МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

Кафедра комп’ютерних систем та мереж

МЕТОДИЧНІ ВКАЗІВКИ

до виконання курсової роботи

з дисципліни

«СИСТЕМНЕ ПРОГРАМУВАННЯ»

для здобувачів першого (бакалаврського) рівня вищої освіти
спеціальності 123 «Комп’ютерна інженерія» усіх форм навчання

Тернопіль 2024

Методичні вказівки до виконання курсової роботи з дисципліни «Системне програмування» для здобувачів першого (бакалаврського) рівня вищої освіти спеціальності 123 «Комп'ютерна інженерія» усіх форм навчання / Укладачі: А.М. Паламар., М.І. Паламар. Тернопіль: ТНТУ, 2024. 40 с.

Укладачі: к.т.н., доц. Паламар А.М., д.т.н., проф. Паламар М.І.

Рецензент: доцент каф. КБ, к.т.н., доц. Стадник М.А.

Затверджено на засіданні кафедри комп'ютерних систем та мереж Тернопільського національного технічного університету імені Івана Пулюя.

Протокол №2 від 27 серпня 2024 р.

Схвалено та рекомендовано до друку науково-методичною комісією факультету комп'ютерно-інформаційних систем і програмної інженерії Тернопільського національного технічного університету імені Івана Пулюя.

Протокол №1 від 02 вересня 2024 р.

Методичні вказівки складені з урахуванням методичних розробок інших закладів вищої освіти, а також матеріалів літературних джерел, наведених у переліку.

ЗМІСТ

ВСТУП

Курсова робота – це перша самостійна праця майбутнього інженера, що моделює реальні задачі, які постають у його практичній діяльності. При виконанні курсової роботи студент поглиблює знання з фундаментальних дисциплін, засвоює методикупроектування та розробки програмних продуктів, проведення експериментальних досліджень та тестувань, оволодіває вмінням вести науковий пошук, навичками співставлення результатів своїх досліджень із літературними даними, їх аналізу, узагальнення і документування, розвиває в собі творчий підхід до роботи.

У курсовій роботі з студент повинен продемонструвати уміння використовувати набуті теоретичні знання для вирішення конкретних прикладних задач. Основна мета виконання курсової роботи з системного програмування — закріплення на практиці основних принципів розроблення системного програмного забезпечення та уміння використовувати наявні засоби для його розробки, тестування та супроводу. Студент повинен зуміти оцінити параметри, що випливають із поставленого завдання, вибрати мову програмування, середовище та засоби для розробки, сформулювати структуру майбутнього програмного продукту та реалізувати його у програмному коді. При цьому важливо звернути увагу у своїй роботі на всі етапи життєвого циклу проектованої програми, а не лише на її розробку, оскільки специфіка системного програмного забезпечення передбачає не лише його надійну і безвідмовну роботу під час всього часу експлуатації, але і простоту його вилучення чи заміни на новіші версії.

Важливим є дотримання існуючих стандартів та наявного програмного забезпечення, використання готових модулів, бібліотек, шаблонів, що не тільки скорочує час і вартість розробки, але і підвищує її якість та зменшує кількість помилок.

Результатом виконання курсової роботи є програмний продукт (програма, бібліотека, модуль, драйвер, пакет програм тощо), представлений як у вигляді

вихідного програмного коду, так і скомпільованих виконуваних файлів чи бібліотек. Програмний продукт повинен бути належним чином задокументований у вигляді пояснювальної записки та додаткових креслень, плакатів, схем, оформлених відповідно до діючих національних чи міжнародних стандартів.

1 ПОРЯДОК ВИКОНАННЯ РОБОТИ

Курсова робота — це складова частина навчального процесу. Її написання є обов'язковим елементом підготовки бакалавра за спеціальністю 123 “Комп'ютерна інженерія”.

Теми курсових робіт визначаються викладачем за особистими письмовими заявами студентів з урахуванням їх бажання, нахилів і практичного досвіду та затверджуються на кафедрі у встановлені терміни. Як правило, завданням курсової роботи є розробка нового або удосконалення існуючого програмного продукту, що є частиною операційної системи або відноситься до допоміжних системних програмних засобів, утиліт чи сервісних програм, трансляторів, компіляторів. Можлива також розробка програмних засобів, що забезпечують підтримку спеціалізованого апаратного пристрою або пов’язані з науково-дослідною тематикою кафедри. Під програмним продуктом, що розробляється у даній курсовій роботі, розуміється програма, бібліотека, модуль, драйвер, пакет програм, що розв’язує одну або декілька конкретних задач, обслуговує пристрій, реалізує функцію операційної системи чи певний вид сервісу.

Курсова робота виконується впродовж одного семестру та захищається під час залікової сесії. Витрати навчального часу студентів на виконання курсової роботи визначаються робочим навчальним планом. Перед виконанням курсової роботи необхідно, виходячи з теми роботи, узгодити з керівником та затвердити завдання на курсову роботу (технічне завдання), оцінити обсяг роботи й передбачити очікувані результати, скласти план роботи, в якому вказати етапи та терміни розробки. Про результати виконання курсової роботи студент обов'язково повинен періодично звітувати перед керівником; у випадку порушення даної вимоги та затвердженого плану, студент не допускається до захисту.

Курсова робота за своїм змістом та структурою повинна відповідати етапам життєвого циклу програмного продукту. При використанні різних

підходів до розробки: традиційного (структурного) чи об’єктно-орієнтованого, є деякі відмінності у формулюванні етапів життєвого циклу, проте основні з них залишаються спільними:

– постановка задачі і формулювання вимог до програмного продукту (розробка технічного завдання — ТЗ);

– розробка специфікації (детальний план виконання розробки);

– проектування (розподіл програми на структурні одиниці: модулі, функції; опис для кожної з них детальних вимог, вхідних та вихідних даних);

– реалізація проекту (програмування окремих структурних одиниць);

– інтегрування проекту (об’єднання структурних одиниць у цілісних програмний продукт, тестування, виправлення помилок);

– експлуатація та підтримка (забезпечення користувача необхідною інформаційною та сервісною підтримкою, реалізація внесення змін у програму та доставки виправлених та нових версій програми до користувача);

– утилізація (забезпечення можливості вилучення програмного продукту із системи та переносу на новий комп’ютер чи операційну систему).

Враховуючи специфіку курсової роботи з системного програмування, всі ці етапи життєвого циклу є важливими, тому відповідні питання повинні бути опрацьованими у курсовій роботі. Складність передбачення всіх аспектів роботи програмного продукту, особливо на ранніх етапах розробки, та необхідність внесення значної кількості змін як у саму програму, так і в завдання та документацію привели до того, що документування та тестування проводиться паралельно з проведенням всіх етапів розробки. Оскільки курсова робота виконується лише однією людиною та в обмежених часових рамках, допускається винесення процесу тестування в окремий етап (між інтегруванням та експлуатацією) та оформлення його в окремий розділ чи підрозділ пояснювальної записки.

Важливим також є правильне використання існуючих рішень, підходів та інформаційних ресурсів, тому відбір, вивчення і реферування літературних джерел (публікацій), ресурсів Internet з теми курсової роботи є її необхідною

частиною. При виконанні курсових робіт недостатньо користуватися лише підручниками і навчальними посібниками, оскільки вони здебільшого розкривають лише основи тієї чи іншої мови або технології програмування, а не описують прикладних задач і проблем та методів їх вирішення. Літературні джерела краще вивчати, переходячи від простих до складніших. При вивченні наукової літератури потрібно навчитися чітко відрізняти головне від другорядного, яке безпосереднього не пов'язане з темою дослідження.

Для полегшення розробки та внесення змін у проект, підвищення якості документації бажано забезпечити достатню кількість якісних та змістовних ілюстрацій (оригінальних схем, діаграм тощо), які у порівнянні з текстовим описом є більш лаконічним та наочним способом відображення інформації.

Детальнішу інформацію про вимоги до виконання, змісту та оформлення курсової роботи наведено в розділах 2 та 3. Порядок захисту та критерії оцінювання курсової роботи наведено в розділі 4. Додатки містять зразки оформлення окремих частин пояснювальної записки.

2 ЗМІСТ КУРСОВОЇ РОБОТИ

Рекомендовану структуру курсової роботи наведено в додатку Б.

2.1 Завдання на курсову роботу

Теми курсових робіт затверджують на засіданні кафедри на початку семестру. Основні напрями та види програмних продуктів, що входять в тематику курсових робіт, розглянуті в попередньому розділі. Тема курсової роботи задає лише основні риси та призначення програмного продукту, але не визначає його конкретних параметрів, і тим більше не задає мови програмування та використання тих чи інших засобів розробки. Ці питання розробляються студентом і узгоджуються з керівником роботи і є одним із важливих етапів розробки.

При практичній реалізації програмних проектів першим етапом розробки (першим етапом життєвого циклу програмного продукту) є постановка задачі і формулювання вимог до програмного продукту. Закінчується цей етап формулюванням ТЗ. Вибір студентом завдання на курсову роботу може оформлятися у скороченому вигляді на стандартному бланку або як повноцінний документ — технічне завдання, відповідно до державних стандартів. Завдання повинно бути підписано студентом і керівником розробки ще до початку її втілення, і визначати підстави та основні вимоги до курсової роботи. Повнота і якість виконання вимог, сформульованих у завданні (ТЗ) визначає оцінку курсової роботи.

Виходячи з теми курсової роботи, провівши попередній аналіз, студент формулює та узгоджує з керівником зміст завдання на курсову роботу (ТЗ). У завданні обов’язково має вказуватись: назва університету; шифр і назва спеціальності; назва кафедри; тема курсової роботи; термін здачі студентом завершеної роботи; початкові дані до роботи; зміст роботи (перелік питань, що розробляються) та календарний план виконання роботи. Завдання, підписане

студентом та керівником розміщується на початку пояснювальної записки (після титульного аркуша). Допускається виконувати завдання з обох боків аркуша білого паперу формату А4. Зразок завдання на курсову роботу подано у додатку В.

У пункті 1 “Завдання” слід вказати тему курсової роботи, яка повинна бути сформульована чітко й лаконічно.

У пункті 2 слід вказати термін, до якого студент повинен здати керівникові закінчену роботу на перевірку.

У пункті 3 формулюють мету розробки, прикладну проблему і задачі, які треба вирішити в курсовій роботі, вид розробки (програма, програмний пакет, бібліотека, модуль тощо), задають формат вхідних і вихідних даних програми (інтерфейс програми).

У пункті 4 зазначають перелік питань (розділів і підрозділів пояснювальної записки), які повинні бути розроблені при виконанні курсової роботи.

У пункті 5 вказують конкретні технічні вимоги, параметри програми, формати команд, файлів, основні пункти меню, класи, модулі, функції, операції й т. д. Потрібно вказати апаратні вимоги, тип та версію програмного середовища, у якому має виконуватись програма.

У пункті 6 вказують дату затвердження завдання на курсову роботу. Календарний план повинен містити перелік етапів розробки із зазначеними термінами їх виконання.

2.2 Вступ

У вступі студент обґрунтовує актуальність вибраної теми, коротко викладає призначення розробки, формулює конкретні практичні завдання для вирішення яких вона може бути використана. Тут потрібно сформулювати мету і задачі роботи, визначити основні підходи та ідеї, вибрати спосіб розв’язання задач. При наявності аналогів програмних продуктів, які розв’язують подібні

задачі, потрібно коротко охарактеризувати основні їх обмеження та недоліки й запропонувати шляхи їх усунення.

У вступі слід акцентувати увагу на прикладній проблемі, яку треба вирішити в курсовій роботі, а не на засобах її вирішення. Недоцільно у цьому розділі наводити означення відомих термінів, давати занадто детальні характеристики й описи використаного програмного забезпечення чи апаратних пристроїв та іншу інформацію, що не стосується теми курсової роботи. Якщо для обґрунтування прийнятих рішень необхідно наводити інформацію з літературних джерел, то слід робити це лаконічно, використовуючи порівняльні таблиці, графіки, зазначаючи посилання на джерело інформації, а не просто констатуючи відомі факти.

Якщо при розробці ТЗ чи завдання на курсову роботу були прийняті важливі технічні рішення, проводився детальний аналіз, розрахунки, патентний пошук, інші суттєві дослідження, тоді їх опис приводиться в окремому розділі. В іншому випадку достатньо короткого обґрунтування, наведеного у вступі.

Обсяг вступу — 1-2 сторінки.

2.3 Аналіз завдання на курсову роботу

Основною метою даного розділу (підрозділу) пояснювальної записки є аналіз вимог завдання на курсову роботу й формулювання додаткових технічних вимог, які безпосередньо випливають з нього та мети роботи.

Якщо на попередньому етапі не було здійснено вибір технології, засобів розробки, мови програмування, то на початку аналізу необхідно обґрунтувати їх вибір. Задля забезпечення високої швидкодії, зменшення розмірів виконуваних файлів та обсягів оперативної пам’яті можливі наступні підходи до розробки:

– використання мови програмування низького рівня (Асемблера) чи мови С з можливим підключенням готових модулів;

– використання мови програмування високого (С, С++) або низького

(Асемблер) рівня з підключенням стандартних бібліотек операційної системи (API).

Перший підхід частіше використовується при розробці низькорівневих драйверів спеціалізованих апаратних пристроїв, критичних частин операційних систем, сервісних програм, що працюють в обхід операційної системи. Також можливе використання цього підходу при написанні мікропрограм (firmware) для вбудованих мікропроцесорів та спецпристроїв.

Другий підхід більш характерний при написанні системних засобів, що доповнюють можливості існуючої операційної системи, наприклад Windows.

Для збільшення ефективності програмування доцільно використовувати інтегровані середовища швидкої розробки (IDE, Visual Studio), проте, при створенні проектів слід відключати стандартні бібліотеки візуальних класів та компоненти високого рівня.

2.4 Розробка структури програми (програмного продукту)

2.4.1 Розробка структури при класичному (не об’єктно-орієнтованому) підході

Третій етап життєвого циклу програмного продукту є найвідповідальнішим і, часто – найоб’ємнішим, адже при використанні сучасних технологій та засобів розробки генерування програмного коду деколи є набагато простішим, майже тривіальним завданням, яке в деяких випадках навіть може бути проведено автоматично, без участі людини. Хоча в системному програмуванні такі задачі трапляються не так часто, як, скажімо, при розробці баз даних, проте, важливість правильного проектування структури має вирішальний вплив як на терміни та вартість виконання розробки, так і на її якість та можливості підтримки і вдосконалення.

Основною задачею, яка розв’язується на даному етапі є функціональна декомпозиція, що полягає в розділенні на окремі блоки (модулі, підпрограми,

функції), що призначені для виконання специфічних функцій у розроблюваному продукті. В залежності від специфіки теми, в курсовій роботі розробляється одна або декілька програмних одиниць, що в більшій чи меншій мірі пов’язані між собою.

Якщо програма є цілісною, то декомпозиція полягає у її розподілі на дрібніші структурні одиниці на основі аналізу алгоритму її роботи, даних та функціонального призначення. Якщо ж розробляється пакет програм, драйверів чи бібліотек, то декомпозиція стосуватиметься кожної програмної одиниці. В будь-якому випадку, розпочинати слід із розробки алгоритму роботи (або алгоритмів для кожної програмної одиниці).

Алгоритм розробляється, виходячи із поставленого завдання та технічних вимог до програми. Він може бути відображений як у вигляді блок-схеми, оформленій згідно з вимогами стандартів, так і словесним описом, що включає послідовність узагальнених операцій та переходів між ними. Такий опис може мати ієрархічну структуру, де спочатку записують більш загальні операції, які потім розшифровують на нижчому ієрархічному рівні.

Для деяких типів програм опис алгоритму зручніше проводити в термінах об'єктів, що взаємодіють між собою за допомогою повідомлень і подій. Тоді використовується об’єктно-орієнтована технологія проектування, в якій загальний алгоритм на етапі розробки сформулювати неможливо, оскільки послідовність операцій визначається самим користувачем на етапі виконання. До таких програм відносяться різноманітні редактори (текстові, графічні), програми опрацювання переривань, ігрові програми та інші. В цьому випадку алгоритм розпадається на кілька незалежних одна від одної послідовностей операцій, що реагують на події та керують опрацюванням повідомлень.

Для систем, що працюють у реальному часі, паралельних потоків та процесів, розподілених систем важливою є часова послідовність опрацювання повідомлень і подій. Для них, крім опису алгоритму, застосовують ще й часові діаграми, що характеризують послідовність виконання різних процесів та їх взаємодію з апаратними засобами.

У загальному, спосіб опису алгоритму вибирають, виходячи з конкретної задачі, однак він повинен бути повним, несуперечливим і достатнім для розуміння суті запропонованого способу її вирішення й принципу роботи спроектованої програми.

Аналізуючи отриманий алгоритм, потрібно визначити в ньому основні структурні частини, які дозволять здійснити декомпозицію (розбиття) проекту на завершені програмні одиниці (модулі, програми, бібліотеки). Це в більшості випадків спрощує як процес розробки, так і відлагодження програмного проекту.

Найчастіше окремі функції, класи та змінні групують за функціональним призначенням та розміщують в окремих програмних одиницях (модулях), які можна окремо компілювати. Для програмування на мові C++ кожен із модулів, як правило, розбивають на два файли: заголовний файл із розширенням *.h (*.hpp), що містить інтерфейсну частину модуля та файл реалізації модуля з розширенням *.cpp.

Крім модулів, потрібна головна програма, яка їх викликає. Для випадку, якщо завданням передбачено розробку повноцінної програми, то ця головна програма й реалізує необхідний алгоритм роботи. Якщо метою розробки є лише набір модулів або бібліотека функцій (наприклад, реалізується програмний засіб побудови графіків на основі даних, сформованих у програмі користувача або бібліотека для реалізації різноманітних операцій з новим пристроєм), то така головна програма буде лише тестовою, призначеною для відлагодження, перевірки та демонстрування можливостей модулів.

Для деяких задач виникає необхідність створення ще й допоміжних програм (утиліт), що виконують певні сервісні функції (наприклад, конвертування файлів, формування тестових даних, виведення графіків тощо). У такому випадку розв’язок задач реалізується у вигляді програмного пакета, що містить головну програму, модулі та допоміжні програми.

Згідно з указаною вище схемою потрібно провести декомпозицію проекту та відобразити її в даному розділі. Для кожного модуля чи функціональної

одиниці слів детально задати специфікацію (інтерфейс) у вигляді опису призначення, вхідних і вихідних параметрів, типів, змінних, констант, повідомлень, формату файлів тощо. За необхідності наводяться також тестові дані, за допомогою яких можна перевірити правильність роботи модуля чи функції.

Обсяг розділу — 2-5 сторінок.

2.4.2 Розробка структури програмного продукту при об’єктно-орієнтованому підході

На етапі проектування структури потрібно чітко уявити, з якими даними буде працювати програма (чи окремий модуль) та які дії над ними доведеться виконувати. Дані та дії над ними утворюють об'єкт. Також можна уявити програму як набір об'єктів (реально існуючих фізичних об'єктів або абстрактних понять, що існують лише в нашій уяві), які взаємодіють між собою.

Наступним етапом буде поділ об’єктів на групи, що мають певні властивості (дані, функціональність). У більшості випадків об’єкти будуть більш чи менш подібні. Деколи в групах можна буде виділити підгрупи й т. д., аж до складного ієрархічного дерева.

У кожній групі виділимо найзагальніші властивості, притаманні всім без винятку об'єктам у групі та дії над ними. Таким чином, отримаємо базовий клас для цієї групи об'єктів, даними якого будуть загальні параметри об'єктів групи, а методами — дії, які можна проводити над даними будь-якого об'єкта групи. Далі для кожної з підгруп (у межах виділеного класу об’єктів) додаємо притаманні лише їй властивості, як поля нового, породженого від базового класу. Додаткові дії, які властиві кожній підгрупі, стають методами породжених класів, а параметри, яких не мали об’єкти базового класу, — даними породжених (похідних) класів. Породження класів продовжується до тих пір, поки не будуть описані всі параметри об'єктів та дії, які необхідно над

ними здійснювати для розв'язку поставлених задач. Описана послідовність повторюється для кожної з групи об'єктів у програмі.

2.5 Реалізація програмного забезпечення (програмування)

2.5.1 Технології програмування

На даному етапі здійснюється реалізація розроблених на попередньому етапі програмних модулів чи програмних одиниць. Маючи алгоритм, опис вхідних та вихідних типів даних, інші специфікації залишається реалізувати конкретні функції, методи класів та інше на вибраній мові програмування.

У залежності від обраного стилю програмування, можливі різні підходи до розробки програми. У найпростіших випадках вона йде послідовно, алгоритм програми реалізується за допомогою послідовності операторів, включно з умовними та операторами циклу. Такі задачі в системному програмуванні зустрічаються доволі рідко, оскільки подібні лінійні алгоритми можуть бути реалізовані у існуючих системах у вигляді стандартних командних файлів, сценаріїв чи фільтрів.

При структурному програмуванні застосовується ієрархічний підхід: загальна задача розділена на модулі, які у свою чергу поділені на окремі функції, кожна з яких виконує певну дію. Кожна функція визначається своїм алгоритмом роботи, вхідними параметрами та результатом, який вона повертає. Опис функції в такому випадку виконується один раз у певному місці програми, а використовується вона багато разів, при цьому лише змінюються параметри, і не потрібно щоразу повторювати одні й ті ж оператори, які включені у функцію. Оскільки на попередньому етапі визначено її інтерфейс та описано алгоритм її роботи, то на даному залишається лише "перекласти" алгоритм роботи із словесного опису на вибрану мову програмування. Важливо правильно використовувати там, де це можливо, наявні системні функції, повідомлення чи переривання, особливо це стосується програм під Windows,

оскільки тільки таким чином можна забезпечити сумісність з апаратними засобами та новими версіями операційних систем.

При розробці віконних програм під Windows програмування в основному зводиться до реалізації інтерфейсу з користувачем, операційною системою та іншими програмами. Основна частина програмного коду розміщується у вигляді функцій вікон, як відповідають за обробку та генерування повідомлень. В таких програмах доводиться реалізовувати не один загальний алгоритм, а велику кількість маленьких процедур, що є реакцією програми на певні події.

При використанні об’єктно-орієнтованої технології програмування, можна, в деяких випадках, отримати в кілька разів вищу продуктивність роботи при розробці складних програмних продуктів у порівнянні зі структурною. Завдяки принципам інкапсуляції, успадкування та поліморфізму об’єктно-орієнтоване програмування дає можливість виділити всі спільні та відмінні риси окремих елементів програми і записати їх у програмі в найлаконічнішій формі. Досить зручним цей підхід є і при побудові віконних програм під Windows.

Фактично, розробка програми при об'єктно-орієнтованому підході починається ще на етапі проектування структури програми, коли програма чи модуль розбивається на класи, описуються глобальні типи даних, а на етапі реалізації проекту залишається лише виписати код методів, змінних та глобальних функцій. Створення об'єктів та організація взаємодії між ними відноситься до головної програми, яка розробляється на етапі інтегрування.

2.5.2 Реалізація програмного продукту при об’єктно-орієнтованій технології програмування

Після того, як завершена робота з інтерфейсною частиною класів, залишилося лише розробити реалізацію методів, змінних та глобальних функцій. Це завдання є досить простим, якщо на попередньому етапі було правильно спроектовано систему класів та детально пророблено інтерфейсні

частини. При програмуванні з використанням мови С чи С++, реалізація будь-якої функції (чи методу) розміщується в *.c (*.cpp) – частині модуля, на відміну від інтерфейсної, яка записується у *.h - файл. Потрібно лише акуратно записати ті дії, які має виконувати метод чи функція, не забуваючи користуватися вже готовими функціями та методами замість того, щоб увесь час змінювати дані за допомогою одних і тих же послідовностей операцій.

Обсяг розділу — 2-5 сторінок.

2.6 Інтеграція програми

П’ятим етапом життєвого циклу програмного продукту є інтегрування розроблених функціональних одиниць в цілісну програму чи бібліотеку. В залежності від обсягу робіт, що необхідно виконати для об’єднання скомпільованих одиниць, виправлення можливих помилок та спільного тестування даний етап відображається або в окремому розділі, або включається у розділ "Реалізація проекту".

При класичному підході на даному етапі пишуться код головної програми, підключаються створені модулі, викликаються функції, описуються глобальні змінні та виділяється пам’ять під об’єкти. Саме тут реалізується загальний алгоритм роботи програми, використовуючи реалізовані на попередньому етапі функціональні одиниці. При об’єктно-орієнтованому підході, інтегрування реалізує процес створення самих об’єктів у програмі, основні компоненти яких розроблені в попередніх підрозділах. Створивши об’єкти, ожемо реалізувати певну послідовність дій або схему взаємодії між об’єктами, яка призводить до вирішення поставленого завдання.

Опис цієї послідовності дій чи схеми взаємодії, можливо, з прикладами для реальних вхідних даних чи повідомлень і є основою даного розділу чи підрозділу. Такий опис повинен бути достатнім для розуміння всіх деталей реалізації алгоритму та можливих випадків, що зустрічаються при роботі

програми.

Обсяг розділу 1-4 сторінки.

2.7 Додаткові розділи

За необхідності у пояснювальну записку додаються такі додаткові розділи, підрозділи і питання:

- опис файлів даних та інтерфейсу;
- тестування програми;
- експлуатація і підтримка;
- утилізація програмного продукту.

2.8 Опис файлів даних та інтерфейсу програми

Якщо програма використовує файли як джерело вхідних даних або для зберігання проміжних чи кінцевих результатів роботи, то в даному підрозділі слід навести опис формату цих файлів. Він може бути виконаний у текстовому, табличному чи графічному вигляді. Можна додавати до нього приклади реальних файлів із даними розробленої програми.

Для інтерактивної програми з розвинутою системою меню та діалогових вікон у цьому підрозділі слід описати призначення елементів меню, роботу з ними, параметри, що вибираються в діалогових вікнах тощо. У цьому ж розділі слід описати інтерфейс програм, які працюють з параметрами командного рядка.

Якщо темою роботи є розробка модулів або бібліотеки, то слід детально описати порядок їх використання, навести в алфавітному порядку всі доступні для користувача глобальні типи даних і змінні, функції та класи.

2.9 Тестування програми і результати її виконання

Як вже зазначалося раніше, при сучасному підході тестування є невід'ємною частиною кожного з етапів життєвого циклу і тому проводиться постійно. Для порівняно простих програмних продуктів, які реалізуються в курсових роботах, для спрощення процес тестування можна виділити в окремий розділ для проекту в цілому, або в підрозділи для кожного з етапів життєвого циклу. У даному розділі треба описати методику тестування програми, тестові дані та навести результати роботи програми. Якщо програма працює в графічному режимі, то слід роздрукувати копію графічного вікна програми. Якщо результатом роботи програми є текстовий файл, то необхідно вивести вміст цього файлу. Для програм з розвинутою системою діалогових вікон і меню слід обмежитися видруком лише найсуттєвіших результатів, які демонструють правильну роботу програми, а не передруковувати весь екран для кожного відкритого пункту меню. Перелік усіх пунктів меню в такому випадку та вміст неосновних діалогових вікон можна подати в текстовому вигляді.

Якщо для відображення роботи програми необхідна значна кількість роздруків, то їх можна подати в додатках.

Обсяг розділу — 1-4 сторінки. У розділі потрібно зробити висновок, який підтверджує (або заперечує) працездатність програми та її відповідність поставленому завданню.

2.10 Висновки

На підставі власних досліджень автором повинні бути написані висновки, у вигляді коротко сформульованих і пронумерованих тез. Кількість висновків залежить від обсягу отриманих результатів і складності програми та прикладних задач.

Обсяг висновків — 1-2 сторінки.

2.11 Список використаних джерел

Перелік використаної літератури оформляти відповідно до чинних стандартів. Приклади його оформлення наведено в додатку Г.

2.12 Додатки

Матеріал, що доповнює текст пояснювальної записки, можна розміщувати в розділі "Додатки". У них можуть бути, наприклад, коди програм, графіки та таблиці. У тексті записки на всі додатки потрібно дати посилання, розміщувати додатки слід у порядку посилань на них у тексті.

Кожний додаток треба розпочинати з нової сторінки із зазначенням зверху посередині сторінки слова "Додаток" і його позначення. Згідно з ДСТУ 3008 – 95, додатки слід позначати послідовно великими літерами української абетки, за винятком літер Ґ, Є, З, І, Ї, О, Ч, Ь, наприклад: додаток А, додаток Б, і т. д. Оформляти додатки можна на аркушах формату А4, А3, А2, А1 згідно з ГОСТ 2.301 - 68.

Першим додатком повинен бути текст програми, розробленої в курсовій роботі або алгоритми роботи програми чи її компонентів. Якщо при розробці використовувалися технології візуального програмування та при значному обсязі коду програми, допускається наводити лише тексти основних модулів програми.

Ілюстрації кожного додатка позначають окремою нумерацією з додаванням перед арабською цифрою буквеного позначення додатка, наприклад: рисунок А.2, рисунок В.3 і т.д. Додатки повинні мати окрему від курсової роботи наскрізну нумерацію сторінок і бути перелічені в змісті роботи із зазначенням їх позначень та заголовків.

3 ОФОРМЛЕННЯ КУРСОВОЇ РОБОТИ

3.1 Вимоги до оформлення текстових документів

Пояснювальна записка (ПЗ) повинна розкривати зміст курсової роботи, містити обґрунтування вибору методів, алгоритмів та програм для виконання поставленого завдання, аналіз отриманих результатів та інші матеріали.

Матеріал пояснювальної записки повинен бути викладений грамотно, чітко та стисло. При цьому, в тексті доцільно подавати посилання на використані літературні та інші джерела.

У тексті пояснювальної записки не рекомендується вживати звороти із займенниками першої особи, наприклад: "Я вважаю ...", "Ми вважаємо ..." тощо. Рекомендується вести виклад, не вживаючи займенників, наприклад: "Вважаємо ...", "... знаходимо ..." тощо.

Без пояснень дозволяється використовувати тільки загальноприйняті скорочення, наприклад: ПЕОМ, ДСТУ, ООП тощо.

ПЗ до курсової роботи виконують на аркушах білого паперу (з одного боку) формату А4 (210 х 297 мм) за формами 5 і 5а (ГОСТ 2.106-68), згідно вимог ГОСТ 2.105-79 та ДСТУ 3008-95, українською мовою одним із наведених нижче способів:

– машинописним, при цьому слід дотримуватися вимог ГОСТ 13.1.002 - 80. Шрифт друкарської машинки повинен бути чітким, висотою не менше 2,5 мм, стрічка тільки чорного кольору (напівжирна);

– рукописним — креслярським шрифтом (ГОСТ 2.304-81) з висотою літер і цифр не менше 2,5 мм. Цифри і літери необхідно писати чітко чорною тушшю. Написання формул, цифр, заголовків розділів і підрозділів, заповнення таблиць виконувати тільки шрифтом згідно з ГОСТ 2.304-81;

– із застосуванням друкуючих і графічних пристроїв виведення ЕОМ (ГОСТ 2.004-88). Обсяг ПЗ повинен складати 20-25 сторінок машинописного тексту (без урахування додатків), надрукованого через 2 інтервали (до 30 рядків

на аркуші А4), з полями: верхнє та нижнє – по 2 см, праве – 1,5 см, а ліве – 3 см.

Допускаються рукописні роботи, написані гарним, розбірливим почерком, обсягом 25-40 сторінок. При наборі тексту на комп'ютері розмір шрифту слід вибирати рівним 14 пунктів, гарнітуру – Times New Roman (або аналогічну за виглядом), міжрядковий інтервал – 1,5, абзацний відступ для першого рядка – 1,27..1,7 см.

ПЗ повинна починатися з титульного аркуша. Виконують його згідно з ГОСТ 2.105-95 на аркуші формату А4 за формою, наведеною у додатку А. Далі розміщують завдання на КР, анотацію та список скорочень (за необхідністю), зміст, основний текст, перелік використаної літератури та додатки.

Нумерацію сторінок ПЗ починають із титульного аркуша, на якому номер не проставляють. Аркуш, розміщений після завдання на КР, нумерують цифрою 3.

ПЗ поділяють на розділи і підрозділи, пункти і підпункти.

Розділи в межах усієї пояснювальної записки повинні мати порядкові номери, позначені арабськими цифрами без крапки.

Підрозділи повинні мати нумерацію в межах розділу: номер підрозділу складається з номера розділу і підрозділу, розділених крапкою, наприклад, 2.3 означає: третій підрозділ другого розділу. У кінці порядкового номера розділу, підрозділу й т. п. крапки не ставлять.

Номер пункту вміщує номер розділу, підрозділу і пункту, які розділені крапками, наприклад, 3.2.1 – перший пункт другого підрозділу третього розділу.

Назви розділів повинні бути короткими і записувати їх слід у вигляді заголовків великими буквами посередині рядка. Назви підрозділів записують у вигляді заголовків меншими буквами (перша велика). Переноси слів у заголовках не допускаються. Крапку в кінці заголовка не ставлять. Між назвами розділів, підрозділів і основним текстом повинен бути пропущений рядок.

Заголовки розділів відділяються від тексту зверху й знизу трьома інтервалами (30..36 пунктів на комп'ютері). Знаки, букви, символи, позначення,

які відсутні в друкарських машинках, а також математичні й хімічні формули, іноземні прізвища та слова можна вписувати від руки чорнилом (пастою) чорного кольору.

Абзаци в тексті починають відступом, що дорівнює п'яти ударам друкарської машини (13-17 мм).

Графічний матеріал у тексті ПЗ (схеми, ескізи, графіки, рисунки) виконують у графічному редакторі, або тушшю чи олівцем.

Кількість ілюстрацій повинна бути достатньою для пояснення тексту, що викладається. Ілюстрації розміщують одразу після посилання на них за текстом ПЗ.

Усі розміщені в ПЗ ілюстрації нумерують арабськими цифрами в межах одного розділу, наприклад, Рисунок 2.3 – розділ 2, рисунок 3.

Посилання на ілюстрації подають за типом: на рис. 2.3, повторно – див. рис. 1.3.

Помилки, описки і графічні неточності, виявлені в процесі виконання документа, допускається виправляти підчищенням або зафарбовуванням білою фарбою і нанесенням на тому ж місці виправленого тексту (графіки) машинописним способом або чорним чорнилом, пастою або тушшю рукописним способом.

Пошкодження аркушів пояснювальної записки, помарки і сліди неповністю видаленого попереднього тексту (графіки) не допускаються.

Порядкові чисельники, які йдуть один за одним, можуть бути написані цифрами з відмінковим закінченням, яке ставлять лише при останній цифрі, наприклад: 1-е; 7, 8, 9-й тощо.

Перелік використаної літератури повинен містити лише ті літературні джерела, які використані при виконанні курсової роботи і на які є посилання в тексті документу.

Робота повинна бути написана державною мовою.

3.2 Позначення документів

Кожному документу курсової роботи присвоюється позначення. Згідно з ГОСТ 19.103-77 (єдина система програмної документації) воно повинно мати наступну структуру (рис. 3.1):

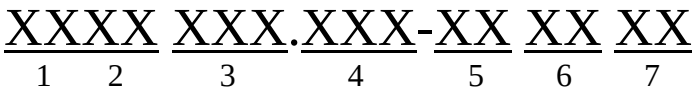

Рисунок 3.1 Структура позначення програм і програмних документів

Позначення програмних документів здійснюється наступним чином:

– перша група — скорочена назва кафедри (структурного підрозділу), на якій виконувалась робота (для кафедри комп'ютерних систем та мереж – КС);

– друга група – скорочена назва виду роботи (курсова робота – КР);

– третя група – дві останні цифри номера групи;

– четверта група – три цифри номера спеціальності (123);

– п'ята група – три останні цифри номера залікової книжки;

– шоста група – цифровий код виду документу, згідно з ГОСТ 19.101-77 (для пояснювальної записки – 81, для тексту програми – 12);

– сьома група – номер документу даного виду (якщо розробляється один документ пояснювальної записки, то в даній групі проставляється 01).

Наприклад, пояснювальна записка до курсової роботи студента групи СІ-32, номер залікової книжки якого 04-033, буде мати позначення:

КСКР 123.033-32 81 01

4 АКАДЕМІЧНА ДОБРОЧЕСНІСТЬ

Учасники освітнього процесу під час навчання, викладання та провадження наукової діяльності дотримуються принципів академічної доброчесності [5] і усвідомлюють наслідки порушення цих принципів.

З метою підвищення якості навчання, розвитку навичок коректної роботи із джерелами інформації та формування звички до сумлінного дотримання вимог наукової етики, активізації самостійності та індивідуальності при створенні авторського твору та підвищення відповідальності студентів усіх форм навчання за порушення правил академічної етики діє Положення про недопущення академічного плагіату в Тернопільському національному технічному університеті імені Івана Пулюя [6].

Основні поняття і визначення:

Автор – фізична особа, яка своєю творчою працею створила твір.

Академічна доброчесність – це сукупність етичних принципів та визначених законодавством правил, якими мають керуватися учасники освітнього процесу під час навчання, викладання та провадження наукової (творчої) діяльності з метою забезпечення довіри до результатів навчання та/або наукових (творчих) досягнень.

Плагіат – оприлюднення (опублікування), повністю або частково, чужого твору під іменем особи, яка не є автором цього твору.

Плагіат академічний − оприлюднення (частково або повністю) наукових (творчих) результатів, отриманих іншими особами, як результатів власного дослідження (творчості) та/або відтворення опублікованих текстів (оприлюднених творів мистецтва) інших авторів без зазначення авторства.

Оприлюднення твору – здійснена за згодою автора чи іншого суб'єкта авторського права і (або) суміжних прав дія, що вперше робить твір доступним для публіки шляхом опублікування, публічного виконання, публічного показу, публічної демонстрації, публічного сповіщення тощо.

Показник оригінальності твору – кількісний показник, виражений у відсотках, який відображає співвідношення авторського тексту до загального обсягу твору.

Твір – результат наукової чи навчально-методичної діяльності автора (співавторів) поданий в університет на паперових носіях або в електронному вигляді, оприлюднений у мережі Інтернет чи на офіційних web-ресурсах університету у формі монографії, підручника, навчального посібника, статті, тез, препринта, автореферату і рукопису дисертації (дисертаційної роботи), дипломної роботи, курсової роботи чи проєкту, реферату, есе, контрольної роботи, тощо.

Цитата – порівняно короткий уривок з літературного, наукового чи будь-якого іншого опублікованого твору, який використовується, з обов'язковим посиланням на його автора і джерела цитування, іншою особою у своєму творі з метою зробити зрозумілішими свої твердження або для посилання на погляди іншого автора в автентичному формулюванні.

Різновиди академічного плагіату:

– відтворення в тексті роботи – без змін, з незначними змінами, або в перекладі – тексту іншого автора (інших авторів), обсягом від речення і більше, без посилання на автора (авторів) відтвореного тексту;

– відтворення в тексті роботи, повністю або частково, тексту іншого автора (інших авторів) через його перефразування чи довільний переказ без посилання на автора (авторів) відтвореного тексту;

– відтворення в тексті роботи наведених в іншому джерелі цитат з третіх джерел без вказування, за яким саме безпосереднім джерелом наведена цитата;

– відтворення в тексті роботи наведеної в іншому джерелі науково-технічної інформації (крім загальновідомої) без вказування на те, з якого джерела взята ця інформація;

– відтворення в тексті роботи оприлюднених творів мистецтва без зазначення авторства цих творів мистецтва;

– парафраза – переказ своїми словами чужих думок, ідей або тексту.

Сутність парафрази полягає в заміні слів (знаків), фразеологічних оборотів або пропозицій при використанні будь-якої авторської наукової праці (збереженої на електронних або паперових носіях, у тому числі розміщеної в мережі Інтернет).

Неприпустимою є також компіляція – створення значного масиву тексту без поглибленого вивчення проблеми шляхом копіювання тексту із низки джерел, з посиланням на авторів та «маскуванням» шляхом написання перехідних речень між скопійованими частинами тексту.

Перед перевіркою і допуском до захисту курсових робіт викладач (керівник) попередньо перевіряє оригінальність електронних версій текстових документів (пояснювальних записок) цих робіт зі встановленням частки оригінального тексту з використанням інтегрованого сервісу перевірки вмісту скриньок для завдань файлообмінника системи електронного навчання Atutor (https://dl.tntu.edu.ua/) відповідного електронного навчального курсу або один з програмно-технічних засобів, які знаходяться у відкритому доступі у мережі Інтернет та визнані наукової спільнотою.

Рекомендовані показники оригінальності навчальних робіт такі:

– понад 80% - текст вважається оригінальним;

– від 60 до 80% - оригінальність задовільна, слід пересвідчитись у наявності посилань на першоджерела для цитованих фрагментів;

– від 40 до 60% - матеріал приймається, але його слід доопрацювати й перевірити на наявність посилань на першоджерела для цитованих фрагментів;

– менше 40% - матеріал до розгляду не приймається.

Програмно-технічні засоби перевірки на академічний плагіат є допоміжним засобом перевірки робіт на предмет виявлення фактів та обсягу неправомірних запозичень у поданій роботі.

5 ЗАХИСТ І ОЦІНЮВАННЯ РОБОТИ

5.1 Підготовка до захисту

Виконану згідно зі стандартами відповідно до завдання і у повному обсязі курсову роботу, підписану виконавцем, у незброшурованому вигляді треба подати на перевірку керівникові. Текст програми подається в додатках, якщо його обсяг перевищує 20 аркушів формату А4, то допускається в друкованому варіанті подавати лише основну частину програмного коду, а решту – в електронному варіанті. Якщо програма вимагає для своєї роботи додаткових динамічних бібліотек (DLL), які не входять до складу операційної системи, то їх також необхідно додати або вказати назву та джерело, з якого можна зчитати ці файли.

Роботу необхідно подати на перевірку не пізніше, як за три робочих дні до захисту. Виявлені при перевірці курсової роботи неточності й помилки студент зобов'язаний виправити, а результати представити керівникові у визначені терміни. Якщо ж при огляді встановлено, що робота в будь-якій частині потребує суттєвого доопрацювання, то визначається обсяг доопрацювання і встановлюється термін подання виправленої роботи на повторну перевірку.

Роботи, що не відповідають затвердженій темі, без затвердженого завдання на курсову роботу, підписаного студентом і викладачем, а також ті, в яких виявлено запозичення з інших джерел, без посилання на джерело, до захисту не допускаються.

5.2 Порядок захисту

До захисту курсової роботи допускаються студенти, які виконали всі вимоги навчальної програми та календарного плану, своєчасно представили

роботу й усі необхідні матеріали.

Захист курсових робіт проводиться відкрито у відповідності з установленим графіком. На захист роботу слід представляти тільки в зброшурованому вигляді.

Захист курсової роботи проходить у такій послідовності:

– доповідь студента про основні результати виконаної роботи;
– відповіді студента на запитання присутніх;
– обговорення доповіді;
– відповіді на зауваження.

Для доповіді про результати виконаної роботи студенту надається 5..10 хвилин. Доповідь повинна складатися з трьох частин (вступна та основна частини й висновки).

У вступній частині доповіді необхідно відзначити актуальність теми, в загальному проаналізувати стан питання, сформулювати основні задачі, з розв’язуванням яких пов'язано виконання роботи.

В основній частині доповіді необхідно навести короткі відомості про зміст виконаних досліджень, відзначити основні підходи та показати ефективність прийнятих рішень, навести короткі відомості про отримані результати. Основну частину доповіді можна супроводжувати посиланням на графічні матеріали та демонструвати роботу програми.

У висновках необхідно чітко сформулювати основні результати курсової роботи, наголосивши на повноті розв’язання поставленого завдання.

Відповіді на запитання повинні бути короткими, за суттю й не виходити за межі поставленого запитання.

5.3 Критерії оцінювання курсової роботи

При визначенні оцінки курсової роботи береться до уваги рівень теоретичної й практичної підготовки студента, виконання ним затвердженого

плану, якість прийнятих інженерних рішень при виконанні курсової роботи.

Оцінювання курсової роботи проводиться за стобальною системою. У випадку чіткого виконання вимог затвердженого плану при проходженні першого, другого та третього модульного контролю за кредитно-модульною системою, студент може отримати максимум по 25 балів за кожен модуль. Решта 25 балів визначається під час захисту роботи.

Максимальну оцінку 25 балів, за окремі розділи, що виносяться на контроль відповідно до затвердженого плану, та за роботу в цілому можна отримати у випадку, коли задовольняються всі перераховані нижче вимоги:

– якщо в роботі немає суттєвих недоліків;

– програма повністю виконує поставлене завдання;

– при розробці програми вміло використано переваги сучасних технологій програмування та існуючі модулі і бібліотеки;

– при захисті роботи студент аргументовано виклав основні технічні рішення, прийняті в процесі розробки та відповів на поставлені запитання;

– робота виконана самостійно.

Оцінку в 20 балів можна отримати, якщо: - у роботі немає суттєвих недоліків;

– програма повністю виконує поставлене завдання, але містить деякі незначні помилки;

– при розробці програми не використано переваг сучасних технологій програмування, програма реалізована не оптимальним чином;

– прийняті в роботі рішення не є достатнім чином обґрунтовані;

– при захисті роботи студентом були допущені неточності, або не було аргументованих відповідей на деякі з поставлених запитань.

Оцінку в 15 балів можна отримати, якщо:

– у роботі є недоліки, однак програма в основному виконує поставлені завдання;

– містить незначні помилки, які не дозволяють використовувати її для

деяких комбінацій вхідних параметрів;

– при розробці програми не використано сучасних технологій програмування;

– робота оформлена зі значними відхиленнями від стандартів і вимог, або в процесі проектування були відхилення від затвердженого календарного плану чи завдання;

– при захисті роботи студентом допущені неточності, але вони були виправлені студентом в процесі відповідей на запитання.

Оцінку в 10 балів можна отримати, якщо:

– у роботі є недоліки програма не повністю виконує поставлене завдання;

– програма містить значні помилки, які не дозволяють використовувати її для деяких комбінацій вхідних параметрів;

– при розробці програми не використано сучасних технологій програмування;

– робота оформлена зі значними відхиленнями від стандартів і вимог, або в процесі проектування були відхилення від затвердженого календарного плану чи завдання;

– при захисті роботи студентом допущені суттєві неточності, або не було аргументованих відповідей на поставлені запитання.

Робота оцінюється в 5 балів і нижче, якщо:

– програма не виконує поставленого завдання, або - робота не оформлена належними чином;

– якщо студент систематично порушував календарний план, не виконав більшу частину завдання;

– програма містить значні недоліки, а наявні помилки не дають можливості встановити її працездатність для реальних вхідних даних.

Якщо під час захисту виявилося, робота виконана студентом не самостійно, при її захисті не було обґрунтовано прийняті рішення, а запитання,

які задавались, залишились без відповіді, або у випадку, якщо студент не з’явився на захист без поважної причини, за роботу виставляється 0 балів.

Подальша процедура захисту робіт студентів, які не з’явилися на основний захист, а також у випадку, якщо студента не задовольняє отримана оцінка визначається чинними правилами Університету.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Рисований О.М. Системне програмування : навч. посіб. Для студентів напрямку «Комп'ютерна інженерія» вищих навчальних закладів у 4 ч. Ч. І. Основи асемблера та його використання. / О. М. Рисований. Харків : «Слово», 2015. 336 с.
2. Рисований О.М. Системне програмування : навч. посіб. Для студентів напрямку «Комп'ютерна інженерія» вищих навчальних закладів у 4 ч. Ч.2. Розширені можливості програмування на асемблері. / О. М. Рисований. Харків : «Слово», 2015. 224 с.
3. Паламар М.І., Стрембіцький М.О., Паламар А.М. Проектування комп'ютеризованих вимірювальних систем і комплексів. Навчальний посібник. Тернопіль: ТНТУ. 2019. 150 с.
4. Електронний навчальний курс "Системне програмування" (ID: 1989), доступний за адресою https://dl.tntu.edu.ua/bounce.php?course=1989.
5. Положення про академічну доброчесність учасників освітнього процесу Тернопільського національного технічного університету імені Івана Пулюя: наказ №4/7-969 від 01.11.2019. URL: https://docs.tntu.edu.ua/base/document?id=465.
6. Положення про недопущення академічного плагіату в Тернопільському національному технічному університеті імені Івана Пулюя: наказ №4/7-964 від 01.11.2019 (зі змінами від 19.12.2019 наказ №4/7-114 від 12.02.2020, зі змінами від 26.01.2021 - наказ №4/7-72 від 02.02.2021). URL: https://docs.tntu.edu.ua/base/document?id=462.

ДОДАТКИ

Додаток А. Бланк титульного аркуша курсової роботи

Міністерство освіти і науки України

Тернопільський національний технічний університет імені Івана Пулюя

(повне найменування вищого навчального закладу)

(повна назва кафедри)

КУРСОВА РОБОТА

з _____

(назва дисципліни)

на тему: _____

_____

_____

Студента (ки) _____ курсу, групи _____

_____

спеціальності _____

_____

_____

(прізвище та ініціали)

Керівник: _____

_____

(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Оцінка за національною шкалою _____

Кількість балів: _____ Оцінка ECTS _____

Члени комісії: _____ _____

(підпис) (прізвище та ініціали)

_____ _____

(підпис) (прізвище та ініціали)

_____ _____

(підпис) (прізвище та ініціали)

м. Тернопіль – 202 _

Додаток Б. Бланк завдання на курсову роботу

Міністерство освіти і науки України

Тернопільський національний технічний університет імені Івана Пулюя

Кафедра _____

Напрям підготовки _____

Спеціальність _____

Курс _____ Група _____ Семестр _____

ЗАВДАННЯ

на курсову роботу

Студентові _____

(прізвище, ім'я, по батькові)

1. Тема роботи _____

2. Термін здачі студентом закінченої роботи _____

3. Вихідні дані до роботи _____

4. Зміст розрахунково-пояснювальної записки (перелік питань, які підлягають розробці) _____

5. Перелік графічного (ілюстративного) матеріалу, якщо передбачено _____

6. Дата видачі завдання _____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів курсової роботи	Термін виконання етапів роботи	Примітка

Студент _____ _____
(підпис) (прізвище, ім’я, по батькові)

Керівник роботи _____ _____
(підпис) (вчений ступінь, посада, прізвище, ім’я, по батькові)

Додаток В. Рекомендована структура курсової роботи

Назва	**Кількість сторінок**
ТИТУЛЬНА СТОРІНКА	1
ЗАВДАННЯ НА КУРСОВИЙ ПРОЕКТ	1 (двостороння)
ЗМІСТ	1-2
ВСТУП	1-2
1 СТРУКТУРА *ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ*	3-7
1.1 Аналіз завдання на курсову роботу	
1.2 Обґрунтування алгоритму і структури програми	
1.3 Розробка блок-схем алгоритмів окремих процедур	
2 РЕАЛІЗАЦІЯ *ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ*	10-12
2.1 Технології програмування, засоби розробки	
2.2 Створення об’єктів та розробка головної програми	
2.3 Розробка окремих процедур і функцій (класів, методів)	
3 ОПИС ФУНКЦІОНУВАННЯ ПРОГРАМИ	8-10
3.1 Опис процедур і функцій	
3.2 Опис вхідних, вихідних даних	
3.3 Опис зовнішніх інтерфейсів	
4 ТЕСТУВАННЯ ПРОГРАМИ ТА РЕЗУЛЬТАТИ ЇЇ ВИКОНАННЯ	3-5
4.1 Результати виконання *програмного забезпечення*	
4.2 Інструкція користувача	
ВИСНОВКИ	1-2
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	1-2
Додаток А. Лістинг програми	

Мінімальна кількість сторінок пояснювальної записки курсової роботи – 25-30 сторінок (без додатків).

Замість тексту, написаного *курсивом*, вставити назву своєї теми курсової роботи.