлу.

54. Функції для реалізації прямого доступу до файлів.

55. Обробка текстових і бінарних файлів.

56. Поняття директив препроцесора. Директиви включення.

57. Директиви макровизначення. Директиви умовної компіляції.

58. Оператор defined. Директива #ERROR. Директива #pragma.

59. Алгоритмізація типових обчислювальних задач

60. Алгоритми сортування. Алгоритми пошуку. Теорія чисел: прості числа: пошук і підрахунок, подільність чисел, арифметика остач. Комбінаторика.

# 2 КОНТРОЛЬ ЗНАНЬ СТУДЕНТІВ

Контроль знань студентів включає в себе – виконання індивідуальних завдань, оформлення звіту з лабораторних робіт, захист звітів із лабораторних робіт, поточне комп'ютерне тестування з кожного модуля, підсумковий модульний контроль з дисципліни. Підсумковий модульний контроль передбачає комп'ютерне тестування знань теоретичних питань з дисципліни і рішення практичного завдання згідно варіанту.

## 2.1 Критерії оцінювання

Підсумкова рейтингова оцінка студента - це сума набраних балів за такі види робіт:

– лабораторна робота; виконання згідно завдання, оформлення звіту і своєчасний захист лабораторної роботи оцінюється максимально – 4бали,

– тематичне тестування; поточний тест з теми – 1бал. Сумарно за виконання цього виду роботи студент може набрати не більше 50% від підсумкової рейтингової оцінки.

– модульний контроль. Модульні контролі (їх є два)- оцінюються максимально сумарно 25% від підсумкової рейтингової оцінки.

Модульний контроль передбачає комп'ютерне тестування і виконання практичного завдання (рішення однієї із трьох запропонованих задач різного рівня складності) за вибором студента.

Підсумковий модульний контроль оцінюється максимально в 25% від підсумкової рейтингової оцінки

## 2.2 Типові тести для контролю знань

В базі тестування з дисципліни є понад 500 тестових завдань, які включають тематичні, модульні і підсумкові тести. Наприклад:

Вказати, яке з наступних тверджень неправильне:

- для вказівників відводиться додаткова пам'ять;
- після ініціалізації значення вказівника не може змінюватись (стати посиланням на інший об'єкт);
- не можна оголосити масив посилань.

Вказати, яке з наступних тверджень правильне:

- посилання може мати тип void;
- початковим значенням вказівника може бути: 0, null або адреса об'єкта того ж типу;
- допускаються посилання на бітові поля.

Нехай оголошено float *nPtr, numbers[size]={0., 1., 2., 3., 4.}; nPtr=numbers; Друк адрес елементів масиву з використанням вказівника на масив і зміщення має наступний вигляд:

- for (int i=0; i<size; i++) cout <<nPtr[i];
- for (int i=0; i<size; i++) cout <<&nPtr[i];
- for (int i=0; i<size; i++) cout <<(nPtr+i);

Нехай оголошено float *nPtr, numbers[size]={0.0, 1.1, 2.2, 3.3, 4.4}; nPtr=numbers; Друк елементів масиву з використанням імені масиву і зміщення має наступний вигляд:

- for (int i=0; i<size; i++) cout <<nPtr[i]<<' ';
- for (int i=0; i<<size; i++) cout <<*(nPtr+i)<<' ';
- for (int i=0; i<size; i++) cout <<*(numbers+i) <<' ';

Нехай оголошено float *nPtr, numbers[]={0., 1., 2., 3., 4.}; Тоді наступний фрагмент програми nPtr=numbers; int i=0; cout <<setw(6)<<*(nPtr+i); здійснюватиме:

- друк всіх елементів масиву з використанням вказівника і зміщення;
- друк всіх елементів масиву з використанням індексації вказівника;
- друк першого елемента масиву з використанням вказівника і зміщення.

Логічна операція «||» повертає фальш, якщо:

- хоча б один з операндів хибний;
- обидва операнди хибні;
- обидва операнди істинні.

Визначте значення виразу 1 / 12 + 23 % 2

- 1
- 2
- 1,083

Визначте значення виразу: 0 && (1 % 2 != 1)

- 0
- 1
- true

Вкажіть помилку в наступному фрагменті коду:

*int p; do (p < 10) cout << p; while;*

- умову виходу з циклу записують після while;
- відсутні фігурні дужки;
- умова записується без дужок.

Скільки чисел буде виведено в результаті роботи наступного коду:

*int p; for(p = 0; p <= 10; p++) cout <<p;*

- 10
- 9
- 11

Якщо у функцію як аргумент передана змінна за значенням, то модифікація цієї змінної всередині тіла функції:

- приведе до її зміни і поза тілом функції;
- не приведе до її зміни поза тілом функції;
- приведе до її зміни поза тілом функції, але тільки при відповідних настройках директив компілятора.

Що виконує наступна функція:

```
void Rec(int n){
    if (n > 1) Rec(n / 2);
    printf("%d",n%2);
}
```

- переводить десяткове число n у двійкове;
- перевіряє, чи число n ділиться на 2;
- визначає остачу від ділення числа n на 2.

Для читання з файлу використовують функції:

- fgetc(f); fgets(s,n,f); fread (p, size, n, f); fscanf(f,format,список адрес змінних);
- getc(f); gets(s,n,f); read (p, size, n, f); scanf(f,format,список адрес змінних);
- fprintf (f, format, список аргументів); fputc (ch, f); fwrite (p, size, n, f);

Які значення будуть мати елементи масиву a після виконання наступного фрагменту коду:

```
int p; int a[5];
for(p = 0; p < 5; p++)
  a[p] = 4 - p;
a[0] = a[0] + a[4]; a[4] = a[0] – a[4];a[0] = a[0] – a[4];
```

- 0 3 2 1 4
- 1 2 3 4 5
- 5 4 3 2 1

Функція вигляду <тип> <ім’я> (<параметр>, <параметр=<значення>); називається:

- функцією з параметрами, заданими по замовчуванню;
- перевантаженою функцією;
- функцією зі змінним числом параметрів.

Режим “w+”, заданий у функції fopen, призначений для:

- створення нового файлу для читання та запису, якщо файл із вказаним ім'ям вже існує, то він перезаписується;
- відкриття файлу тільки для дозапису інформації в кінець файлу, якщо файл не існує, він створюється;
- відкриває файл у режимі читання та запису для додавання нової інформації у кінець файлу; якщо файл не існує, він створюється.

Нехай оголошено структуру:

```
struct customer {
   char firstName[15], lastName[15];
   struct A{
       char phoneNumber[11], city[15];
  } personal;
} *customerPtr;
```

Як здійснити доступ до елемента city з допомогою вказівника customerPtr:

- customerPtr. personal. city
- customerPtr->personal. city
- customerPtr. -> personal-> city

Перевантажені функції застосовуються тоді, коли:

- необхідно змоделювати вкладеність функцій;
- функція повинна виконувати різні дії залежно від типу і кількості її параметрів;
- потрібно оголосити глобальну функцію.

Якщо функції відрізняються типом або кількістю параметрів, то:

- їх можна перевантажувати;

- їх не можна перевантажувати;
- можливість їх перевантаження залежить від настройок директив компілятора.

Що виведеться на екран в результаті виконання фрагменту програми:

```
int *p1,p2(2);
p1=&p2;
printf("%p",p1);
```

- адреса p2
- 2
- синтаксична помилка

Що виведеться на екран в результаті виконання фрагменту програми?

*char q1[10] = "qwerty", q2[10] = "01234"; strcpy ( q1+2, q2 ); printf("%s",q1);*

- qw01234
- qwer01234
- 012345qw

Вкажіть допустимі способи передачі двомірного масиву у функцію, якщо typedef int TR[5];

- void print1(TR ar) { }, void print6(int ar[][5]) { }
- void print5(int ar[5][]) { }, void print4(int ar[5][5]) {}
- void print3(int ar[][]) { }, void print2(int **ar) { }

Нехай enum E { left, right, ctrl, alt, shift }; E but; Як присвоїти значення змінній but?

- but="ctrl";
- but=right;
- E->but=alt;

Що надрукується?

```
#define A(a) # a
#define B(a, b) A(a ## b ## c)
int main() {
  printf("%s", B(foo, bar));
  return 0;
}
```

- foobarc
- foobar

– foo ## bar ## c

Для наступного оголошення структури

```
struct data{
   int day, month, year;
};
struct field{
  char tools[20], type[20];
  data *taken, done;
   int term;
}*array=new field[size];
```

виберіть правильне звернення до її елемента:

– A) array[1].tools
– Б) array[1].taken.year
– В) array[1].taken.*(month)

## 2.3 Типові практичні завдання до модульних і екзаменаційного контролів

Алгоритм проходження модульного контролю наближено наступний: студентам пропонується пройти тест, розрахований на 30 хвилин, який включає 25 питань змістового модуля. Другою складовою є завдання практичного плану. Кожен варіант практичного завдання включає 3 рейтингові задачі, з яких студент обирає одну. Приклади варіантів завдань наведено нижче.

Варіант1.

1) Дано натуральне число n. Написати програму обчислення значення виразу:

$$\cos\pi+\cos\frac{\pi}{2}+\cos\frac{\pi}{4}+\ldots+\cos\frac{\pi}{2^n}.$$

2) Число, рівне сумі всіх своїх дільників, включаючи одиницю, називається досконалим. Знайти і надрукувати всі досконалі числа, менші за b.

3) Дано 4 цілих числа. Знайти їх найбільший спільний дільник, використовуючи алгоритм Евкліда (рекурсія).

Варіант2.

1) Обчислити значення виразу: $\frac{1}{\sin 1}*\frac{1+2}{\sin 1+\sin 2}*\ldots*\frac{1+2+\ldots+n}{\sin 1+\ldots+\sin n}$

2) У n-значному числі визначити найменшу цифру.
3) Написати рекурсивну функцію, що визначає, чи є задане натуральне число простим.

Варіант3.

1) В одновимірному масиві, що складається з N дійсних елементів, обчислити суму елементів масиву, розміщених між першим і останнім додатними елементами.

2) Написати програму для збереження інформації про товари на складі: назву товару; кількість товару; ціну одиниці товару. Вивести на екран загальну вартість товару, назва якого введена з клавіатури; якщо такого товару на складі немає, то вивести відповідне повідомлення.

3) Інформація про товари на складі (назва товару; кількість товару; ціна одиниці товару) зберігається у файлі. Вивести на екран загальну вартість товару, назва якого введена з клавіатури; якщо такого товару на складі немає, то вивести відповідне повідомлення.

2.5 Критерії оцінювання результатів навчання студентів

Форма підсумкового семестрового контролю – екзамен

| Модуль 1 | | | Модуль 2 | | | Підсумковий контроль | | Разом |
|---|---|---|---|---|---|---|---|---|
| Аудиторна та самостійна робота | | | Аудиторна та самостійна робота | | | | | |
| Теоретичний курс (тестування) | Практична робота | | Теоретичний курс (тестування) | Практична робота | | | | |
| **20** | **20** | | **15** | **20** | | **25** | | **100** |
| № лекції | Вид робіт | Бал | № теми | Вид робіт | Бал | Теоретичне завдання | 10 | |
| Лекція1 | | | Лекція10 | Лаб. роб.6 | 5 | Практичне завдання | 15 | |
| Лекція2 | | | Лекція11 | Лаб. роб.7 | 5 | | | |
| Лекція3 | Лаб. роб.1 | 4 | Лекція11 | Лаб. роб.8 | 5 | | | |
| Лекція4 | Лаб. роб.2 | 4 | Лекція13 | Лаб. роб.9 | 5 | | | |
| Лекція5 | Лаб. роб.3 | 4 | Лекція14 | | | | | |
| Лекція6 | Лаб. роб.4 | 4 | Лекція15 | | | | | |
| Лекція7 | Лаб. роб.4 | 4 | | | | | | |
| Лекція8 | Лаб. роб.5 | 4 | | | | | | |
| Лекція9 | | | | | | | | |

## НАВЧАЛЬНО-МЕТОДИЧНЕ ЗАБЕЗПЕЧЕННЯ

1. «Основи програмування» для студентів напряму 6.050103 – Програмна інженерія. / Уклад. Петрик О.Ю. –Тернопіль: ТНТУ імені Івана Пулюя, 2014. [Електронний ресурс]. - Режим доступу: URL: https://dl.tntu.edu.ua/content.php? course=1060
2. О. Петрик, М. Петрик. Технологія програмування в системі С++. Лабораторний практикум. Тернопіль, видавництво ТДТУ ім. Ів. Пулюя, 2005 - 107с.
3. Методичні вказівки щодо самостійної роботи студентів та модульного контролю знань з дисципліни “Основи програмування” для студентів напряму підготовки 6.050103 – Програмна інженерія. / Уклад.: М.Петрик, О.Петрик - Тернопіль: ТНТУ 2015 - 22 с. [Електронний ресурс]. - Режим доступу: URL: http://elartu.tntu.edu.ua/handle/123456789/17787
4. Основи програмування. Курс лекцій для студентів першого рівня вищої освіти за спеціальністю No 121 Інженерія програмного забезпечення/ Уклад.: М.Р. Петрик, О.Ю.Петрик - Тернопіль: ТНТУ 2018- 64 с. . [Електронний ресурс]. - Режим доступу: URL: http://elartu.tntu.edu.ua/handle/lib/26027

## РЕКОМЕНДОВАНА ЛІТЕРАТУРА

Базова


5. Шпак З.Я. Програмування мовою С. – Львів: Видавництво львівської політехніки, 2011.-436с. [Електронний ресурс]. - Режим доступу: URL:
6. http://programming.in.ua/programming/basisprogramming/275-programming-c-book-shpak.html
7. Ковалюк Т. В. Алгоритмізація та програмування: Підручник. — Львів: «Магнолія 2006», 2013. — 400 с[Електронний ресурс]. - Режим доступу: URL:
8. http://library.kpi.kharkov.ua/files/new_postupleniya/kovaluk.pdf
9. Шилдт Г. С++: базовый курс, 3-е изд. : Пер. с. англ. – М.:Изд. дом «Вильямс», 2010. – 624с.
10. Страуструп Б. Дизайн и эволюция С++: Пер. с англ. - М.:ДМК Пресс; СПб: Питер, 2016 - 446 с.
11. Айвор Хортон. Visual С++ 2010: полный курс. Изд-во: Диалектика-Вильямс, 2011. – 1216 с.
12. Глушаков С.В., Коваль А.В., Смирнов С.В. Язык программирования С++: Учебный курс.- Харьков: Фолио; М.:ООО "Издательство АСТ", 2001.-500 с.
13. Дейтел Х., Дейтел П. Как программировать на С++. 5-е изд. - М.:ООО „Бином-Пресс”, 2008г.–1456с
14. Романов Е.Л. Практикум по программированию на С++: Учебное пособие. СПб: БХВ-Петербург, Новосибирск: Изд-во НГТУ, 2004. – 432 с.
15. Семакин И.Г., Шестаков А.П. Основы программирования: Учебник. - М.: Мастерство,2002.- 432с.


Допоміжна


16. Златопольский Д.М. Сборник задач по программированию. - 2-е изд. СПб.: БХВ - Петрербург, 2007. - 240 с.

17. Караванова Т.П. Інформатика: основи алгоритмізації та програмування: 777 задач з рекомендаціями та прикладами: Навч. посіб. для 8-9 кл. із поглибл. вивч. інформатики – К.: Генеза. – 2006.- 286 с.
18. Кнут Д. Искусство программирования. т.2. Получисленные алгоритмы.- М., СПб., К.:Вильямс, 2000.- 832 с.
19. Прата С. Язык программирования C++. Лекции и упражнения. 5-е изд. –М.: ООО «И. Д. Вильямс», 2007 – 1184с.
20. Шилдт Г. Полный справочник по C++. 4-е издание. - М.: Изд.дом «Вильямс», 2006г.–800с.
21. Шилдт Г. C++: руководство для начинающих, 2-е изд.: Пер. с. англ. – М.:Изд.дом «Вильямс», 2005. – 672с.

Інформаційні ресурси

22. Основи програмування на мовах C та C++ для початківців [Електронний ресурс]. - Режим доступу: URL: http://cppstudio.com/uk/
23. Довідник по C++ [Електронний ресурс]. - Режим доступу: URL: https://msdn.microsoft.com/uk-ua/library/3bstk3k5.aspx
24. Система для проведення дистанційних олімпіад та змагань зі спортивного програмування [Електронний ресурс]. - Режим доступу: URL: https://www.e-olymp.com/uk/
25. Система онлайн-тестування [Електронний ресурс]. - Режим доступу: URL: http://www.quizful.net/category/cpp
26. IDE Visual C++ [Електронний ресурс]. - Режим доступу: URL https://support.microsoft.com/uk-ua/help/2977003/the-latest-supported-visual-c-downloads