Міністерство освіти і науки України

Вінницький національний технічний університет

МЕТОДИЧНІ ВКАЗІВКИ

до виконання курсової роботи з дисципліни
"Програмування"
для студентів другої вищої освіти з напрямів підготовки
"Комп'ютерна інженерія" та "Інформаційна безпека"

Вінниця ВНТУ 2017

Рецензенти:
Ю. В. Булига, кантидат технічних наук, доцент кафедри ГМ
В. П. Майданюк, кандидат технічних наук, доцент кафедри ПЗ

Методичні вказівки до виконання курсової роботи з дисципліни "Програмування" для студентів другої вищої освіти з напрямів підготовки "Комп'ютерна інженерія" та "Інформаційна безпека" / Уклад. О. Д. Азаров, О. І. Черняк, Л. А. Савицька – Вінниця : ВНТУ, 2017 – 26 с.

У методичних вказівках викладено вимоги до змісту розділів пояснювальної записки до курсової роботи та її оформлення. Наведено завдання до курсової роботи, зразки оформлення титульного аркуша та індивідуального завдання. Надано список літератури. Методичні вказівки можуть використовуватись для виконання контрольних робіт студентами заочної форми навчання та для самостійної роботи студентів

ЗМІСТ

1.ВСТУП.....4
2.РЕКОМЕНДАЦІЇ ЩОДО ОФОРМЛЕННЯ КУРСОВОЇ РОБОТИ.....5
2.1.Правила оформлення пояснювальної записки.....5
2.2.Структура пояснювальної записки.....5
3.ВАРІАНТИ ЗАВДАНЬ ДО КУРСОВОЇ РОБОТИ.....8
4.ДОВІДКОВА ІНФОРМАЦІЯ.....13
5.РЕКОМЕНДОВАНА ЛІТЕРАТУРА.....23
ДОДАТОК А.....24
ДОДАТОК Б.....25

1. ВСТУП

Дані методичні вказівки призначені для виконання курсові роботи з дисципліни "Програмування".

У результаті виконання курсової роботи студенти повинні отримати практичні навички зі створення програм мовою С++ з використанням технологій процедурного і структурного програмування.

Виконання даної курсової роботи сприяє засвоєнню студентами таких теоретичних питань: вбудовані типи даних, прості і складні змінні, вказівники і посилання, вказівники і масиви, робота з текстовими рядками, константи і перерахування, структури і об’єднання, операції мови С++, прості, умовні та циклічні оператори, функції та макроси, використання стандартних функцій при роботі з даними, параметри командного рядка, робота з файлами, динамічне створення одно- і багатовимірних масивів, робота зі списками та деревами, стеки і черги.

Для виконання курсової роботи необхідно знати будову комп’ютера та принципи організації і обробки інформації, а також мати практичні навички з роботи у середовищі Visual C++. На комп’ютері повинна бути встановлена операційна система Windows і середовище програмування Visual Studio.

У процесі курсового проектування студент отримує від викладача варіант завдання, описує основні теоретичні відомості, розробляє та відлагоджує програму мовою С++ з використанням структур і функцій, а також перевіряє роботу програми на тестовому прикладі. Після цього студент оформлює пояснювальну записку до курсової роботи відповідно до вимог, наведених у даних методичних вказівках.

Кожне завдання полягає в обробці даних, що описують певні геометричні фігури: точки, трикутники, кола, відрізки, чотирикутники та інше. Виконання завдання потребує розробки методу вирішення і написання програми.

Кожна програма складається з таких частин: зчитування даних з файлу, вирішення завдання відповідно до варіанту та виведення результатів на екран.

На захист студент подає працездатну програму, файл з вхідними даними та пояснювальну записку, оформлену відповідно до вимог, наведених у даних методичних вказівках.

2. РЕКОМЕНДАЦІЇ ЩОДО ОФОРМЛЕННЯ КУРСОВОЇ РОБОТИ

2.1.Правила оформлення пояснювальної записки

Пояснювальна записка повинна бути оформлена у відповідності до вимог ДСТУ 3008:2015 [1], ”Єдиної системи програмної документації (ЄСПД)” та Методичних вказівок до оформлення курсових проектів (робіт) у Вінницькому національному технічному університеті. Текст курсової роботи повинен бути надрукований у редакторі Microsoft Word на стандартних аркушах формату А4. Обсяг пояснювальної записки не повинен перевищувати 25-30 сторінок. Шрифт – Times New Roman, розмір 14. Міжрядковий інтервал – 1,5. На сторінках пояснювальноїзаписки потрібно використовувати береги такої ширини: верхній і нижній – не менше 20 мм, лівий – не менше 25 мм, правий – не менше 10 мм. Абзац – 5 знаків. Нумерація сторінок в правому верхньому кутку. На титульній сторінці номер не ставиться, проте, він входить до нумерації. Заголовки розділів друкувати посередині рядка великими літерами напівжирним шрифтом без крапки в кінці. Розділи "**ВСТУП**", "**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ**", "**ВИСНОВКИ**" не нумерується. Заголовки підрозділів, пунктів і підпунктів потрібно друкувати з абзацного відступу з великої літери без крапки в кінці. Перелік джерел посилань оформлюється відповідно до ДСТУ ГОСТ 7.1.2006 [2]. Кожен розділ рекомендується починати з нової сторінки. Розділи нумерують порядковими номерами в межах всього документа (І, 2, і т.д.). Після номера крапку не ставлять. Підрозділи нумерують в межах кожного розділу, пункти в межах підрозділу і т.д. за формою (3.1, 3.2, 3.2.1, 3.2.2, 3.2.2.1 і т.д.). Цифри, які вказують номер, не повинні виступати за абзац. Посилання в тексті на розділи виконується за формою: “...наведено в розділі 3”. Текст кожного додатка починають з нової сторінки. Посередині першого рядка додатка друкують слово "**ДОДАТОК**" і відповідну велику літеру української абетки, крім літер Ґ, Є, З, І, І, Й, О, Ч, Ь, яка позначає додаток. Друга стрічка повинна містити тематичний заголовок додатку, який друкують посередині рядка малими літерами з першої великої.

2.2.Структура пояснювальної записки

Курсова робота повинна мати таку структуру.

1. Титульна сторінка.

Оформлюється за зразком, наведеним у додатку А. Титульна сторінка не нумерується, але враховується при нумерації.

2. Індивідуальне завдання до курсової роботи.

Оформлюється за зразком, наведеним у додатку Б. Підписується студентом,

керівником та завідувачем кафедри. Індивідуальне завдання не нумерується, але враховується при нумерації. Даний розділ не нумерується.

3. Анотація.

Містить 5-7 речень, у яких наводиться короткий опис пояснювальної записки. Анотація не нумерується, але враховується при нумерації.

4. Зміст.

З даної сторінки починається нумерація, але сама сторінка у змісті не вказується. У змісті вказуються усі розділи і підрозділи, а також додатки у такому ж форматі, в якому вони подані у тексті пояснювальної записки. Заголовок даного розділу не нумерується.

5. Вступ.

Вступ висвітлює: стан розвитку проблеми в даній галузі, до якої має відношення розробка; галузь використання та призначення; мету та загальну постановку задачі; актуальність, яка повинна подаватись в останньому абзаці вступу. Кількість сторінок вступу не повинна перевищувати, 1 - 2 сторінок.

6. Аналіз сучасного стану технологій програмування та обгрунтування теми.

Цей розділ є обов’язковим та передбачає посилання до відомих літературних джерел, враховуючи тенденції розвитку та сучасний стан програмування. У цьому розділі необхідно порівняти основні відомі технології програмування та мови, що їх застосовують.

7. Розробка програми виконання завдання.

7.1. Розробка методу виконання завдання.

Описуються і пояснюються дії, необхідні для виконання завдання. При необхідності подаються формули та рисунки. Може бути наведено узагальнений алгоритм роботи програми.

7.2. Структура даних і функцій.

Описуються: окремі змінні та функції. В описі змінних потрібно вказувати їх тип, модифікацію, призначення, область видимості та час існування.

8. Розробка та виконання тестового прикладу.

Сюди входять: опис форматів даних, роздруковані тексти вихідних даних та одержаних результатів, роздруковані графічні результати виконання програми.

9. Інструкція користувача.

Містить: вимоги щодо апаратної частини та програмного забезпечення комп’ютера, на якому планується використовувати програмний продукт (процесор, об’єм пам’яті, відеокарта, тип операційної системи тощо); рекомендації щодо інсталяції та запуску програмного продукту; інструкція для роботи з програмою.

10. Висновки.

Подається коротка узагальнена характеристика одержаних результатів, їх

відповідності індивідуальному завданню на курсову роботу.

11. Перелік джерел посилань.

Заголовок розділу з посиланнями на літературні джерела літературного джерела повинен мати назву "**ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ**". У даному розділі наводяться лише ті літературні джерела, на які є посилання у тексті пояснювальної записки. Літературні джерела нумеруються у порядку посилань.

12. Додатки.

Першим обов’язковим додатком повинен бути роздрукований текст програми, що виконує індивідуальне завдання. Можуть бути також довідкові додатки.

3. **ВАРІАНТИ ЗАВДАНЬ ДО КУРСОВОЇ РОБОТИ**

1. У файлі задано координати вершин прямокутників. Відсортувати у файлі координати по зменшенню площі прямокутників і вивести на екран найменший та найбільший за периметром прямокутники. Вивести їх діагоналі. Окремим кольором вивести діагональ, що має найменший кут нахилу до горизонталі.

2. У двох файлах задано координати протилежних вершин прямокутників, сторони яких розташовані вертикально або горизонтально. Координати вершин тих прямокутників, периметр яких більше середнього значення периметрів, перемістити в інший файл. Вивести на екран прямокутник, периметр якого найближчий до суми двох найменших прямокутників.

3. У файлі задано координати кінців відрізків прямої. Вивести на екран відрізки, що знаходяться у верхній половині екрану. Окремим кольором виділити ті з них, що знаходяться у лівій половині екрану. Другим кольором виділити ті з них, що знаходяться у правій половині екрану та мають найменшу і найбільшу довжину.

4. У файлі задано координати вершин трикутників. Вивести на екран трикутники. Залишити у файлі тільки координати тупокутних трикутників. Трикутник з найбільшим та найменшим тупим кутом виділити окремим кольором. Другим кольором виділити рівнобедрені трикутники.

5. У першому файлі задано координати кінців відрізку прямої. У другому - координати різних точок. Утворити третій файл. Розділити координати таким чином, щоб координати точок, що лежать по одну сторону від прямої, знаходились в другому файлі, а інші - в третьому. Вивести на екран відрізок та прямокутник, діагоналлю якого є даний відрізок. Окремим кольором вивести точки з другого файлу, що потрапляють в цей прямокутник. Навколо найближчої до центру прямокутника точки намалювати коло з радіусом, що дорівнює найменшій відстані між точками.

6. У файлі задано координати точок, кількість яких більша десяти. Вивести на екран десять точок, у яких відстань від центра екрану найменша. Провести через ці точки ламану лінію. Окремим кольором виділити найкоротший та найдовший відрізки ламаної.

7. У файлі задано координати центра та радіуса кола, а також точок, що лежать на колі. Вивести на екран вписані трикутники з найбільшою та найменшою площею. Якщо в коло вписаний квадрат, то вивести його

окремим кольором.

8. У одному файлі задано координати кінців відрізку прямої. У іншому - величини відхилень по осі У від прямої десяти рівномірно віддалених по осі Х точок, на які поділено відрізок. Вивести на екран одним кольором відрізок прямої, другим - ламану лінію, що проходить через десять точок; третім - точки, відстань між якими менша від середньої довжини відрізка ламаної.

9. У файлі задано центри та радіуси кіл. Вивести їх на екран. Кола, що лежать всередині інших кіл виділити окремим кольором. Використовуючи центри виділених кіл як вершини, побудувати найбільший за площею трикутник.

10. У файлі задано координати вершин трикутників. Вивести на екран ці трикутники. Окремим кольором виділити рівнобедрені трикутники. Навколо двох найбільших трикутників описати прямокутник з горизонтальними та вертикальними сторонами. Третім кольором виділити ті трикутники, що знаходяться в середині прямокутника.

11. У файлі задано координати та радіуси кіл. Вивести їх на екран. Кола, що перетинаються з іншими колами, виділити окремим кольором. Другим кольором виділити ті кола, площа яких менша середньої площі кіл, виділених першим кольором.

12. У файлі задано координати двох точок прямої та вершин трикутників. Вивести на екран ці трикутники і пряму. Одним кольором виділити трикутники, що лежать по одну сторону від прямої, другим – трикутники, що лежать по іншу сторону, а третім – ті, що перетинаються з прямою.

13. У файлі задано координати та радіуси кіл. Вивести їх на екран. Окремим кольором виділити кола, що перетинаються з іншими колами. Другим кольором виділити кола, що вкладені в інші кола. Третім кольором виділити найбільше та найменше коло.

14. У файлі задано координати вершин прямокутників з горизонтальними та вертикальними сторонами. Вивести їх на екран. Прямокутники, що вкладені в інші виділити окремим кольором. Другим кольором виділити вкладені прямокутники з максимальною і мінімальною площами.

15. У файлі задано координати кінців відрізків. Вивести їх на екран. З’єднати відрізками окремого кольору точки, відстань між котрими не перевищує половини суми найбільшого та найменшого відрізків. Другим

кольором виділити ті з виділених відрізків, що утворюють найбільший за периметром трикутник. Якщо такого трикутника немає, то вивести повідомлення.

16. У файлі задано координати точок. Вивести їх на екран. Використовуючи точки як вершини, побудувати найбільший та найменший за площею тупокутні трикутники. Навколо вершин цих трикутників побудувати кола радіусом, що дорівнює координаті X найвищої точки.

17. У файлі задано координати вершин прямокутника з горизонтальними та вертикальними сторонами. В іншому файлі задано координати вершин трикутників. Вивести на екран прямокутник та трикутники, що цілком лежать всередині прямокутника. Окремим кольором виділити найбільший та найменший за площею трикутники.

18. У файлі задано координати вершин багатокутника. Вивести на екран цей багатокутник. Навколо кожної вершини, відстань від якої до хоча б однієї з суміжних не перевищує величини, заданої в іншому файлі, провести коло радіусом, що дорівнює половині суми найбільшої та найменшої діагоналей.

19. У файлі задано координати вершин трикутників. В іншому файлі задано координати вершин прямокутників. Вивести на екран трикутники та прямокутники. Окремим кольором виділити прямокутники, що мають площу меншу від середньої площі трикутників.

20. У одному файлі задано координати центру та радіуси кіл. В іншому файлі задано координати вершин прямокутників. Вивести на екран кола та прямокутники. Кола, що перетинаються з прямокутниками, виділити окремим кольором. Другим кольором виділити найменше та найбільше кола.

21. У файлі задано координати вершин багатокутника. Вивести на екран цей багатокутник та всі його діагоналі. Навколо кожної вершини намалювати коло радіусом, що дорівнює третині суми найкоротшої діагоналі та довжин сторін, прилеглих до цієї вершини. Кола з найбільшим та найменшим радіусами виділити окремим кольором.

22. У файлі задано координати вершин чотирикутників. Вивести на екран ці чотирикутники і їх діагоналі. Окремим кольором вивести чотирикутники, у яких відстань між лівими верхніми вершинами мінімальна та максимальна. Використовуючи ці відстані як радіуси, побудувати чотири кола. Кола, що перетинаються, виділити окремим кольором.

23. У файлі задано координати вершин двох опуклих багатокутників, один з яких повністю лежить в середині іншого. Вивести на екран ці багатокутники різними кольорами. Сторони зовнішнього багатокутника, що мають довжину, більшу ніж найбільша діагональ внутрішнього багатокутника, виділити окремим кольором.

24. У одному файлі задано координати прямої. У другому файлі у випадковому порядку задано координати вершин ламаної. Вивести на екран пряму та ламану. Окремим кольором виділити відрізки ламаної, що перетинаються з прямою.

25. У файлі задано координати вершин трикутників та чотирикутників. Вивести їх на екран. Окремим кольором виділити ті тупокутні трикутники, периметр яких менший ніж половина периметру найбільшого чотирикутника. Другим кольором виділити трикутник, що відповідає умові та має найбільший периметр.

26. У файлі задано координати центрів та радіуси кіл, а також координати кінців прямої. Вивести на екран відрізок і кола, що він перетинає. Центри двох найближчих кіл з'єднати прямою. Через решту центрів провести ламану, що не перетинається.

27. У файлі задано координати вершин трикутників. Вивести на екран трикутники, периметр яких знаходиться в межах, заданих в іншому файлі. Серед виведених трикутників окремим кольором виділити той, що має площу, найближчу до величини, заданої в іншому файлі, та не перетинається з іншими трикутниками.

28. У двох окремих файлах задано координати точок двох ламаних. Вивести на екран ламані. Окремим кольором виділити відрізки першої ламаної, що перетинають другу ламану.

29. У одному файлі знаходяться координати крайніх точок десяти відрізків прямих. У другому задано параметри прямої. Виконати паралельне перенесення відрізків таким чином, щоб одні їх кінці лежали на прямій. Вивести на екран пряму та відрізки. Окремим кольором вивести перенесені відрізки.

30. У файлі задано координати центрів та радіуси кіл. Вивести на екран кола та вписані в них квадрати. Кола, у яких вписані квадрати мають площу, що лежить в межах між середньою площею кіл та середньою площею квадратів, виділити окремим кольором.

31. У файлі задано координати центрів та радіуси кіл. В іншому файлі задано координати кінців відрізка прямої. Вивести на екран ці кола і квадрат, діагоналлю якого є відрізок. Окремим кольором виділити кола, що перетинаються з іншими колами та лежать всередині квадрата.

32. У файлі задано координати точок. Вивести ці точки на екран. З'єднати між собою точки, що є вершинами ромбів. Ромб з найбільшою площею виділити окремим кольором. Другим кольором виділити квадрати.

33. У файлі задано координати точок. Вивести ці точки на екран. З'єднати між собою точки, що є вершинами паралелограмів. Паралелограм з найбільшим периметром виділити окремим кольором. Другим кольором виділити прямокутники.

34. У файлі задано координати вершин опуклого багатокутника. Вивести на екран цей багатокутник та найбільший за площею вписаний в нього трикутник.

35. У файлі задано координати прямокутників, сторони яких розташовані горизонтально або вертикально. Вивести на екран чотирикутники. Окремим кольором вивести чотирикутники, що перетинаються з іншими. Третім кольором виділити чотирикутники, вкладені в інші. Четвертим кольором виділити найбільший з вкладених чотирикутників та найменший з чотирикутників, що перетинаються.

36. У файлі задано координати точок. Провести через них ламані лінії, що мають найбільшу та найменшу довжину. Вивести ламані на екран. Найдовший та найкоротший відрізок ламаних вивести окремим кольором.

Завдання можуть змінюватись та доповнюватись викладачем.

4. ДОВІДКОВА ІНФОРМАЦІЯ

Деякі стандартні функції мови С++

Таблиця 3.1 – Стандартні функції для роботи із символами (ctype.h)

Функція	Опис	Тип повернення
isalnum(int c);	Перевіряє, чи є символ літерою або цифрою.	int
isalpha(int c);	Перевіряє, чи є символ літерою.	int
iscntrl(int c);	Перевіряє, чи є символ керуючим.	int
isdigit(int c);	Перевіряє, чи є символ десятковою цифрою.	int
isgraph(int c);	Перевіряє, чи є символ видимим.	int
islower(int c);	Перевіряє, чи є символ літерою нижнього регістра.	int
ispunct(int c);	Перевіряє, чи є символ знаком пунктуації.	int
isspace(int c);	Перевіряє, чи є символ пробільним.	int
isupper(int c);	Перевіряє, чи є символ літерою верхнього регістру.	int
isxdigit(int c);	Перевіряє, чи є символ шістнадцятковою цифрою.	int
tolower(int c);	Перетворює символ у символ нижнього регістра.	int
toupper(int c);	Перетворює символ у символ верхнього регістра.	int

Таблиця 3.2 – Стандартні функції для роботи з каталогами (dir.h)

Функція	Опис	Тип повернення
chdir(char *pathname);	Змінює поточний робочий каталог.	int
findfirst(char *pathname, struct ffblk *buf, int attr);	Шукає перше розташування файла або каталогу.	int
fnmerge(char *path, char *drive, char *dir, char *name,char *ext)	Складає ім’я файла з окремих частин.	void
fnsplit(char *path, char *drive, char *dir, char *name,char *ext);	Розкладає ім’я файла на окремі компоненти.	int
getcurdir(int drive,char *directory);	Повертає поточний каталог на вказаному диску.	int
getcwd(char *buf, int n);	Повертає повне ім’я поточного каталогу.	char *
getdisk(void);	Повертає поточний диск.	int
mkdir(char *pathname);	Створює новий каталог.	int
mktemp(char *template);	Генерує унікальне ім’я файла.	char *
rmdir(const char *path);	Знищує каталог.	int
searchpath(char *filename);	Продовжує пошук файла, початого функцією findfirst.	char *
setdisk(int drive);	Встановлює поточний диск.	int

Таблиця 3.3 – Стандартні функції для роботи з ОС (dos.h)

Функція	Опис	Тип повернення
absread(int drive, int nsect, int sectno, void *buffer);	Читає інформацію із сектора.	int
abswrite(int drive, int nsect, int sectno, void *buffer);	Записує інформацію у сектор.	int
bdos(int dosfun, unsigned dosdx, unsigned dosal);	Викликає MS-DOS.	int
ctrlbrk(int (*handler)(void));	Встановлює реакцію на CTRL-Break.	void
delay(unsigned milliseconds);	Призупиняє роботу програми на вказане число мілісекунд.	void
getcbkr(void);	Повертає поточну встановлену реакцію на CTRL-Break.	int
getdate(struct date *datep);	Повертає поточну дату.	void
getdfree(int drive, struct dfree *dtable);	Повертає об’єм вільного місця на диску.	void
getfat(int drive, struct fatinfo *fatblkp);	Отримує інформацію FAT.	void
getfatd(struct fatinfo *dtable);	Отримує інформацію FAT про поточний диск.	void
getftime(int handle,struct ftime, *ftimep);	Повертає дату і час створення файла.	int
gettime(struct time *timep);	Повертає поточний системний час.	void
inp(unsigned portid);	Читає один байт з вхідного порту port.	int
inport(int portid);	Читає слово(два байти) з вхідного порту.	int
inportb(int portid);	Читає байт з порту введення.	unsigned char
int86x(int intno, union REGS *inregs, union REGS *outregs, struct SREGS *segregs);	Виконує системне переривання.	int
intr(int intno, struct REGPACK *preg);	Виконує системне переривання.	void
keep(unsigned char status, unsigned size);	Завершує роботу і залишає програму резидентною.	void
nosound(void);	Відключає звук.	void
outp(unsigned portid, int value);	Записує байт у порт.	int
peek(unsigned segment, unsigned offset); peekb(unsigned segment, unsigned offset);	Отримує значення байта або слова за адресою.	int char
poke(unsigned segment, unsigned offset, int value); pokeb(unsigned segment, unsigned offset, char value)	Записує значення байта або слова за адресою.	void void
settime(struct time *timep);	Встановлює поточний час.	void
sleep(unsigned seconds);	Призупиняє виконання програми на задану кількість секунд.	void
sound(unsigned frequency);	Генерує звуковий сигнал із заданою частотою.	void

Таблиця 3.4 – Стандартні функції для виведення у графічному режимі (graphics.h)

Функція	Опис	Тип повернення
1	2	3
bar(int left, int top, int right, int bottom);	Рисує зафарбований прямокутник.	void far
arc(int x, int y, int stangle, int endangle, int radius);	Рисує дугу кола.	void far
bar3d(int left, int top, int right, int bottom, int depth, int topflag);	Рисує тривимірний стовпець.	void far
circle(int x, int y, int radius);	Рисує коло із заданими координатами центра та радіусом.	void far
cleardevice(void);	Очищає екран.	void far
clearviewport(void);	Очищає графічне вікно.	void far
closegraph(void);	Закриває графічний режим.	void far
detectgraph(int far *graphdriver, int far *graphmode);	Повертає тип графічного драйвера.	void far
drawpoly(int numpoints, int far *polypoints);	Рисує ламану лінію.	void far
ellipse(int x, int y, int stangle, int endangle, int xradius, int yradius);	Рисує еліптичну дугу від початкового кута до кінцевого.	void far
fillellipse(int x, int y, int xradius, int yradius);	Рисує заштрихований еліпс.	void far
fillpoly(int numpoints, int far *polypoints);	Рисує заштрихований багатокутник.	void far
floodfill(int x, int y, int border);	Зафарбовує замкнену область.	void far
getaspectratio(int far *xasp, int far *yasp);	Повертає відношення сторін графічного екрана.	void far
getbkcolor(void);	Повертає поточний колір фону.	int far
getcolor(void);	Повертає поточний колір.	int far
getfillpattern(char far *pattern);	Повертає поточний тип штрихування.	void far
getfillsettings (struct fillsettingstype far *fillinfo);	Повертає поточний тип і колір штрихування.	void far
getimage(int left, int top, int right, int bottom, void far *bitmap);	Зберігає бітовий образ частини екрана.	void far
getlinesettings(struct linesettingstype far *lineinfo);	Повертає поточний стиль, шаблон і товщину штрихування.	void far
getmaxcolor(void);	Повертає максимальний колір, який можна задавати у параметрах.	int far
getmaxx(void); getmaxy(void);	Повертають, відповідно, максимальну X-координату та Y-координату екрана.	int far int far
getpixel(int x, int y);	Повертає колір піксела з координатами (x,y)	unsigned far
gettextsettings(struct textsettingstype far *texttypeinfo);	Повертає поточний шрифт, розмір та вирівнювання тексту.	void far
getx(void); gety(void);	Повертають, відповідно, X- та Y-координати поточного вказівника.	int far int far
graphresult(void);	Повертає код помилки для останньої графічної операції.	int far

Продовження табл. 3.4

1	2	3
imagesize(int left, int top, int right, int bottom);	Повертає число байт, що необхідні для зберігання прямокутної частини екрана.	unsigned far
initgraph(int far *graphdriver, int far *graphmode, char far *pathtodriver);	Ініціалізує графічний режим роботи адаптера.	void far
line(int x1, int y1, int x2, int y2);	Рисує лінію від точки (x1,y1) до точки (x2,y2).	void far
linerel(int dx, int dy);	Рисує лінію від поточного положення вказівника до точки, заданої приростом координат.	void far
lineto(int x, int y);	Рисує лінію від поточного положення вказівника до заданої точки.	void far
moverel(int dx, int dy);	Переміщує вказівник до точки, заданої приростом координат.	void far
moveto(int x, int y);	Переміщує вказівник до точки із заданими координатами.	void far
outtext(char far *textstring);	Виводить текстовий рядок на екран.	void far
outtextxy(int x, int y, char far *textstring);	Виводить текстовий рядок у задане місце екрана.	void far
pieslice(int x, int y, int stangle, int endangle, int radius);	Рисує і заштриховує сектор кола.	void far
putimage(int left, int top, void far *bitmap, int op);	Виводить бітовий образ на екран.	void far
putpixel(int x, int y, int color);	Виводить точку із заданими координатами і кольором.	void far
rectangle(int left, int top, int right, int bottom);	Рисує прямокутник.	void far
sector(int x, int y, int stangle, int endangle, int xradius, int yradius);	Заштриховує сектор еліпса.	void far
setaspectratio(int xasp, int yasp);	Змінює масштабний коефіцієнт відношення сторін екрана.	void far
setbkcolor(int color);	Встановлює колір фону.	void far
setcolor(int color);	Встановлює колір для рисування та тексту.	void far
setfillpattern(char far *upattern, int color);	Встановлює тип штрихування (довільний).	void far
setfillstyle(int pattern, int color);	Встановлює тип і колір штрихування.	void far
setlinestyle(int linestyle, unsigned upattern, int thickness);	Встановлює товщину і стиль лінії.	void far
settextjustify(int horiz, int vert);	Встановлює вирівнювання тектсу.	void far
settextstyle(int font, int direction, int charsize);	Встановлює поточний шрифт, стиль і розмір тексту.	void far
setviewport(int left, int top, int right, int bottom, int clip);	Визначає вікно для графічного виведення.	void far
textheight(char far *textstring);	Повертає висоту рядка у пікселах.	int far
textwidth(char far *textstring);	Повертає довжину рядка у пікселах.	int far

Таблиця 3.5 – Стандартні математичні функції (math.h)

Функція	Опис	Тип повернення
abs(int x);	Повертає модуль цілого числа.	int
acos(double x); acosl(long double (x));	Повертає арккосинус аргументу.	double long double
asin(double x); asinl(long double (x));	Повертає арксинус аргументу.	double long double
atan(double x); atanl(long double (x));	Повертає арктангенс аргументу.	double long double
atan2(double y, double x); atan2l(long double(y), long double (x));	Повертає арктангенс відношення аргументів.	double long double
ceil(double x); ceill(long double (x));	Заокруглює до найменшого цілого, більшого або рівного заданому числу.	double long double
cos(double x); cosl(long double x);	Обчислює косинус.	double long double
cosh(double x); coshl(long double (x));	Обчислює гіперболічний косинус.	double long double
exp(double x); exp(long double (x));	Повертає степінь числа e.	double long double
fabs(double x); fabsl(long double @E(x));	Повертає модуль числа (для дійсних чисел).	double long double
floor(double x); floorl(long double (x));	Заокруглює до найменшого цілого, меншого або рівного заданому числу.	double long double
fmod(double x, double y); fmod(long double (x), long double (y));	Повертає залишок від ділення аргументів.	double long double
frexp(double x, int *exponent); frexp(long double (x), int *(exponent));	Виділяє з числа мантису і експоненціальну частину.	double long double
ldexp(double x, int expon); ldexpl(long double (x), int (expon));	Перетворює мантису і показник степеня у число.	double long double
log(double x); logl(long double (x));	Обчислює натуральний логарифм.	double long double
log10(double x); log10l(long double (x));	Обчислює десятковий логарифм.	double long double
modf(double x, double *ipart); modfl(long double (x), long double *(ipart));	Розбиває число на цілу і дробову частини.	double long double
pow(double x, double y); pow(long double (x), long double (y));	Підносить число до вказаного степеня.	double long double
sinl(long double x) sin(double x);	Обчислює синус аргументу.	long double double
sinh(double x); sinhl(long double (x));	Обчислює гіперболічний синус аргументу.	double long double
sqrt(double x); sqrtl(long double @E(x));	Обчислює квадратний корінь аргументу.	double long double
tan(double x); tanl(long double x);	Обчислює тангенс аргументу.	double long double
tanh(double x); tanhl(long double (x));	Обчислює гіерболічний тангенс аргументу.	double long double

Таблиця 3.6 – Стандартні функції I/O (stdio.h)

Функція	Опис	Тип повернення
1	2	3
clearerr(FILE *stream);	Очищує прапорець помилок для вказаного потоку.	void
fclose(FILE *stream);	Закриває потік.	int
fcloseall(void);	Закриває всі відкриті (на верхньому рівні) файли (потоки).	int
feof(FILE *stream);	Перевіряє на кінець потоку.	int
ferror(FILE *stream);	Перевіряє прапорець помилок потоку.	int
fflush(FILE *stream);	Записує дані з буфера у потік.	int
fgetc(FILE *stream);	Читає символ з потоку.	int
fileno(FILE *stream);	Отримує дескриптор, пов’язаний з потоком.	int
fgetchar(void);	Читає символ із стандартного потоку введення.	int
fgetpos(FILE *stream, fpos_t *pos);	Повертає поточну позицію у файлі.	int
fgets(char *s, int n, FILE *stream);	Читає рядок з потоку.	char *
fdopen(int handle, char *type);	Відкриває потік (відкриває файл і зв’язує його з потоком).	FILE*
fprintf (FILE *stream, const char *format [, argument, ...]);	Записує форматовані дані у файл (потік).	int
fputc(int c, FILE *stream);	Записує символ у файл (потік).	int
fputchar(int c);	Записує символ у стандартний потік.	int
fputs(const char *s, FILE *stream);	Записує рядок у файл (потік).	int
fread(void *ptr, size_t size, size_t n, FILE *stream);	Читає дані з файла (потоку).	size_t
freopen(const char *filename, const char *mode, FILE *stream);	Повторно відкриває файл або потік у новому режимі.	FILE *
fscanf (FILE *stream, const char *format [, address, ...]);	Читає форматовані дані з файла (потоку).	int
fseek(FILE *stream, long offset, int whence);	Змінює позицію вказівника файла.	int
fsetpos(FILE *stream, const fpos_t *pos);	Переміщує вказівник файла відносно початку файла.	int
ftell(FILE *stream);	Повертає поточну позицію вказівника файла.	long
fwrite(const void *ptr, size_t size, size_t n, FILE*stream);	Записує дані із заданого буфера у файл (потік).	size_t
getc(FILE *stream);	Читає символ з файла (потоку).	int
getchar(void);	Читає символ з потоку stdin.	int
gets(char *s);	Читає рядок з потоку stdin.	char*
getw(FILE *stream);	Читає слово (два байти) з потоку.	int
printf (const char *format [, argument, ...]);	Записує форматовані дані у потік stdout.	int
putc(int c, FILE *stream);	Записує символ у файл (потік).	int
putchar(int c);	Записує символ у потік stdout.	int
puts(const char *s);	Записує рядок у потік stdout.	int

Продовження табл. 3.6

1	2	3
putw(int w, FILE *stream);	Записує слово (два байти) у файл (потік).	int
remove(const char *filename);	Знищує файл.	int
rename(const char *oldname, const char *newname);	Перейменовує файл.	int
rewind(FILE *stream);	Встановлює вказівник файла на його початок	void
scanf (const char *format [, address, ...]);	Читає форматовані дані з потоку stdin.	int
setbuf(FILE *stream, char *buf);	Встановлює буферизацію файла (потоку).	void
setvbuf(FILE *stream, char *buf, int type, size_t size);	Встановлює буферизацію і розмір файла (потоку).	int
sprintf (char *buffer, const char *format [, argument, ...]);	Записує форматовані дані у рядок.	int
sscanf (const char *buffer, const char *format [, address, ...]);	Читає форматовані дані з рядка.	int
tempnam(char *dir, char *prefix);	Генерує ім’я тимчасового файла у заданому каталозі.	char *
ungetc(int c, FILE *stream);	Повертає символ у файл (потік).	int
vfscanf(FILE *stream, const char *format, va_list arglist);	Читає форматовані дані з файла (потоку) з використанням списку аргументів.	int
vprintf (const char *format, va_list arglist);	Записує форматовані дані у стандартний потік виведення з використанням списку аргументів.	int
vsprintf(char *buffer, const char *format, va_list arglist);	Виводить форматовані дані у рядок з використанням списку аргументів.	int
vsscanf(const char *buffer, const char *format, va_list arglist);	Читає форматовані дані з рядка з використанням списку аргументів.	int

Таблиця 3.7 – Стандартні функції для роботи з рядками (string.h)

Функція	Опис	Тип повернення
1	2	3
strcat(char *dest, const char *src);	Об’єднує рядки.	char *
strchr(const char *s, int c);	Шукає символ у рядку.	char *
strcmp(const char *s1, const char*s2);	Порівнює рядки.	int
strcpy(char *dest, const char *src);	Копіює один рядок в інший.	char *
strcspn(const char *s1, const char *s2);	Знаходить перше входження символа із заданого набору символів у рядку.	size_t
strdup(const char *s);	Дублює рядок.	char *
strerror(int errnum);	Повертає вказівник на рядок з описом помилки.	char *
strlen(const char *s);	Повертає довжину рядка.	size_t
strlwr(char *s);	Перетворює рядок у нижній регістр.	char *
strncat(char *dest, const char *src, size_t maxlen);	Об’єднує один рядок з n символами іншого.	char *

Продовження табл. 3.7

1	2	3
strncmp (const char *s1, const char *s2, size_t maxlen);	Порівнює один рядок з n символами іншого.	int
strncpy(char *dest, const char *src, size_t maxlen);	Копіює перші n символів одного рядка в інший.	char *
strnset(char *s, int ch, size_t n);	Заповнює n символів рядка заданим значенням.	char *
strpbrk(const char *s1, const char *s2);	Знаходить перше входження будь-якого символа із заданого набору у рядок.	char *
strrchr(const char *s, int c);	Шукає перше входження заданого символа у рядок.	char *
strrev(char *s);	Інвертує рядок.	char *
strncat(char *dest, const char *src, size_t maxlen);	Встановлює всі символи рядка у задане значення.	char *
strspn(const char *s1, const char *s2);	Шукає перший символ одного рядка, відсутній в іншому.	size_t
strstr(const char *s1, const char *s2);	Шукає частину рядка в іншому рядку.	char *
strupr(char *s);	Перетворює рядок у верхній регістр.	char *

Таблиця 3.8 – Консольні стандартні функції I/O (conio.h)

Функція	Опис	Тип повернення
1	2	3
cgets(char *str);	Читає рядок з консолі.	char *
clreol(void);	Стирає частину рядка від поточного положення курсора до правої границі вікна.	void
clrscr(void);	Очищає екран або вікно.	void
cprintf (const char *format [arg, ...]); .	Виводить форматовані дані у текстове вікно.	int
cputs(const char *str);	Виводить рядок у текстове вікно	int
cscanf (char *format [, address, ...]);	Читає форматовані дані з консолі.	int
delline(void);	Знищує поточний рядок у текстовому вікні.	void
getch(void);	Читає символ з консолі без ехо-друку.	int
getche(void);	Читає символ з консолі з ехо-друком.	int
getpass(const char *prompt);	Читає 8 символів з консольного терміналу без ехо-друку	char *
gettext(int left, int top, int right, int bottom, void*destin);	Копіює частину текста з екрана у заданий буфер	int
gettextinfo(struct text_info *r);	Дає інформацію про текстовий режим.	void
gotoxy(int x, int y);	Переводить курсор у вказане місце екрана або текстового вікна.	void
highvideo(void);	Встановлює високу яскравість символів.	void
insline(void);	Вставляє порожній рядок у текстове вікно.	void
inp(unsigned portid);	Читає байт з порту.	int
inport(int portid);	Читає 2 байти з порту.	int

Продовження табл. 3.8

1	2	3
inportb(int portid);	Читає байт з порту.	unsigned char
inpw(unsigned portid);	Читає з порту 2 байти.	unsigned
kbhit(void);	Перевіряє натискання кнопки.	int
lowvideo(void);	Встановлює низьку яскравість символів.	void
movetext(int left, int top, int right, int bottom, int destleft, int desttop);	Копіює текст з однієї прямокутної частини екрана в іншу.	int
normvideo(void);	Встановлює стандартну яскравість символів.	void
outp(unsigned portid, int value);	Записує байт у порт.	int
outport(int portid, int value);	Записує 2 байти у порт.	void
outportb(int portid, unsigned char value);	Записує байт у порт.	void
outpw(unsigned portid, unsigned value);	Записує 2 байти у порт.	unsigned
putch(int ch);	Виводить символ у текстове вікно.	int
puttext(int left, int top, int right, int bottom, void*source);	Виводить текст з вказаного буфера на екран.	int
_setcursortype(int cur_t);	Встановлює тип курсора.	void
textattr(int newattr);	Встановлює колір тексту і фону.	void
textbackground(int newcolor);	Встановлює колір фону.	void
textcolor(int newcolor);	Встановлює колір тексту.	void
textmode(int newmode);	Встановлює текстовий режим.	void
ungetch(int ch);	Повертає символ, введений з клавіатури.	int
wherex(void); wherey(void);	Повертають відповідно поточну X- або Y-координати	int int
window(int left, int top, int right, int bottom);	Встановлює координати поточного текстового вікна.	void

5. РЕКОМЕНДОВАНА ЛІТЕРАТУРА

1. ДСТУ 3008:2015 "Інформація та документація. Звіти у сфері науки і техніки. Структура і правила оформлення".
2. ДСТУ ГОСТ 7.1.2006 "Система стандартів з інформації, бібліотечної та видавничої справи. Бібліографічний запис, бібліографічний опис. Загальні вимоги та правила складання".
3. Методичні вказівки до оформлення курсових проектів (робіт) у Вінницькому національному технічному університеті /Уклад. Г.Л. Лисенко, А.Г. Буда, Р.Р. Обертюх, – Вінниця: ВНТУ, 2006. – 60 с.
4. Страуструп Б. Программирование: принципы, практика и использование C++ . : Пер. с англ. – М. : ООО "И. Д. Вильямс", 2011. 1248 с.
5. Прокофьев В. П. Сухарев Н. Н., Храмов Ю. Е. Графические средства Turbo C и Turbo C++. - М.: Финансы и статистика, 1992.
6. Златопольский Д. М. Сборник задач по программированию. – 2-е изд., перераб. и доп. – СПб. : БХВ-Петербург, 2007. – 240 с.

ДОДАТОК А

Зразок титульної сторінки

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій і комп’ютерної інженерії
Кафедра обчислювальної техніки

КУРСОВА РОБОТА

з дисципліни "Програмування"
на тему "Розробка програм мовою С++ "

Студента(ки) _____ курсу _____ групи
напряму підготовки _____
спеціальності _____

_____
(прізвище, ініціали)

Керівник _____

_____
(посада, вчене звання, прізвище, ініціали)

Національна шкала _____
Кількість балів_____ Оцінка: ECTS_____

Члени комісії _____ _____
(підпис) (прізвище, ініціали)

_____ _____
(підпис) (прізвище, ініціали)

_____ _____
(підпис) (прізвище, ініціали)

м. Вінниця – 201_ рік

ДОДАТОК Б

Зразок індивідуального завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп’ютерної інженерії

Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ
Зав. кафедри ОТ, проф., д.т.н.
_____Т. Б. Мартинюк
(підпис)
"___"_____201_ р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу з дисципліни "Програмування"

студенту _____

факультету_____групи_____

ТЕМА: Розробка програм мовою С++

1. Завдання №17

У файлі задано координати кінців відрізків. Вивести їх на екран. Між тими кінцями відрізків, відстань між якими не перевищує половини суми довжин найбільшого та найменшого відрізків, що перетинаються, провести прямі лінії. Окремим кольором виділити проведені відрізки, що утворюють трикутники.

2. Постановка задачі

2.1. Розробити метод розв’язку завдання.
2.2. Розробити програму мовою С для виконання завдання.
2.3. Програма повинна зчитувати дані з текстового файлу, динамічно виділяти для них оперативну пам’ять, за допомогою технології функціонального і структурного програмування реалізувати розроблений метод виконання завдання та виводити результати у графічному режимі на екран.

3. Вихідні дані

3.1. Координати відрізків задати у текстовому файлі за допомогою текстового редактора. Формат запису координат повинен бути зручним для редагування, Наприклад:

Vidrizok1: x1=22; y1=130; x2=340; y2=14;
Vidrizok2: x1=45; y1=39; x2=17; y2=200;

...

3.2. Кількість відрізків наперед невідома.

Дата видачі "____" _____201_ р.

Керівник:
к.т.н., доц. кафедри ОТ Черняк О.І._____
(підпис)

Завдання отримав_____ _____
(підпис) *(ПІБ)*