МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
“ЛЬВІВСЬКА ПОЛІТЕХНІКА”

СИСТЕМНЕ ПРОГРАМУВАННЯ

МЕТОДИЧНІ ВКАЗІВКИ
до виконання курсового проєкту
для студентів базового напрямку “Комп’ютерна інженерія”

на тему
“Розробка системних програмних модулів та компонент систем програмування”

Індивідуальне завдання

“Розробка транслятора з вхідної мови програмування”

Львів 2024

Системне програмування: Методичні вказівки до виконання курсового проєкту для студентів базового напрямку “Комп’ютерна інженерія”

Укладачі Мархивка В. С., старший викладач каф. ЕОМ
Ногаль М. В., старший викладач каф. ЕОМ

Відповідальний за випуск

Рецензент

МЕТА КУРСОВОГО ПРОЄКТУ

Метою курсового проєктування з дисципліни “Системне програмування” є опанування студентами теоретичних аспектів і методик та отримання практичних навиків з проєктування і реалізації мовних процесорів з використанням новітніх технологій програмування; вироблення у студентів чітких систематизованих знань про особливості функціонування трансляторів і компіляторів мов програмування, принципів та способів їх побудови, а також використовуваних структур даних.

ТЕМАТИКА КУРСОВОГО ПРОЄКТУВАННЯ

Уніфікована тематика курсового проектування “**Розробка системних програмних модулів та компонент систем програмування**”.

Індивідуальне завдання “**Розробка транслятора з вхідної мови програмування**”.

В проекті підлягають реалізації:

- створення граматики для заданої мови програмування;
- проектування та реалізація інформаційних таблиць для транслятора;
- розробка та реалізація модуля лексичного аналізатора;
- розробка та реалізація модуля синтаксичного аналізатора;
- розробка та реалізація модуля семантичного аналізатора;
- розробка генератора коду.

Варіант вхідної мови програмування формується індивідуально для кожного студента керівником курсового проєкту. Після чого студент розробляє деталізований опис вхідної мови програмування і погоджує його з викладачем.

ЗАВДАННЯ НА КУРСОВИЙ ПРОЄКТ

1. Цільова мова транслятора – мова програмування С або асемблер для 32/64 розрядного процесора.
2. Для отримання виконуваного файлу на виході розробленого транслятора скористатися середовищем Microsoft Visual Studio або будь-яким іншим.
3. Мова розробки транслятора: C/C++.
4. Реалізувати графічну оболонку або інтерфейс з командного рядка.
5. На вхід розробленого транслятора має подаватися текстовий файл, написаний на заданій мові програмування.
6. На виході розробленого транслятора мають створюватись такі файли:

   файл з лексемами;
   файл з повідомленнями про помилки (або про їх відсутність);
   файл на мові С або асемблера;
   об’єктний файл;
   виконуваний файл.
7. Назва вхідної мови програмування утворюється від першої букви у прізвищі студента та останніх двох цифр номера його варіанту. Саме таке розширення повинні мати текстові файли, написані на цій мові програмування.
8. В додатки пояснювальної записки включають:

   таблиці лексем для тестових прикладів;
   С код (або код на асемблері), отриманий на виході транслятора для тестових прикладів;
   документований текст програмних модулів (лістинги);
   листи графічної частини проєкту (графічна частина є обов’язковою);
   інші матеріали (за потребою).

Деталізація завдання на проектування:

1. В кожному завданні передбачається блок оголошення змінних; змінні зберігають значення цілих чисел і, в залежності від варіанту, можуть бути 16/32 розрядними. За потребою можна реалізувати логічний тип даних.
2. Необхідно реалізувати арифметичні операції – додавання, віднімання, множення, ділення, залишок від ділення; операції порівняння – перевірка на рівність і нерівність, більше і менше; логічні операції – заперечення, “логічне І” і “логічне АБО”.

   Пріоритет операцій наступний – круглі дужки (), логічне заперечення, мультиплікативні (множення, ділення, залишок від ділення), адитивні (додавання, віднімання), відношення (більше, менше), перевірка на рівність і нерівність, логічне І, логічне АБО.
3. За допомогою оператора вводу можна зчитати з клавіатури значення змінної; за допомогою оператора виводу можна вивести на екран значення змінної, виразу чи цілої константи.
4. В кожному завданні обов’язковим є оператор присвоєння за допомогою якого можна реалізувати обчислення виразів з використанням заданих операцій і операції круглі дужки (); у якості операндів можуть бути цілі константи, змінні, а також інші вирази.
5. В кожному завданні обов’язковим є оператор типу “блок” (складений оператор), його вигляд має бути таким, як і блок тіла програми.
6. Необхідно реалізувати задані варіантом оператори, синтаксис операторів наведено у таблиці 1.1. Синтаксис вхідної мови має забезпечити реалізацію обчислень лінійних алгоритмів, алгоритмів з розгалуженням і циклічних алгоритмів. Опис формальної мови студент погоджує з викладачем.
7. Оператори можуть бути довільної вкладеності і в будь-якій послідовності.
8. Для перевірки роботи розробленого транслятора, необхідно написати три тестові програми на вхідній мові програмування.

Індивідуальні завдання на курсовий проєкт студент отримує у викладача!

Таблиця 1.1

Синтаксис операторів

Мова	Оператор	Синтаксис	Примітки

Бейсік	**for – to – next**	`for змінна = початок to кінець`
		`  оператори`
		`next змінна`	

	if – then goto; goto	`if умова then goto мітка1` `goto мітка2`	необхідно реалізувати два оператори: `if` і `goto`

	if – goto; goto	`if умова`
		`  goto мітка1`
		`goto мітка2`
			необхідно реалізувати два оператори: `if` і `goto`

Паскаль	**for – to – do**	`for змінна = початок to кінець do` `  оператор`	
	for – downto – do	`for змінна = кінець downto початок do` `  оператор`	
	while – do	`while (умова) do` `  оператор`	

	repeat – until	`repeat`
		`  оператори`
		`until (умова)`	

	if – then [–else]	`if (умова) then оператор`
		`if (умова) then оператор1`
		`  else оператор2`
			необхідно реалізувати дві форми – скорочену і повну

Rust	**Loop; break**	`loop оператор` `break`	(безумовний цикл, для виходу з якого необхідно ще реалізувати оператор `break`)
	while	`while умова` `  оператор`	
	for – in	`for змінна in початок..кінець` `  оператор`	

	if [–else]	`if умова оператор`
		`if умова оператор1`
		`  else оператор2`
			необхідно реалізувати дві форми – скорочену і повну

Ruby	**if [–else]**	`if умова операторі end`
		`if умова операторі`
		`  else операторі end`
			необхідно реалізувати дві форми – скорочену і повну

	for – in	`for змінна in початок..кінець` `  оператор`	

	while	`while умова do`
		`  операторі`
		`end`	

	until	`until умова do`
		`  операторі`
		`end`
			(цикл виконується доти, поки умова є хибною)

Swift	**if [–else]**	`if (умова) оператор`
		`if (умова) оператор1`
		`  else оператор2`
			необхідно реалізувати дві форми – скорочену і повну

	for – in	`for змінна in (початок.. кінець)` `  оператор`	
	while	`while (умова)` `  оператор`	

	repeat – while	`repeat`
		`  оператор`
		`while (умова)`	

Тестові програми для перевірки роботи розробленого транслятора:

Назва тестової програми має бути на початку програми, записана як коментар.

Символічні позначення операторів, знаків операцій і ідентифікаторів мають відповідати індивідуальному завданню!

Опис даних тестових задач і результати їх виконання мають бути наведені у 4-му розділі пояснювальної записки!

Тестова програма «*Лінійний алгоритм*»

1. Ввести два числа `A` і `B` (імена змінних можуть бути іншими і мають відповідати правилам запису ідентифікаторів згідно індивідуального завдання).
2. Вивести на екран:

   `A + B` (результат операції додавання);
   `A - B` (результат операції віднімання);
   `A * B` (результат операції множення);
   `A / B` (результат операції ділення);
   `A % B` (результат операції отримання залишку від ділення).

3. Обрахувати значення виразів

```
X = (A - B) * 10 + (A + B) / 10
У = X + X % 10
```

4. Вивести значення `X` і `У` на екран.

Тестова програма «*Алгоритм з розгалуженням*»

1. Ввести три числа `A`, `B`, `C` (імена змінних можуть бути іншими і мають відповідати правилам запису ідентифікаторів згідно індивідуального завдання).

Використання вкладеного умовного оператора:

2. Знайти найбільше з них і вивести його на екран.

Використання простого умовного оператора:

3. Вивести на екран число `1`, якщо усі числа однакові (логічний вираз в умовному операторі має виглядати так: «`(A=B) і (A=C) і (B=C)`»), інакше вивести `0`.
4. Вивести на екран число `-1`, якщо хоча б одне з чисел від'ємне (логічний вираз в умовному операторі має виглядати так: «`(A<0) або (B<0) або (C<0)`»), інакше вивести `0`.
5. Вивести на екран число `10`, якщо число `A` більше за суму чисел `B` і `C` (логічний вираз в умовному операторі має виглядати так: «`!(A<(B+C))`»), інакше вивести `0`.

Тестова програма «*Циклічний алгоритм*»

1. Ввести два числа `A` і `B`, причому `A<B` (імена змінних можуть бути іншими і мають відповідати правилам запису ідентифікаторів згідно індивідуального завдання).

Використання простого оператора циклу:

2. Вивести на екран квадрати чисел від A до B включно.

Використання вкладеного оператора циклу:

3. Обрахувати `X=A*B` за наступним алгоритмом:

```
X = 0
Цикл від 1 до A з кроком 1
    Цикл від 1 до B з кроком 1
         X = X + 1
```

4. Вивести значення `X` на екран.

ПОРЯДОК ВИКОНАННЯ КУРСОВОГО ПРОЄКТУ

Курсовий проєкт починається з отримання від керівника індивідуального завдання. У процесі проектування студент консультується з керівником у разі потреби й у зв'язку з виникаючими питаннями.

Рекомендується виконання курсового проєкту рівномірно розподілити протягом всього семестру та розбити на наступні етапи:

Підготовчий етап. Цей етап формує перший розділ пояснювальної записки до курсового проєкту. Студент повинний зрозуміти поставлену перед ним задачу, ознайомитися з рекомендованою літературою. На даному етапі потрібно описати вхідні та вихідні дані, сформулювати вимоги до функціонування транслятора.

Опис вхідної мови програмування. Цей етап формує другий розділ пояснювальної записки до курсового проєкту. На цьому етапі студент повинен описати граматику вхідної мови програмування згідно індивідуального завдання. Необхідно визначити алфавіт мови та набір зарезервованих слів.

Реалізація. Цей етап формує третій розділ пояснювальної записки до курсового проєкту. Проектується лексичний аналізатор з представленням блок-схеми алгоритму та описом програми його реалізації; проектується синтаксичний аналізатор на основі дерева граматичного розбору з представленням блок-схеми алгоритму та описом програми його реалізації; проектується генератор коду з представленням блок-схеми алгоритму та описом програми його реалізації.

Налагодження і тестування. Цей етап формує четвертий розділ пояснювальної записки до курсового проєкту. Розробляються інструкції користувачу, тестується робота розробленого транслятора.

Оформлення. Студент зобов'язаний оформити пояснювальну записку і графічний матеріал згідно з вимогами до оформлення технічної документації, відповідно до діючих стандартів.

Заключний етап. На цьому етапі проводиться перевірка і захист курсових робіт. Студент зобов'язаний представити керівникові остаточний вихідний код транслятора і оформлену пояснювальну записку до курсового проєкту не пізніше, ніж за 5 робочих днів до захисту.

Керівник перевіряє проєкт і дає вказівки про виправлення або доповнення, після яких оцінює програмний проєкт і пояснювальну записку. На цьому курсовий проєкт вважається закінченим і може бути представленим до захисту.

Після закінчення роботи над курсовим проєктом і допуском до захисту **студент має завантажити** у ВНС у відповідні завдання:

- пояснювальну записку з додатками в форматі pdf;
- вихідні код програми (програмного проєкту) – в форматі архіву (rar, zip).

Захист проєкту. В процесі захисту роботи студент демонструє роботу транслятора на комп'ютері і дає пояснення та відповіді на поставлені питання членами комісії. Захист роботи здійснюється студентом тільки один раз. При отриманні оцінки "незадовільно" студент отримує нове завдання на курсовий проєкт, який він зможе захистити перед комісією.

СТРУКТУРА ТА ОБСЯГ КУРСОВОГО ПРОЄКТУ

Обсяг пояснювальної записки до курсового проєкту повинен становити не менше 30 сторінок друкованого тексту (без врахування додатків).

Рекомендується такий склад пояснювальної записки до курсового проєкту:

ТИТУЛЬНА СТОРІНКА

ЗАВДАННЯ НА КУРСОВЕ ПРОЄКТУВАННЯ

АНОТАЦІЯ

ЗМІСТ

ВСТУП

1. ОГЛЯД МЕТОДІВ ТА СПОСОБІВ ПРОЄКТУВАННЯ ТРАНСЛЯТОРІВ

2. ФОРМАЛЬНИЙ ОПИС ВХІДНОЇ МОВИ ПРОГРАМУВАННЯ

2.1. Деталізований опис вхідної мови в термінах розширеної нотації Бекуса-Наура
2.2. Опис термінальних символів та ключових слів

3. РОЗРОБКА ТРАНСЛЯТОРА З ВХІДНОЇ МОВИ ПРОГРАМУВАННЯ

3.1. Вибір технології програмування
3.2. Проектування таблиць транслятора та вибір структур даних
3.3. Розробка лексичного аналізатора
  3.3.1. Розробка блок-схеми алгоритму
  3.3.2. Опис програми реалізації лексичного аналізатора
3.4. Розробка синтаксичного та семантичного аналізатора
  3.4.1. Розробка дерев граматичного розбору
  3.4.2. Розробка блок-схеми алгоритму
  3.4.3. Опис програми реалізації синтаксичного та семантичного аналізатора
3.5. Розробка генератора коду
  3.5.1. Розробка блок-схеми алгоритму
  3.5.2. Опис програми реалізації генератора коду

4. НАЛАГОДЖЕННЯ ТА ТЕСТУВАННЯ РОЗРОБЛЕНОГО ТРАНСЛЯТОРА

4.1. Опис інтерфейсу та інструкції користувачу
4.2. Виявлення лексичних і синтаксичних помилок
4.3. Перевірка роботи транслятора за допомогою тестових задач

ВИСНОВКИ

СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

ДОДАТКИ

А. Таблиці лексем для тестових прикладів
Б. С код (або код на асемблері), отриманий на виході транслятора для тестових прикладів;
Б. Документований текст програмних модулів (лістинги)

Обов'язковою складовою курсового проєкту є **наявність графічної частини** – 1 лист формату А2 чи А3 з штампом (можливі варіанти графічної частини – дерево граматичного розбору, структура програмного забезпечення, діаграма класів, діаграма послідовності дій користувача, блок-схеми алгоритму роботи програми чи її окремих частин, тощо).

У **ЗАВДАННІ** вказується перелік конкретних вхідних даних. Студент отримує у керівника курсового проєктування індивідуальне завдання.

В **АНОТАЦІЇ** (1 сторінка) викладаються короткі відомості про курсовий проєкт.

У **ЗМІСТІ** вказуються назви структурних частин курсового проєкту та номери сторінок, де вони починаються. ЗАВДАННЯ, ВСТУП, ВИСНОВКИ, СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ і ДОДАТКИ не нумеруються! АНОТАЦІЯ у ЗМІСТ не включається!

У **ВСТУПІ** (1 сторінка) здійснюють опис проблеми у загальному вигляді, визначають її актуальність, формулюють мету курсового проєкту.

У **1-му розділі** (до 5 сторінок) має бути теоретична частина. Цей розділ виконується із використанням літературних джерел і повинен містити опис методів та способів вирішення задачі. Необхідно здійснити порівняння описаних методів та способів.

У **2-му розділі** (до 5 сторінок) описується тип граматики вхідної мови програмування згідно завдання. При цьому визначається алфавіт мови та набір зарезервованих слів. Рекомендується опис вхідної мови подавати в термінах розширеної нотації Бекуса-Наура або за допомогою синтаксичних діаграм.

У **3-му розділі** (не менше 15 сторінок):

- здійснюється вибір технології програмування;
- розробляється структура постійних таблиць для зберігання алфавіту мови, зарезервованих слів, знаків операцій, роздільників тощо, а також здійснюється вибір структур даних і алгоритму пошуку елементів у впорядкованих таблицях;
- розробляється структура перемінних таблиць для зберігання ідентифікаторів і констант та алгоритми пошуку/вставки лексем і їх атрибутів;
- проектується лексичний аналізатор на основі детермінованих кінцевих автоматів з представленням блок-схеми алгоритму та описом програми його реалізації;
- проектується синтаксичний аналізатор на основі дерева граматичного розбору з представленням блок-схеми алгоритму та описом програми його реалізації;
- проектується генератор коду з представленням блок-схеми алгоритму та описом програми його реалізації.

Програма повинна бути написана та реалізована на мові програмування C/C++. Текст програми повинен бути документований і написаний згідно вимог структурного програмування. Кожний програмний модуль на початку повинний містити інформацію про тему курсового проєкту, прізвище автора та дату створення. Кожна підпрограма також мусить буди документованою. Опис програми проводиться в такій послідовності: спершу описується блок-схема алгоритму, а пізніше програмна реалізація.

У **4-му розділі** в підрозділі інструкції користувачу наводять детальну послідовність дій для запуску транслятора та описують всі можливі режими роботи програми. Для ілюстрації режимів роботи програми необхідно навести зображення екранних форм, вікон, меню, блоків діалогу, форм документів.

Технологія налагодження програми – це послідовність дій та засоби виявлення, аналізу та виправлення помилок програми. Описуються використані засоби автоматизованого налагодження та оптимізації програми (автономні та вбудовані налагоджувачі, профайлери). Проводиться аналіз помилок, допущених в ході програмування задачі, спосіб їх виявлення та усунення.

Для підтвердження працездатності програмного продукту розробляється система тестів та приводяться результати тестування з їх аналізом.

У **ВИСНОВКАХ** (1 сторінка) перераховуються основні результати курсового проєкту, вказуються його позитивні сторони та недоліки, даються рекомендації з практичного застосування розроблених алгоритмів та програм.

ВИМОГИ ДО ОФОРМЛЕННЯ ПОЯСНЮВАЛЬНОЇ ЗАПИСКИ

Пояснювальна записка має бути виконана **шрифтом 14 пт**, міжрядковий **інтервал 1.5 пт**. **Поля** – рекомендовані розміри лівого поля – 25 мм, правого - 15 мм, верхнього і нижнього - по 20 мм. На аркушах, де починаються розділи, зміст, анотації, вступ, висновки, список літератури рекомендується збільшувати розмір верхнього поля до 40 мм.

Текст основної частини пояснювальної записки поділяють на розділи, підрозділи, пункти та підпункти. Заголовки структурних частин пояснювальної записки "АНОТАЦІЯ", "ЗМІСТ", "ВСТУП", "РОЗДІЛИ ОСНОВНОЇ ЧАСТИНИ", "ВИСНОВКИ", "СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ", "ДОДАТКИ" **друкують великими літерами** симетрично до тексту. **Крапку в кінці заголовка не ставлять**. Після номера розділу ставиться крапка. Заголовки підрозділів друкують маленькими літерами (крім першої великої) з абзацного відступу. Розділи повинні бути пронумеровані арабськими цифрами послідовно у всій записці. **АНОТАЦІЯ, ЗМІСТ, ВСТУП, ВИСНОВКИ, СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ не нумеруються!**

Підрозділи нумеруються арабськими цифрами послідовно у всьому розділі. Номер підрозділу повинен містити номер розділу і порядковий номер підрозділу, розділених крапкою. Наприклад: "7.3." – третій підрозділ (параграф) сьомого розділу. Пункти нумеруються арабськими цифрами послідовно у всьому підрозділі. Номер пункту повинен включати у собі номер розділу, підрозділу і пункту, розділених крапками. У кінці номера пункту також ставлять крапку. Наприклад; "7.3.4." – четвертий пункт третього підрозділу сьомого розділу.

Нумерація сторінок записки повинна бути наскрізною: перша сторінка – ТИТУЛЬНИЙ АРКУШ, друга – ЗАВДАННЯ, третя – АНОТАЦІЯ і так далі. Номер сторінки проставляють арабськими цифрами у **правому верхньому куті** (крапку після цифри не ставлять).

Рисунки, формули і таблиці нумеруються двома цифрами – номер розділу, крапка, номер у розділі.

Формули в пояснювальній записці (якщо їх більше однієї) **нумерують у межах розділу**. Номер формули складається з номера розділу і порядкового номера формули в розділі, між якими ставлять крапку. Номери формул пишуть біля **правого берега аркуша** на рівні відповідної формули в круглих дужках, наприклад: (3.1) – перша формула третього розділу.

Всі ілюстрації (фотографії, схеми, креслення) у записці повинні називатися однаково – **рисунками**. Рисунки позначаються скорочено: "Рис." Рисунки **нумеруються послідовно у розділі** арабськими цифрами. Номер рисунка повинен містити номер розділу і порядковий номер рисунка, які розділяються крапкою, наприклад: "Рис.1.2." – другий рисунок першого розділу.

Кожна **таблиця** позначається словом "Таблиця" з порядковим номером, що розміщується за словом "Таблиця" з **правого боку**. Таблиця може мати заголовок, який розміщується у наступному рядку після слова "Таблиця". Номер таблиці включає у себе номер розділу і порядковий номер таблиці, що розділені крапкою. Наприклад: "Таблиця 3.2." – друга таблиця третього розділу.

Додатки слід позначати послідовно великими літерами української абетки, за винятком літер Ґ, Є, З, І, Ї, Й, О, Ч, Ь, наприклад, ДОДАТОК А, ДОДАТОК Б і т. д. Один додаток позначається як ДОДАТОК А.

Бібліографічні описи джерел у переліку наводять згідно з ДСТУ ГОСТ 7.1. Приклади*:*

Підручники, монографії, довідники, навчальні посібники:

Мельник А.О. Архітектура комп’ютера: Підручник.- Луцьк: Вид-во обласної друкарні. - 2008.-468с.

Статті:

Мельник А. О. Кіберфізичні системи: проблеми створення та напрями розвитку / Мельник А. О. // Вісник НУ "Львівська політехніка" "Комп’ютерні системи та мережі". – 2014. – № 806. – С. 154–161.

Нормативно-технічні документи:

ДСТУ 3582-97. Інформація та документація. Скорочення слів в українській мові в бібліографічному описі. Загальні вимоги та правила.

Інформаційний лист (datasheet):

Інформаційний лист STM32H743AI [Електронний ресурс] – Режим доступу до ресурсу: https://www.st.com/resource/en/datasheet/stm32h743ai.pdf.

Електронні ресурси:

Системне програмування (курсовий проект) [Електронний ресурс] – Режим доступу до ресурсу: https://vns.lpnu.ua/course/view.php?id=11685.

КРИТЕРІЇ ОЦІНЮВАННЯ КУРСОВОГО ПРОЄКТУ

Курсовий проєкт студента оцінюється за шкалою ECTS (100-бальною шкалою). Найбільш вагомим фактором при оцінювані курсового проєкту є самостійність його виконання студентом.

Оцінювання знань студентів проводиться відповідно до робочого навчального плану у вигляді семестрового контролю, який проводиться в кінці семестру і включає в себе три складові:

- якість оформлення і повнота змісту пояснювальної записки – відповідність оформлення пояснювальної записки до курсового проєкту до вимог нормативних документів, вимог університету та даних методичних вказівок; змістовне наповнення пояснювальної записки. Оцінюється у 30 балів.
- якість і функціональність програмного коду – стиль написання коду, робота у відповідності до завдання, наявність коментарів, зручність інтерфейсу користувача. Оцінюється у 30 балів.
- захист проєкту – повнота та правильність відповідей на запитання членів комісії, стиль викладання матеріалу презентації. Оцінюється у 40 балів.

Захист проекту є обов’язковим видом контролю і проводиться в усній формі перед комісією.

Фактична кількість балів за кожним із параметрів курсового проєкту визначається шляхом знаходження відсоткової частки від максимальної кількості балів:

- 100%, якщо студент вміє творчо мислити, узагальнювати вивчений матеріал, використовує для цього теоретичні положення і конкретні параметри, чітко, вичерпно, глибоко розуміє і знає матеріал і вміє їм користуватися в межах вимог програми курсу, вміє вести співбесіду з членами комісії, при відповідях на навідні і додаткові питання самостійно виправляє окремі допущені неточності, якісно й у повному обсязі виконав розрахункові і графічні роботи, що входять у курсовий проєкт.
- 80%, коли за перерахованими показниками є окремі несуттєві недоліки, наприклад, чіткість і повнота звітів не цілком достатні, маються дрібні недоліки в розрахунках і конструкторській документації, деяка недбалість оформлення тощо;
- 50%, при наявності істотних недоліків за перерахованими показниками, що студент самостійно зміг виправити;
- 30%, при наявності істотних недоліків за перерахованими показниками, що студент самостійно виправити не міг, навіть якщо окремі відповіді були правильними;
- 0%, якщо студент виявив нерозуміння і незнання основного змісту курсового проєкту, допустив грубі помилки в пояснювальній записці, розрахунках, у креслярській документації, не зміг правильно відповісти на питання комісії.

За підсумковою оцінкою визначається оцінка за шкалою ECTS та традиційною національною шкалою відповідно до положення університету.

СПИСОК ЛІТЕРАТУРИ ДЛЯ ВИКОНАННЯ КУРСОВОГО ПРОЄКТУ

1. Основи проектування трансляторів: Конспект лекцій : [Електронний ресурс] : навч. посіб. для студ. спеціальності 123 – «Комп'ютерна інженерія» / О. І. Марченко ; КПІ ім. Ігоря Сікорського. – Київ: КПІ ім. Ігоря Сікорського, 2021. – 108 с.
2. Формальні мови, граматики та автомати: Навчальний посібник / Гавриленко С.Ю. – Харків: НТУ «ХПІ», 2021. – 133 с.
3. Сопронюк Т.М. Системне програмування. Частина І. Елементи теорії формальних мов: Навчальний посібник у двох частинах. – Чернівці: ЧНУ, 2008. – 84 с.
4. Сопронюк Т.М. Системне програмування. Частина ІІ. Елементи теорії компіляції: Навчальний посібник у двох частинах. – Чернівці: ЧНУ, 2008. – 84 с.
5. Alfred V. Aho, Monica S. Lam, Ravi Seth, Jeffrey D. Ullma. Compilers, principles, techniques, and tools, Second Edition, New York, 2007. – 1038 с.
6. Системне програмування (курсовий проект) [Електронний ресурс] – Режим доступу до ресурсу: https://vns.lpnu.ua/course/view.php?id=11685.
7. MIT OpenCourseWare. Computer Language Engineering [Електронний ресурс] – Режим доступу до ресурсу: https://ocw.mit.edu/courses/6-035-computer-language-engineering-spring-2010.
8. Рисований О.М. Системне програмування: підручник для студентів напрямку "Комп'ютерна інженерія" вищих навчальних закладів в 2-х томах. Том 1. – Видання четверте: виправлено та доповнено – Х.: "Слово", 2015. – 576 с.
9. Рисований О.М. Системне програмування: підручник для студентів напрямку "Комп'ютерна інженерія" вищих навчальних закладів в 2-х томах. Том 2. – Видання четверте: виправлено та доповнено – Х.: "Слово", 2015. – 378 с.