

**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ**

О.В. Кудін, А.Г. Кривохата

**ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СКЛАДНИХ
ОБ'ЄКТІВ І СИСТЕМ**

Методичні рекомендації до виконання лабораторних робіт
для здобувачів ступеня вищої освіти бакалавра
спеціальності «Інженерія програмного забезпечення»
освітньо-професійної програми «Програмна інженерія»

Затверджено
вченою радою ЗНУ
Протокол № 7 від 25.02.2020

Запоріжжя
2020

УДК: 004.896(075.8)
К887

Кудін О.В., Кривохата А.Г. Основи автоматизованого проектування складних об'єктів і систем: методичні рекомендації до виконання лабораторних робіт для здобувачів ступеня вищої освіти бакалавра спеціальності «Інженерія програмного забезпечення» освітньо-професійної програми «Програмна інженерія». Запоріжжя : ЗНУ, 2020. 53 с.

У методичних рекомендаціях подано зміст лабораторних робіт з курсу «Основи автоматизованого проектування складних об'єктів і систем». Основна увага приділяється формуванню навичок роботи з системами автоматизованого проектування, набуттю знань про технічне та математичне забезпечення САПР, а також засвоєнню методів побудови математичних моделей.

До кожної лабораторної роботи підібрано короткий теоретичний матеріал, надано детальні рекомендації з виконання завдання, запитання для самоперевірки. У глосарії наведено тлумачення основних термінів, а в україно-англійському словнику подано терміни англійською мовою.

Для здобувачів ступеня вищої освіти бакалавра спеціальності «Інженерія програмного забезпечення» освітньо-професійної програми «Програмна інженерія».

Рецензент:

С.М. Гребенюк, доктор технічних наук, доцент, завідувач кафедри фундаментальної математики

Відповідальний за випуск:

А.О. Лісняк, кандидат фізико-математичних наук, доцент, завідувач кафедри програмної інженерії

Зміст

Вступ.....	4
Загальні рекомендації до виконання лабораторних робіт	6
Лабораторна робота №1. Основи роботи в ANSYS Workbench.....	7
Лабораторна робота №2. Геометричне моделювання у програмі ANSYS	12
Лабораторна робота №3. Визначення фізичних констант у програмі ANSYS...	16
Лабораторна робота №4. Створення сітки скінченних елементів. Процесор Solution.....	19
Лабораторна робота №5. Розрахунок конструкцій при статичних навантаженнях.....	26
Лабораторна робота №6. Геометричне моделювання у програмі FreeCAD.....	33
Лабораторна робота №7. Створення сценаріїв засобами програми FreeCAD....	37
Лабораторна робота №8. Геометричне моделювання засобами бібліотеки PythonOCC.....	41
Україно-англійський словник найбільш вживаних термінів.....	49
Глосарій	50
Рекомендована література	52

Вступ

Швидкий розвиток обчислювальної техніки та її впровадження практично у всі сфери життя сприяв тому, що сьогодні грамотний фахівець у будь-якій галузі повинен добре орієнтуватись у світі комп'ютерів і володіти необхідними програмними засобами. Комп'ютери дозволяють створювати числові моделі різних об'єктів. З їхньою допомогою можна побачити ще не існуючий виріб, отримати його геометричні характеристики, здійснити експерименти з дослідження його фізичних властивостей, внести необхідні зміни, підготувати виробництво і, нарешті, виготовити об'єкт. Інструментом для всього цього служать CAD/CAM/CAE системи. Сучасний програмний інженер має володіти знаннями про системи автоматизованого проектування (CAD – Computer Aids Design), автоматичного виробництва (CAM – Computer Aids Manufacturing) та автоматичного інженерного аналізу (CAE – Computer Aids Engineering). CAE-системи інженерного аналізу (ABAQUS, ANSYS, COSMOS, I-DEAS, NASTRAN, та інші) дозволяють виконати не тільки якісне моделювання систем різної фізичної природи, але й дослідити їхній вплив на зовнішній простір.

Метою вивчення навчальної дисципліни «Основи автоматизованого проектування складних об'єктів і систем» є оволодіння основними поняттями архітектури сучасних систем автоматизованого проектування та систем інженерного аналізу.

Курс «Основи автоматизованого проектування складних об'єктів і систем» є фаховим, професійно-орієнтованим. Його опанування передбачає набуття навичок та вмінь користування сучасними інструментальними засобами розробки, оволодіння основними технологічними методами практичного застосування мовних засобів програмування для розробки систем автоматизованого проектування.

Основними завданнями вивчення дисципліни «Основи автоматизованого проектування складних об'єктів і систем» є: ознайомлення з сучасними поглядами на розробку САПР та їх супроводження, засвоєння головних принципів моделювання складних технічних об'єктів, опанування основних підходів до геометричного моделювання складних об'єктів, набуття знань про порядок роботи у сучасній системі інженерного аналізу ANSYS.

У результаті вивчення дисципліни студенти повинні:

знати:

- основні принципи проектування систем автоматизованого проектування;
- основні підходи до геометричного моделювання складних об'єктів;
- компоненти сучасних систем автоматизованого проектування;
- особливості створення графічних інтерфейсів систем автоматизованого проектування;

- особливості тестування програмних систем автоматизованого проектування;
- основні бібліотеки, що використовуються при розробці систем автоматизованого проектування;
- вміти:
 - користуватись сучасними системами автоматизованого проектування;
 - проектувати структуру компонентів систем автоматизованого проектування;
 - використовувати сучасні бібліотеки та інструментальні засоби для розробки систем автоматизованого проектування;
 - програмувати графічні елементи інтерфейсу користувача;
 - виконувати тестування та оптимізацію роботи програмних систем автоматизованого проектування.

Методичні рекомендації містять 8 лабораторних робіт, у яких потрібно ознайомитися з типовими структурами САПР та отримати чисельні розв'язки задач механіки. Виконання запропонованих робіт дозволить студентові набути практичних навичок роботи з сучасними САЕ-системами інженерного аналізу; пройти основні етапи проектування, ознайомитися з типовими структурами САПР, алгоритмами та програмними засобами, що використовуються при проектуванні складних об'єктів і систем; дослідити питання геометричного моделювання у програмі ANSYS, побудови сітки скінченних елементів, розрахунку конструкцій при статичних навантаженнях, згину прямокутної та круглої пластин, стійкості прямокутної пластини.

Загальні рекомендації до виконання лабораторних робіт

Лабораторні заняття з дисципліни «Основи автоматизованого проектування складних об'єктів і систем» призначені для того, щоб студент під керівництвом викладача, особисто проводив імітаційні експерименти, з метою практичного підтвердження окремих теоретичних положень даної навчальної дисципліни; набував практичних навичок роботи з обчислювальною технікою, методикою експериментальних досліджень за допомогою системи інженерного аналізу ANSYS.

Захист лабораторної роботи відбувається таким чином: студент або демонструє етапи реалізації розрахунків та чисельні результати, або пояснює розв'язання обраних викладачем завдань, відповідає на запитання. Після цього студент надсилає звіт з виконання лабораторної роботи до системи Moodle.

При виконанні кожної роботи необхідно ознайомитись із загальними теоретичними відомостями та матеріалами відповідної лекції (деякі моменти потрібно опрацювати самостійно); виконати розрахункові завдання; оформити звіт з виконання лабораторної роботи, який повинен містити:

- титульний аркуш;
- тему роботи;
- хід роботи відповідно до завдання;
- результати обчислень та аналіз отриманих розв'язків;
- короткі відповіді на контрольні запитання.

Готовий звіт завантажується до СЕЗН Moodle у відведені викладачем терміни. Назва файлу, що завантажується, має вигляд: Прізвище_Ім'я_ІгN (де N – номер лабораторної роботи) та формат текстового документу (приймаються розширення doc, docx або pdf). Якщо розмір файлів перевищує 5Мб, то відповіді розміщуються на зовнішніх ресурсах, а до СЕЗН додається посилання на ці ресурси.

Лабораторна робота №1. Основи роботи в ANSYS Workbench

Мета: вивчити основні елементи графічного інтерфейсу ANSYS.

Теоретичні відомості

ANSYS – це багатоцільовий пакет програм скінченно-елементного аналізу, який розв’язує задачі у різних областях інженерної діяльності (міцність конструкцій, термодинаміка, механіка рідин і газів).

ANSYS є одною з найвідоміших програмних систем у своєму класі та відноситься до CAE-систем (Computer-Aided Engineering), а також забезпечує двосторонній зв’язок з багатьма CAD пакетами.

Математичні моделі, які описують певні фізичні процеси, а також чисельні методи їх розв’язання реалізовано у компонентах (модулях) ANSYS, які називаються вирішувачі. Для розв’язання конкретної фізичної задачі необхідно обрати потрібні модулі програмного комплексу.

В ANSYS реалізовано спеціальні модулі для розв’язання таких типів задач:

- **Fluid Mechanics** – моделювання руху рідин або газів;
- **Structural Mechanics** – моделювання задач динаміки та міцності конструкцій;
- **Electromagnetics** – розрахунок радіоелектронних компонент та пристроїв;
- **Міждисциплінарні задачі** – моделювання задач на перетині різних розділів фізики.

Оболонка ANSYS Workbench забезпечує єдиний інтерфейс для доступу до різних модулів програмного комплексу, керування процесом розрахунку та візуалізацію результатів моделювання.

Інтерфейс користувача оболонки Workbench (рис. 1.1) містить наступні панелі та вікна:

- 1 – вікно **Project Schematic**, в якому відображається схема проекту (набір модулів, які використовуються у проекті);
- 2 – панель інструментів, яка об’єднує загальні властивості проекту, налаштування способу відображення, одиниць виміру тощо;
- 3 – панель **Properties**, яка містить опис властивостей об’єкта у вікні Project Schematic;
- 4 – панель **Toolbox**, яка відображає всі доступні модулі програмного пакета ANSYS.

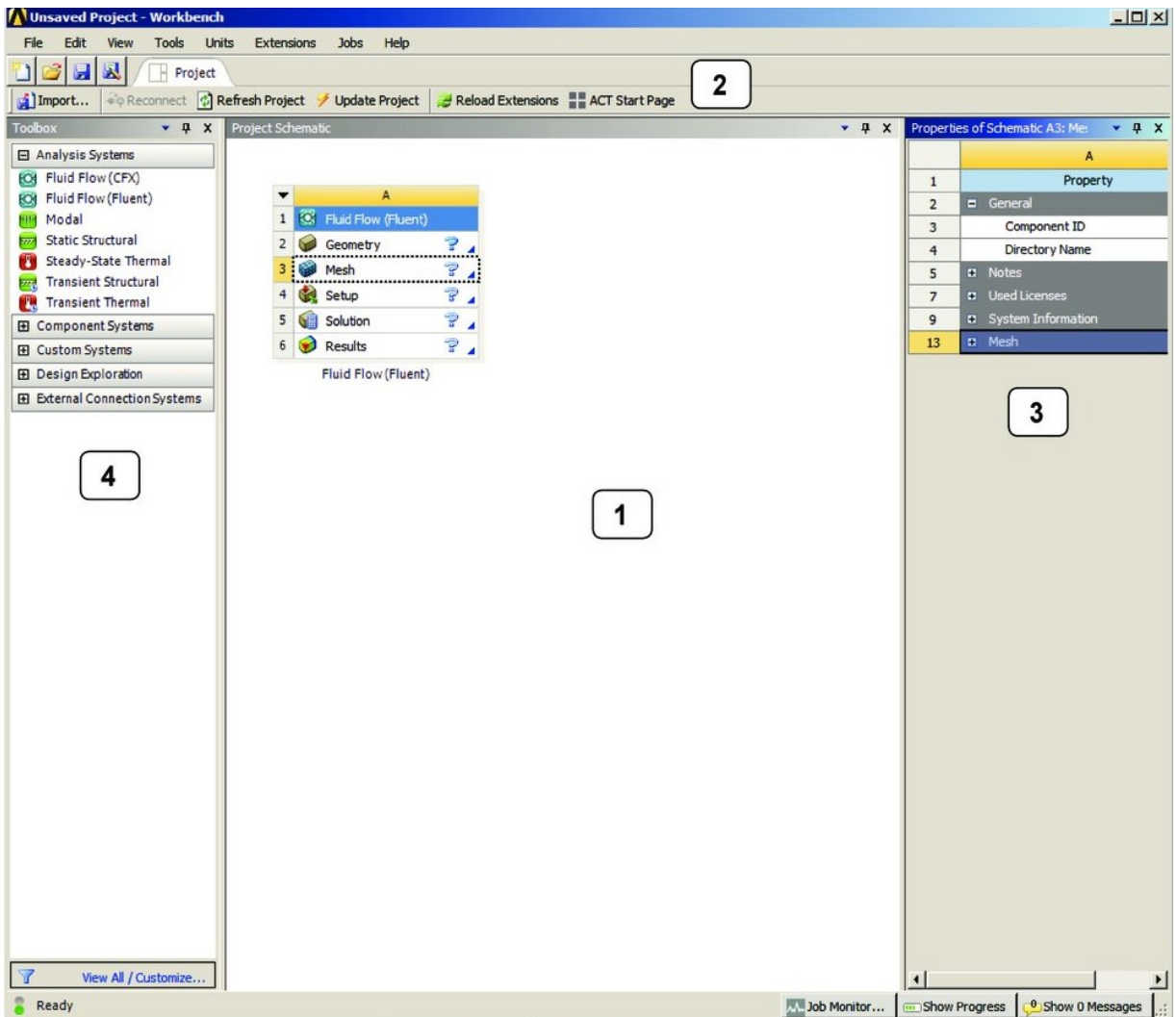


Рисунок 1.1 – Графічний інтерфейс оболонки ANSYS Workbench

Панель **Toolbox** містить п'ять розділів:

- **Analysis Systems** містить готові шаблони для різних типів чисельного аналізу. Зазвичай, кожен шаблон містить ядро у вигляді певного вирішувача, а також допоміжні модулі для підготовки геометричної моделі, скінченно-елементної моделі, визначення властивостей матеріалів конструкції та постпроцесорної обробки результатів обчислень. Наприклад, на рисунку 1.2 зображено шаблон **Static structural** для статичного аналізу конструкцій. Одночасно в одному проекті може знаходитись декілька шаблонів, між якими можна організувати передачу даних (рис. 1.3).

- **Component Systems** включає основні та допоміжні модулі, які можуть використовуватись при розв'язанні задач.
- **Custom Systems** містить готові шаблони для розв'язання міждисциплінарних задач.
- **Design Exploration** дозволяє розв'язувати задачі оптимізації.
- **External Connection Systems** дозволяє інтегрувати у проект зовнішні програмні застосунки.

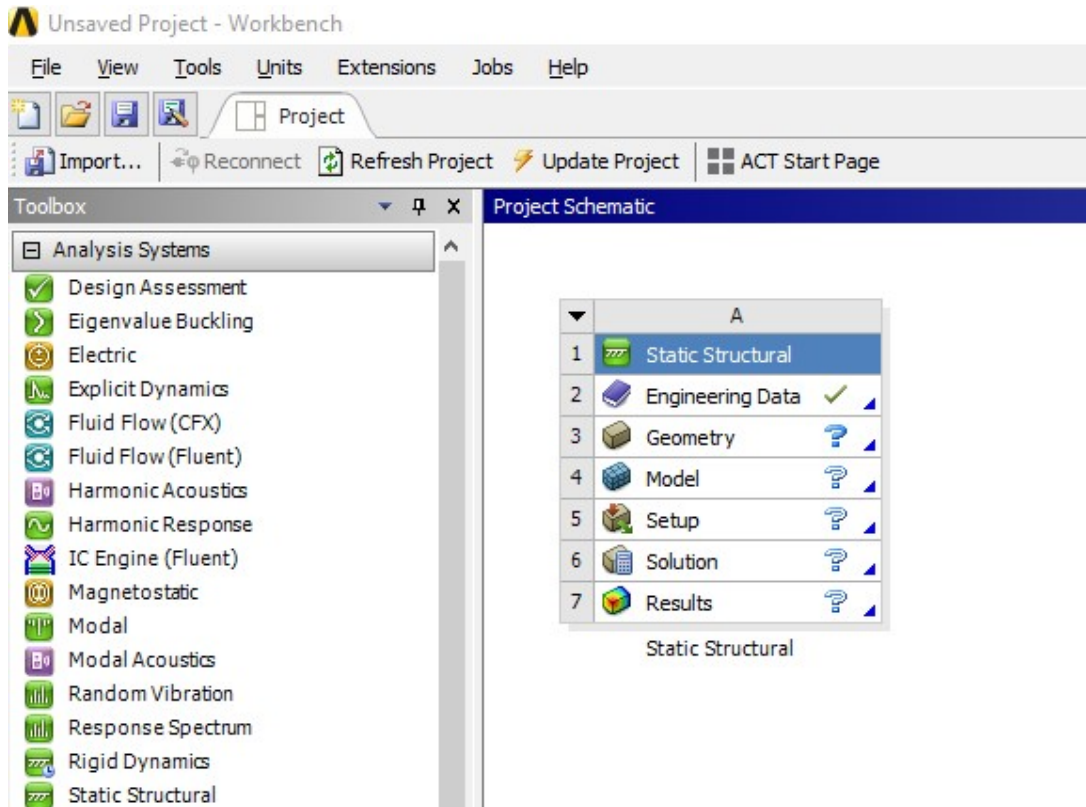


Рисунок 1.2 – Шаблон **Static structural** механічного аналізу

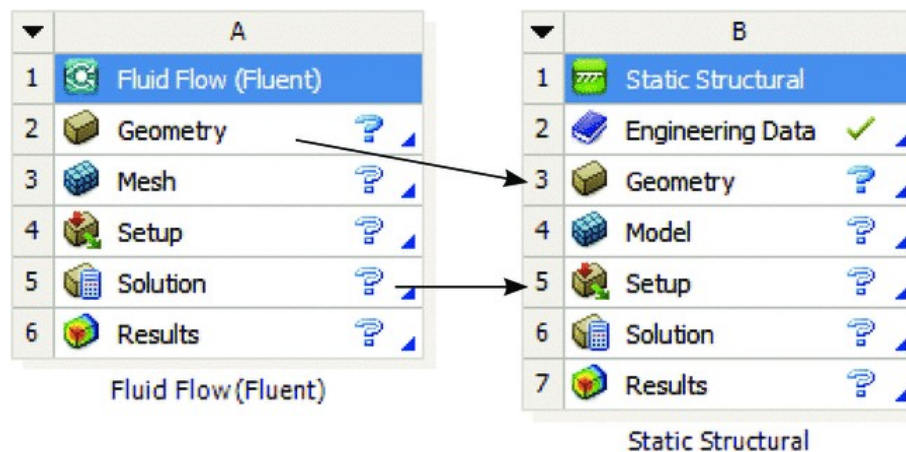


Рисунок 1.3 – Обмін даними між шаблонами

При збереженні проекту у робочому каталозі створюється файл із розширенням **.wbproj**, а також каталог, який містить структуру файлів і каталогів у відповідності до модулів, що використовувались. Для перегляду структури файлів необхідно виконати команду **Files** у меню **View**.

Розв’язання задач за допомогою пакета програм ANSYS відбувається у декілька етапів (рис. 1.4).

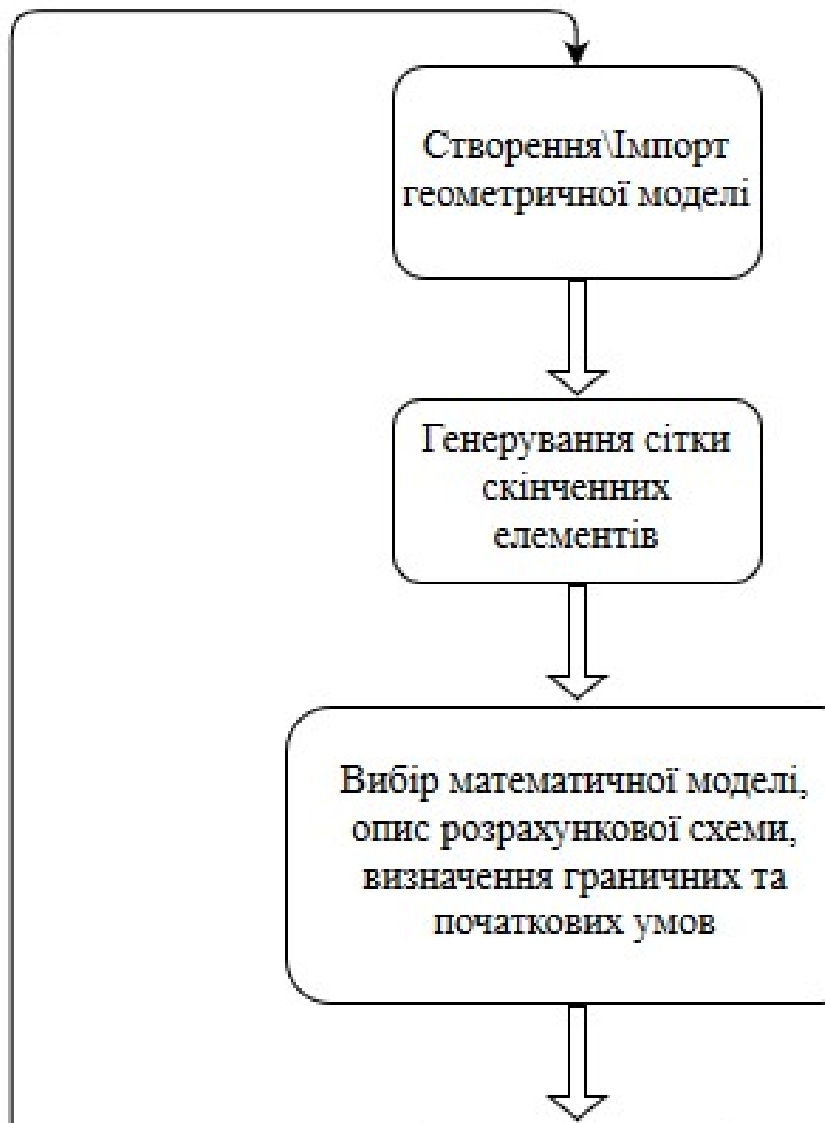


Рисунок 1.4 – Етапи розв’язання задачі в ANSYS

Таким чином, оболонка ANSYS Workbench дозволяє повністю візуалізувати процес розв’язання задачі у відповідності до наведених етапів.

✍ Завдання до лабораторної роботи

1. Запустити програму ANSYS.
2. Ознайомитись з елементами графічного інтерфейсу.
3. Переглянути всі пункти ANSYS Workbench.
4. За результатами виконання лабораторної роботи скласти звіт.

? Контрольні запитання

1. До якого класу автоматизованих систем відноситься програмна система ANSYS?
2. Які основні елементи графічного інтерфейсу ANSYS.
3. Які режими роботи з ANSYS вам відомі?
4. Для якого класу задач призначено шаблон **Static structural**?
5. Які шаблони містяться у розділі **Custom Systems**?
6. Яке призначення оболонки ANSYS Workbench?
7. Які основні етапи розв'язання задач за допомогою систем САПР?

Лабораторна робота №2. Геометричне моделювання у програмі ANSYS

Мета: вивчити основні засоби геометричного моделювання програми ANSYS.

Теоретичні відомості

Розв'язанню будь-якої задачі інженерної механіки передують створення геометричної моделі досліджуваного об'єкта.

Геометрична модель – це математична модель, яка описує геометрію реального об'єкта.

Виділяють два основні принципи побудови геометричної моделі: «знизу-вгору» та «згори-вниз».

При побудові «знизу-вгору» на робочій площині позначаються ключові точки, які потім об'єднуються у лінії, потім – у поверхні чи об'ємні тіла. При побудові «згори-вниз» геометричну модель збирають з готових «примітивів». Такими «примітивами» в ANSYS виступають прямі, лінії, поверхні та об'єми. Часто при моделюванні використовують обидва принципи.

Основними програмними модулями ANSYS, які використовуються для створення геометричних моделей, є:

- Design Modeler;
- SpaceClaim Direct Modeler;
- Mechanical APDL.

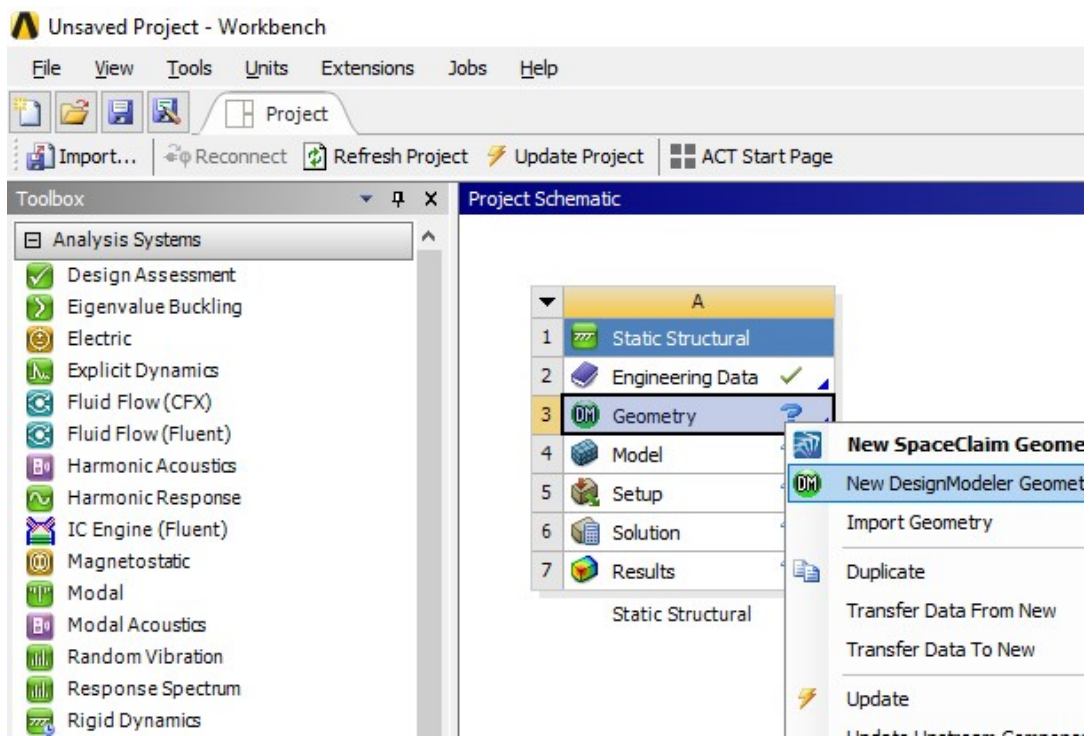


Рисунок 2.1 – Виклик програми Design Modeler

Розглянемо далі більш детально роботу з програмою Design Modeler. Інтерфейс програми має вигляд, який продемонстровано на рисунку 2.2.

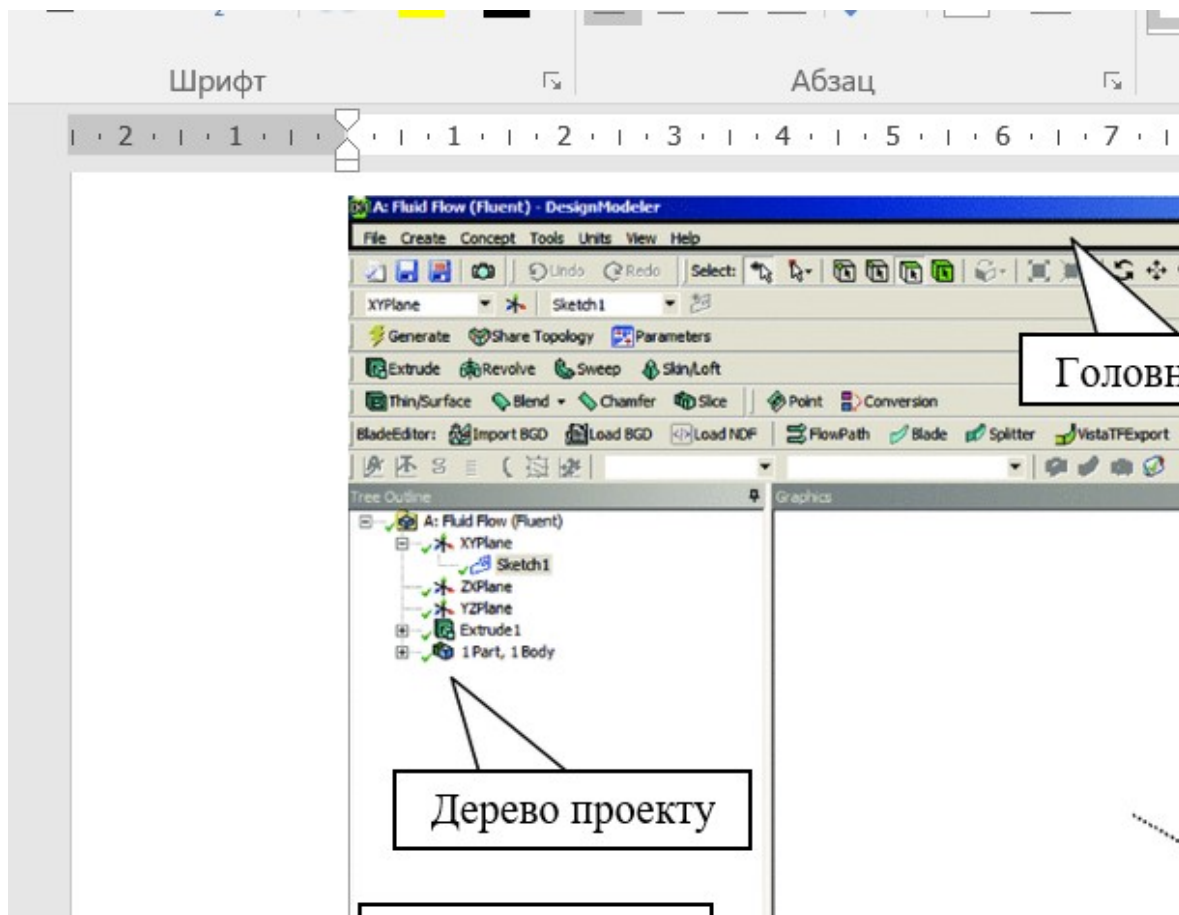


Рисунок 2.2 – Інтерфейс Design Modeler


Модуль Design Modeler має два основних режими роботи: **Modeling** та **Sketching**.

У режимі **Modeling** відображається дерево проекту (**Tree Outline**), яке містить список всіх декартових площин та операцій побудови геометрії, які використовувалися у проекті. Кожна операція над геометрією об'єкта містить певні властивості, які відображаються у вікні **Details View**.

Режим **Sketching** (Розробка ескізу) призначено для створення 2D-ескізу моделі.


Розробка геометричної моделі починається зі створення ескізу, який потім перетворюється у фінальну модель за допомогою функцій геометричного процесора. Таким чином, початковий ескіз може бути перетворений в 1D\2D\3D геометричний об'єкт.


Ескізи в **Design Modeler** створюються у два етапи:

1. Визначення площини для створення ескізу. Всі ескізи прив'язані до певної площини. За замовченням існує три глобальні площини XY, ZX, YZ у глобальній декартовій системі координат. Для створення нової площини слід натиснути кнопку  на панелі інструментів.

В Design Modeler існує шість варіантів створення площини:

- **From Plane;**
- **From Face;**
- **From Centroid;**
- **From Circle/Ellipse;**
- **From Point and Edge;**
- **From Point and Normal;**
- **From Three Points;**
- **From Coordinates.**

Для створення площини можна також застосовувати такі перетворення: операції повороту, переміщення. Для завершення процесу створення площини необхідно натиснути кнопку  **Generate**.

2. Створення ескізу на площині за кнопкою . На кожній площині може бути створена необмежена кількість ескізів. Для роботи з ескізом необхідно виділити його у дереві проекту та перейти у режим **Sketching**. Основні інструменти для роботи з ескізами містяться на вкладках **Draw, Modify, Dimensions, Settings, Constrains**.

Геометричні моделі у Design Modeler можуть бути створені як на основі ескізів, так і з використанням готових 3D примітивів (Primitives), набір яких міститься у меню **Create**.

До створених на основі примітивів 3D-тіл можна застосовувати логічні операції об'єднання, віднімання, перетину, які містяться у меню **Boolean**.

Для створення об'ємних об'єктів з ескізів застосовується операція видавлювання **Extrude**, обертання **Revolve** та витягування за траєкторією **Sweep**.

Завдання до лабораторної роботи

1. Створити геометричну модель прямокутної пластини з такими параметрами: довжина – 0,5 м; ширина – 0,2 м; товщина – 0,01 м.

2. Створити геометричну модель круглої пластини з центральним круглим отвором з такими параметрами: зовнішній радіус – 0,4 м; внутрішній радіус – 0,2 м; товщина – 0,01 м.

3. Створити геометричну модель сфери з центральним отвором (рисунок 2.3). Радіус сфери – 0,5 м, радіус отвору – 0,2 м.

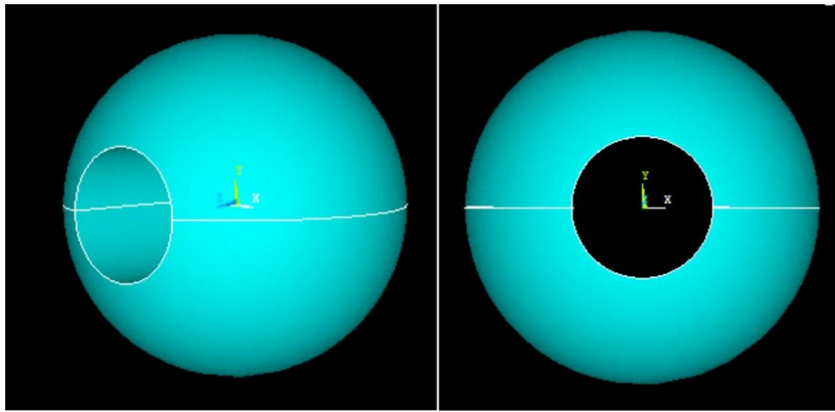


Рисунок 2.3 – Геометрична модель сфери з центральним отвором

4. Побудувати геометричну модель гайки. Зовнішній радіус – 0,5 м; внутрішній радіус – 0,2 м; товщина – 0,2 м (рис.2.3).

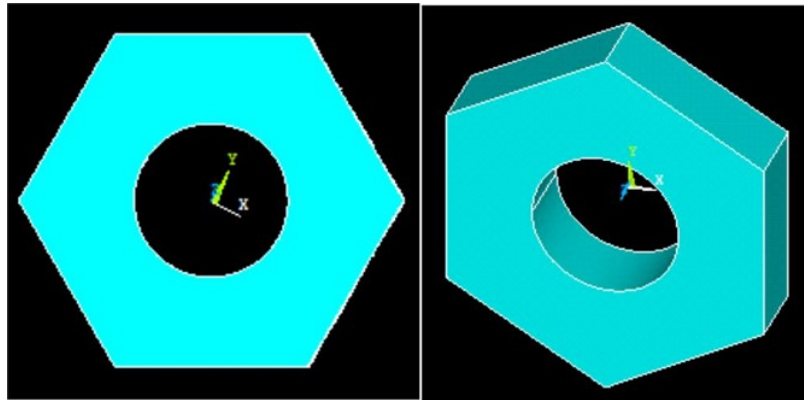


Рисунок 2.3 – Геометрична модель гайки

? Контрольні запитання

1. Які два підходи до моделювання складних геометричних моделей вам відомі?
2. Які засоби геометричного моделювання існують для моделювання поверхонь?
3. Які засоби геометричного моделювання існують для моделювання об'ємних тіл?
4. Які логічні операції з геометричними об'єктами вам відомі?
5. Які операції виконуються з геометричними об'єктами в пункті меню **Extrude**?
6. Яким чином за допомогою меню **Boolean** вирізати з об'ємного тіла деякий отвір?

Лабораторна робота №3. Визначення фізичних констант у програмі ANSYS

Мета: вивчити основні засоби визначення фізичних констант матеріалів програми ANSYS.

Теоретичні відомості

Побудова геометричної моделі задачі – процес досить трудомісткий. Для подальшої роботи з моделлю необхідно визначити матеріали для створених моделей та визначити їх фізичні константи.

Для визначення фізичних параметрів моделі в Ansys Workbench використовується блок **Engineering Data** (рис. 3.1).



Рисунок 3.1 – Визначення фізичних параметрів моделі

Інтерфейс модуля керування матеріалами (рис 3.2) має такі основні вікна:

- **A (Toolbox)** – містить властивості, які можуть бути використані при визначенні матеріалу;
- **B (Outline Filter)** – відображає доступні бази даних матеріалів та їх властивостей;
- **C (Outline Pane)** – відображає структуру обраної бази даних матеріалів;
- **D (Properties Pane)** – відображає властивості обраного елемента на панелі структури даних Outline Pane;
- **E (Table Pane)** – відображає табличні дані для обраного елемента на панелі властивостей Properties Pane;
- **F (Chart Pane)** – відображає діаграму елемента, обраного на панелі властивостей.

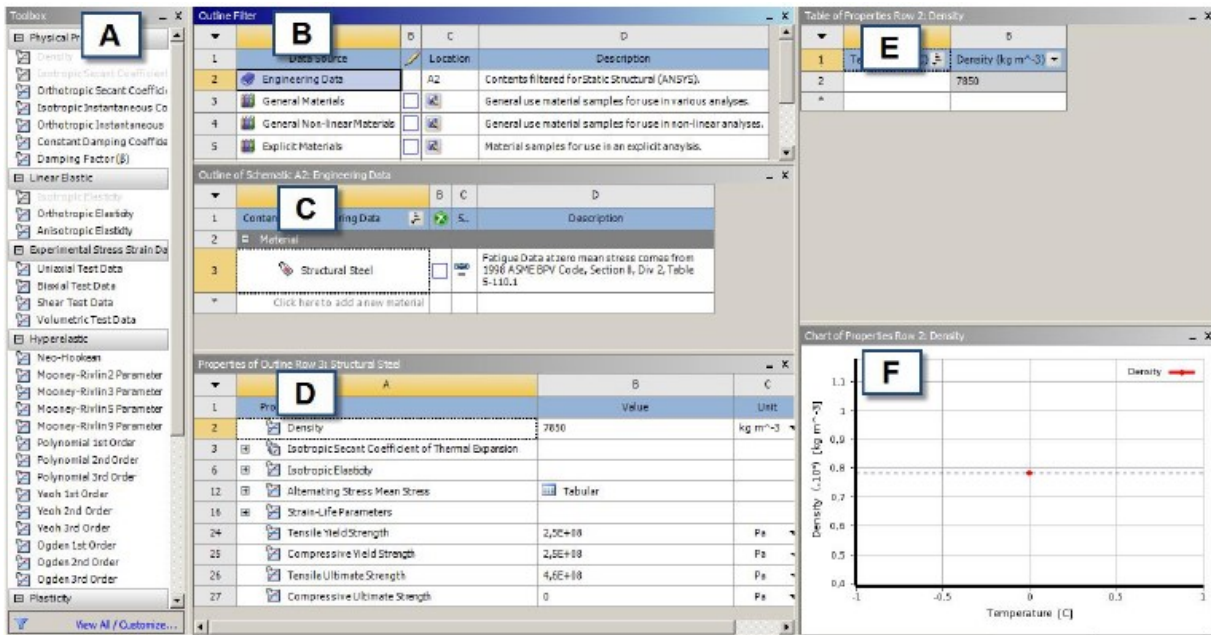


Рисунок 3.2 – Модуль керування матеріалами

Для додання нового матеріалу у шаблон проекту необхідно зайти у базу даних матеріалів (рис. 3.3) та у вікні **Outline of General Materials** з контекстного меню обрати **Add to**

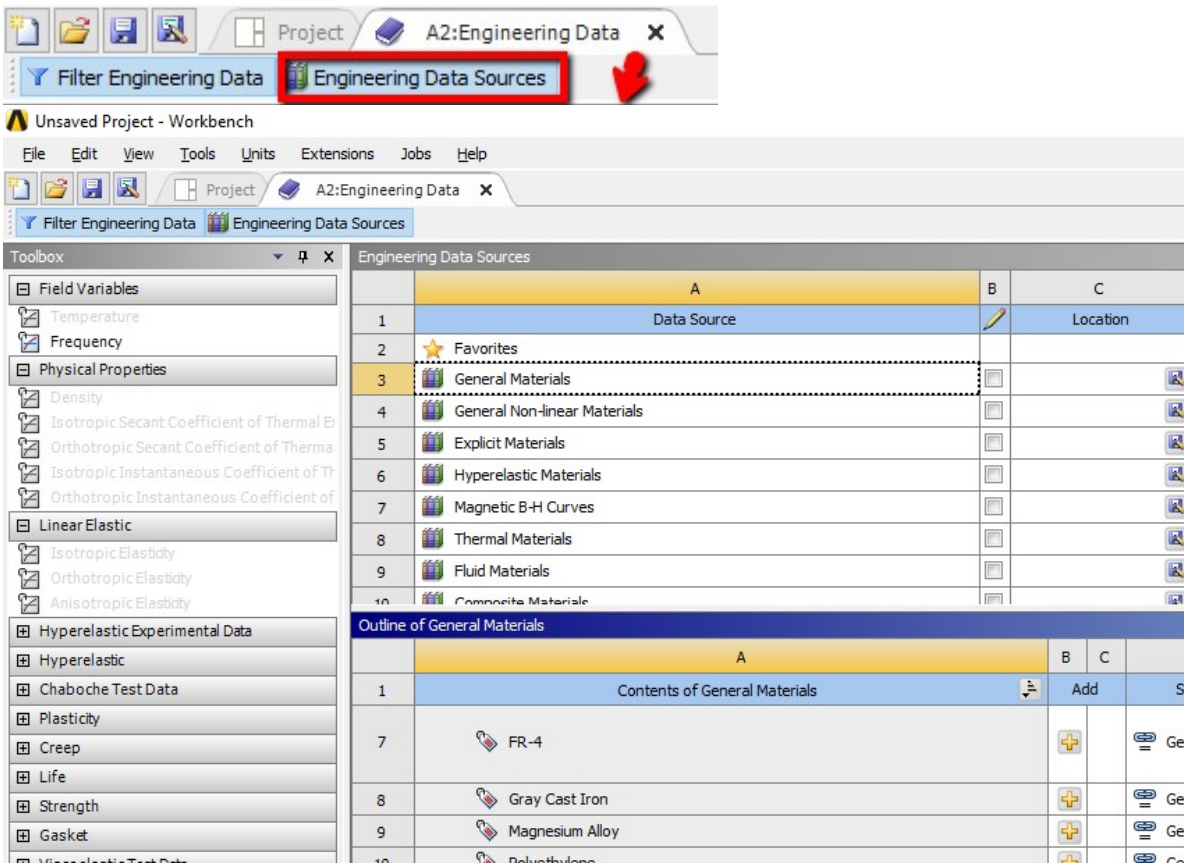


Рисунок 3.3 – База даних матеріалів

Після цього, новий матеріал з'явиться на закладці **Filter Engineering Data** (рис. 3.4) та буде використовуватися у подальших розрахунках.

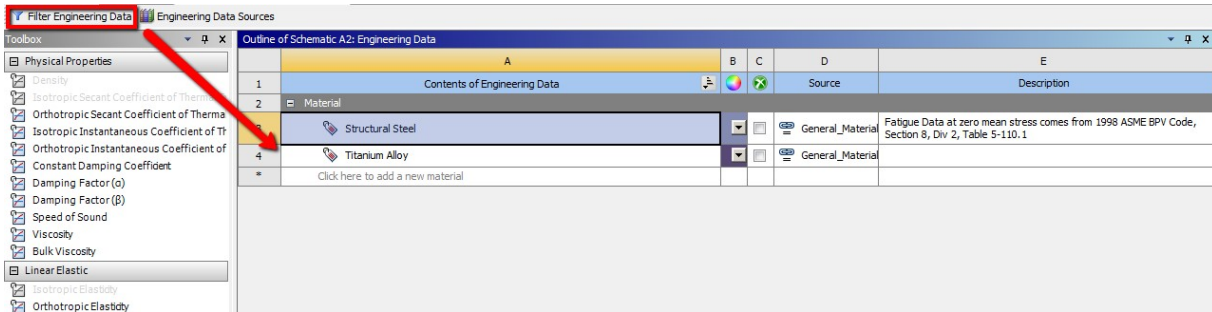


Рисунок 3.4 – Новий матеріал для розрахунків

Завдання до лабораторної роботи

1. Для 1-го завдання 2-ї лабораторної роботи визначити фізичні константи для сталі.
2. Для 2-го завдання 2-ї лабораторної роботи визначити фізичні константи для титана.
3. Для 3-го завдання 2-ї лабораторної роботи визначити фізичні константи для міді.
4. Для 4-го завдання 2-ї лабораторної роботи визначити фізичні константи для алюмінію.

? Контрольні запитання

1. Що таке фізична модель матеріалу в ANSYS?
2. Які типи задач можуть вирішуватись в ANSYS?
3. Який матеріал називається пружним ізотропним матеріалом? Наведіть приклади таких матеріалів.
4. Які константи задають ізотропний пружний матеріал?
5. Який фізичний зміст модулю Юнга та коефіцієнта Пуассона?
6. В яких одиницях вимірюється модуль Юнга?
7. В яких одиницях вимірюється коефіцієнт Пуасона?

Лабораторна робота №4. Створення сітки скінченних елементів. Процесор Solution

Мета: вивчити основи методу скінченних елементів і засоби побудови скінченно-елементної сітки в ANSYS.

Теоретичні відомості

Задачі визначення напружено-деформованого стану конструкцій розв'язуються в ANSYS за допомогою методу скінченних елементів. Метод скінченних елементів – це чисельний метод розв'язання інтегральних і диференціальних рівнянь. Одним із етапів застосування методу скінченних елементів є дискретизація досліджуваного об'єкта. Тобто, представлення геометричної моделі об'єкта у вигляді сітки скінченних елементів (рис. 4.1). При цьому, можливе визначення рівномірної (рис. 4.1, а) або нерівномірної (рис. 4.1, б) сітки.

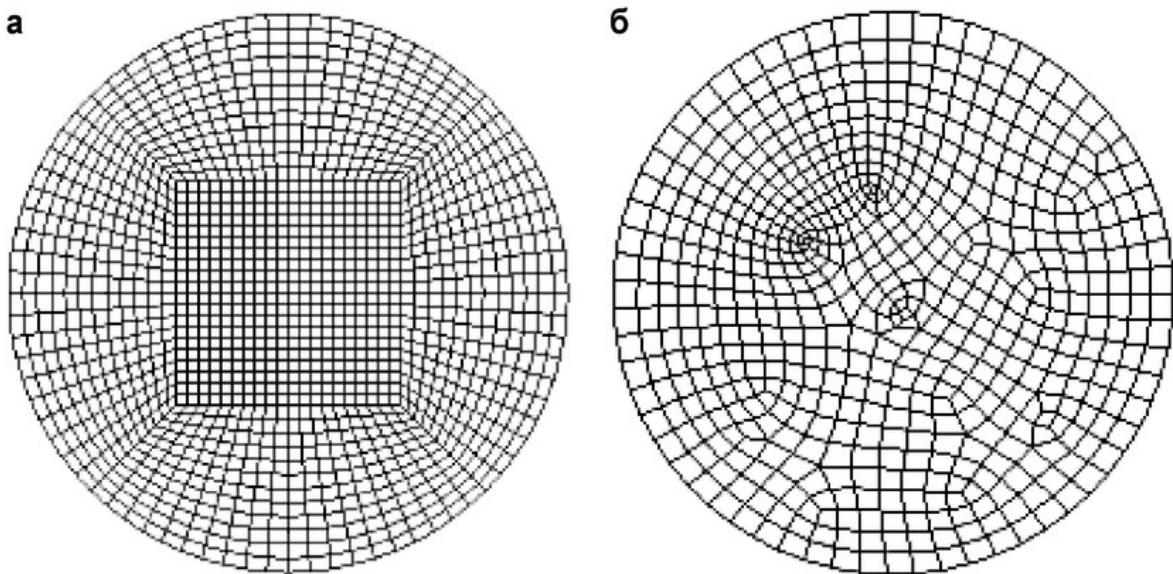


Рисунок 4.1 – Сітка скінченних елементів

В основі скінченно-елементної сітки, що генерується спеціальною програмою-генератором, можуть бути різноманітні 2D\3D елементи: трикутники (рис. 4.2, а); чотирикутники (рис. 4.2, б); гексаедри (рис. 4.2, в); тетраедри (рис. 4.2, г); призми (рис. 4.2, д); піраміди (рис. 4.2, е).

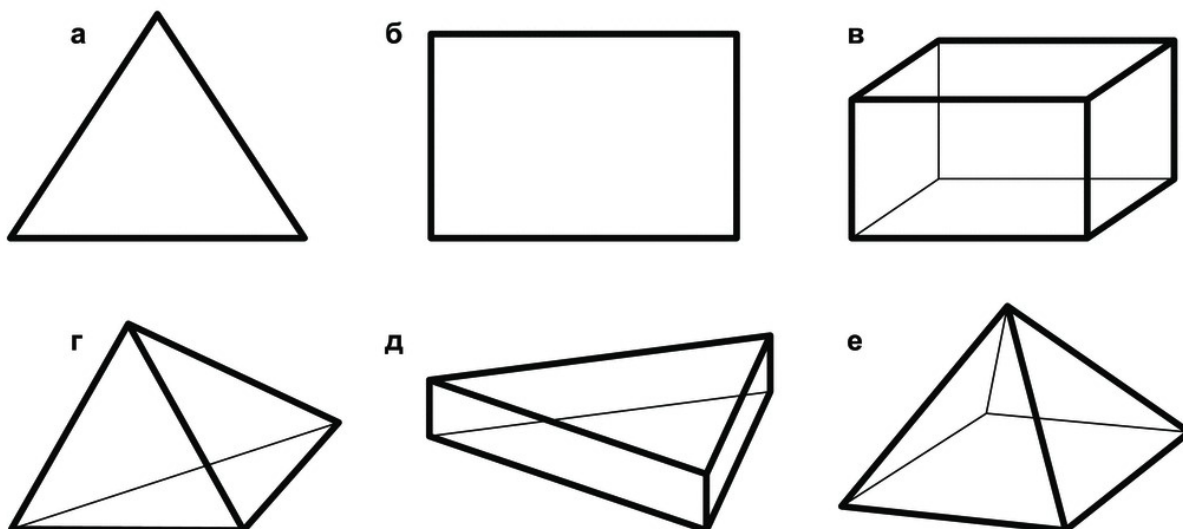


Рис. 4.2 – Елементи скінченно-елементної сітки

Програмна оболонка Ansys Workbench надає можливість роботи з такими генераторами сіток:

- **Ansys Meshing** – універсальний 2D\3D генератор сіток, який дозволяє генерувати сітки на основі гекса-, тетра- та призматичних елементів. Для 2D областей використовуються чотири- та трикутні елементи;
- **Ansys TurboGrid** – генератор, який призначено для автоматизованої побудови сіток на основі гексаєдрів для лопастей машин (гребні гвинти, лопаті турбін тощо);
- **Ansys ICEM CFD** – потужний генератор сіток, який використовується для розширення функціоналу програм Ansys Meshing Design Modeler.

Робота з Ansys Meshing

Препроцесор Ansys Meshing інтегровано в Ansys Workbench. Розглянемо далі основні етапи роботи з програмою.

Для завантаження програми необхідно викликати меню Mesh з Toolbox (рис. 4.3, а) або редагувати рядок Mesh/Model у шаблоні задачі (рис. 4.3, б).

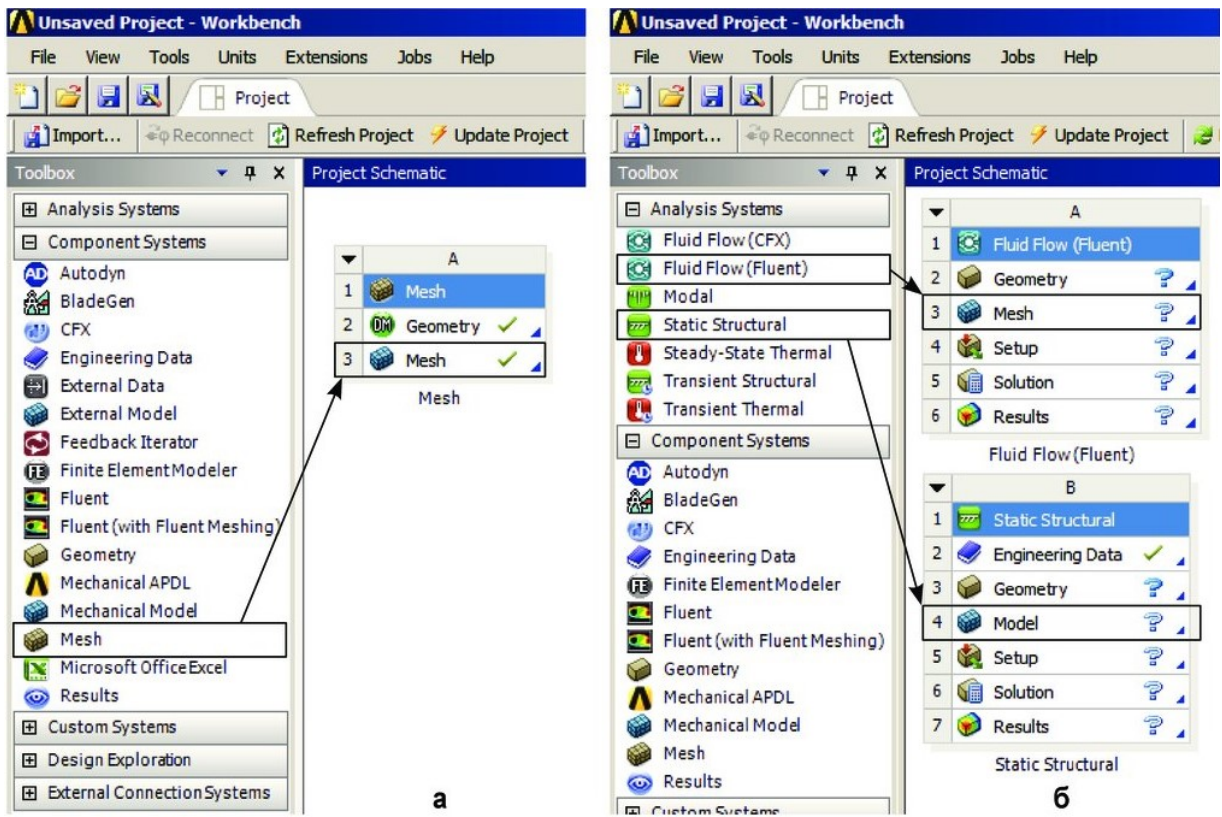


Рисунок 4.3 – Завантаження програми-генератора

Інтерфейс програми містить дерево проектів, графічне вікно, вікно налаштувань та панель інструментів (рис. 4.4).

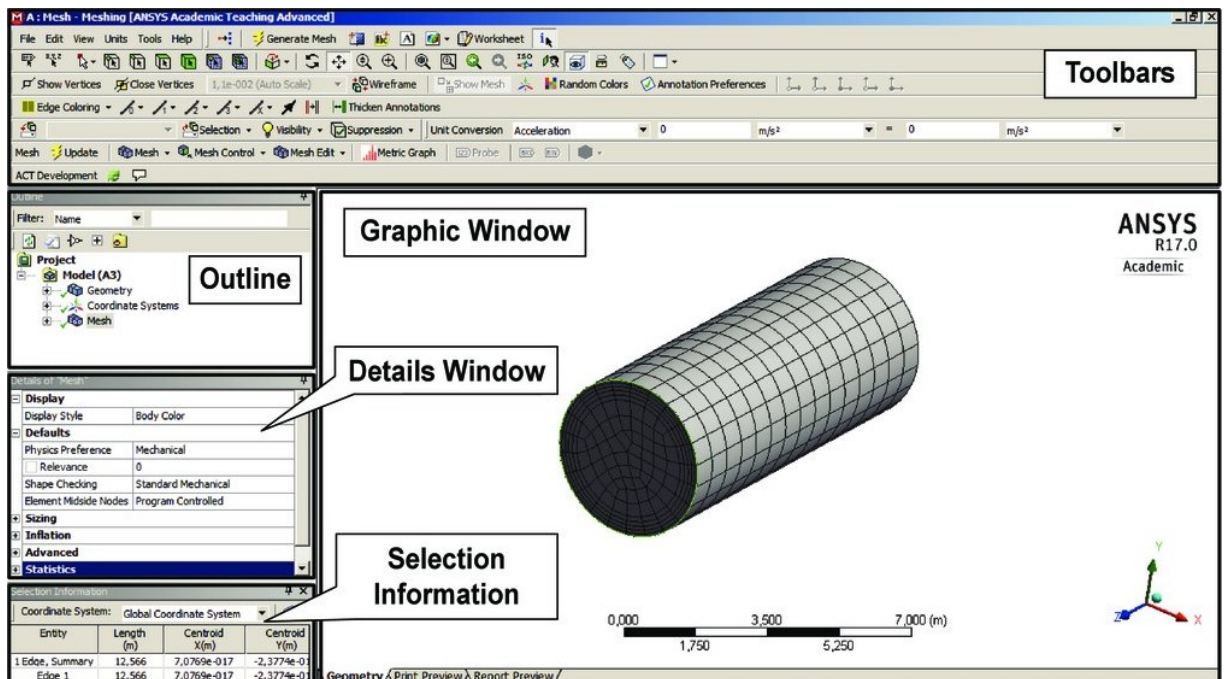


Рисунок 4.4 – Інтерфейс програми Mesh

Перед генерацією сітки, можна попередньо переглянути результат дискретизації за допомогою меню **Preview Surface Mesh** (рис. 4.5)

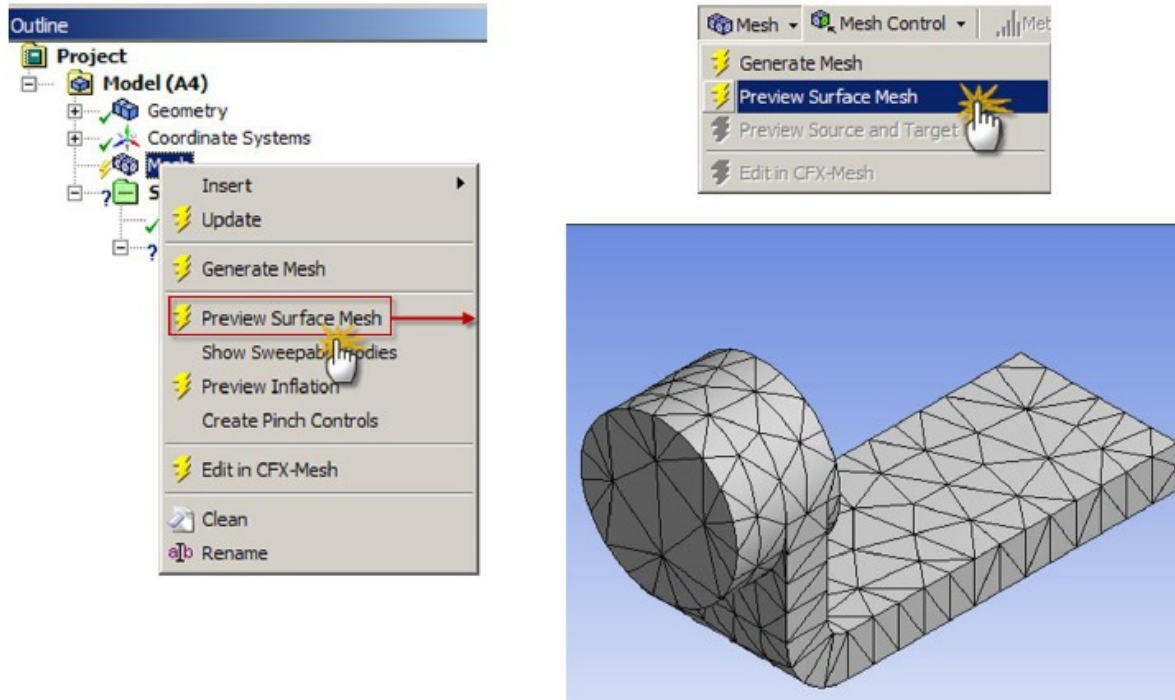


Рисунок 4.5 – Результат дискретизації об’ємного тіла

Налаштування генератора сіток

Вікно **Details of Mesh** містить різноманітні налаштування якості сітки.

Розділ **Defaults** містить налаштування типу фізичної задачі (**Physics Preference**); фактор густини сітки (**Relevance**), який змінюється у діапазоні від -100 до 100 (-100 – рис. 4.6,а; 0 – рис. 4.6,б; 100 – рис. 4.6,в).

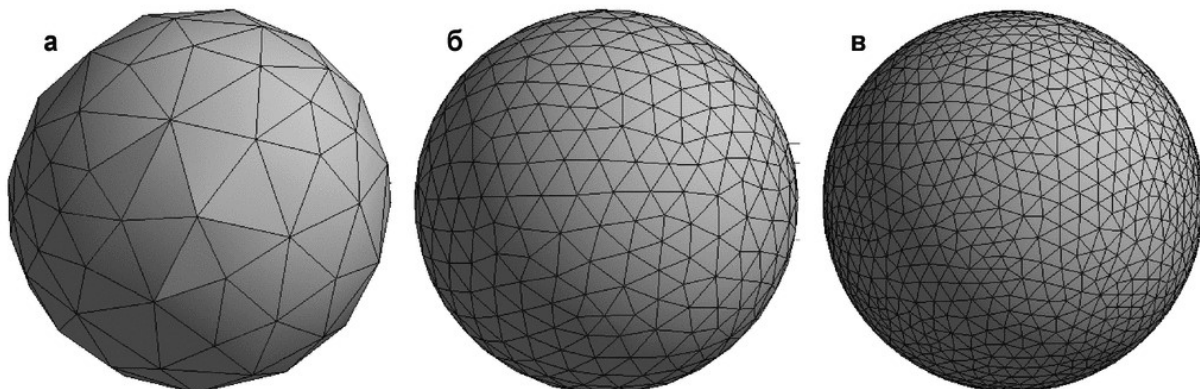


Рисунок 4.6 – Зразки густини сітки

Розділ **Statistics** містить відомості про кількість елементів та відповідних узлів (рис. 4.7)

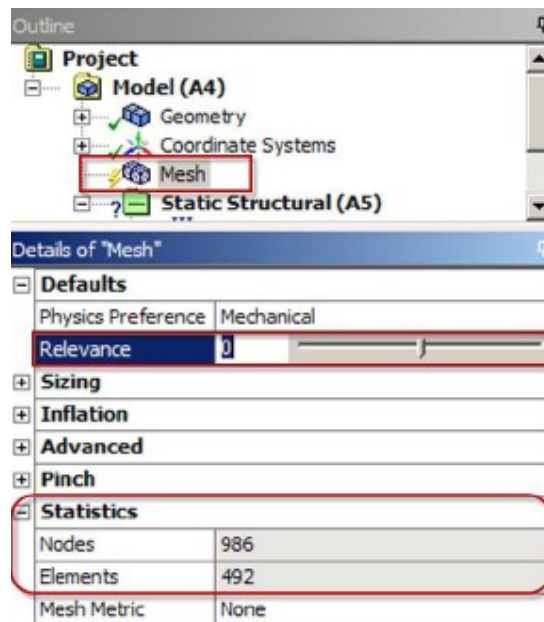


Рисунок 4.7 – Візуалізація статистичної інформації про сітку

Густина сітки впливає на точність подальших розрахунків. При цьому, слід враховувати, що дуже детальна сітка призводить до збільшення часу генерації сітки та розрахунків. Отже, необхідно обрати оптимальний варіант параметрів густини сітки, розміру скінченного елемента, якості сітки тощо. Крім параметра **Relevance**, можна використовувати його середнє значення **Relevance Center** з розділу **Sizing**. Можливі значення цього параметра: **Coarse** (груба сітка), **Medium** (середня сітка), **Fine** (мала сітка).

Співвідношення між параметрами **Relevance** та **Relevance Center** можна побачити на рисунку 4.8.

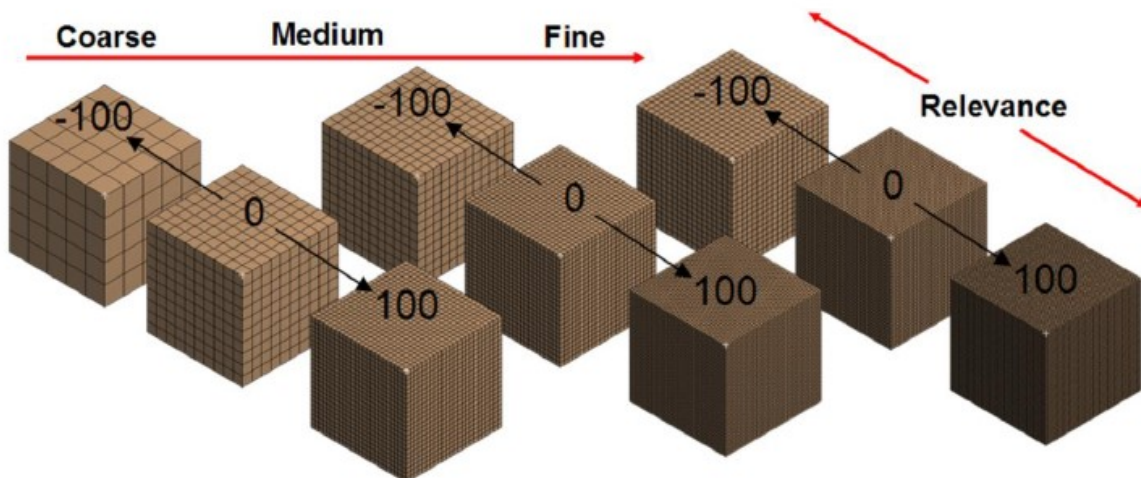


Рисунок 4.8 – Співвідношення між параметрами сітки

Налаштування розміру елемента (**Element Size**) визначає розмір елемента всієї моделі. Ця опція не буде активною, коли використовується

функція додаткових налаштувань розміру елемента (**Use Advanced Size Function**).

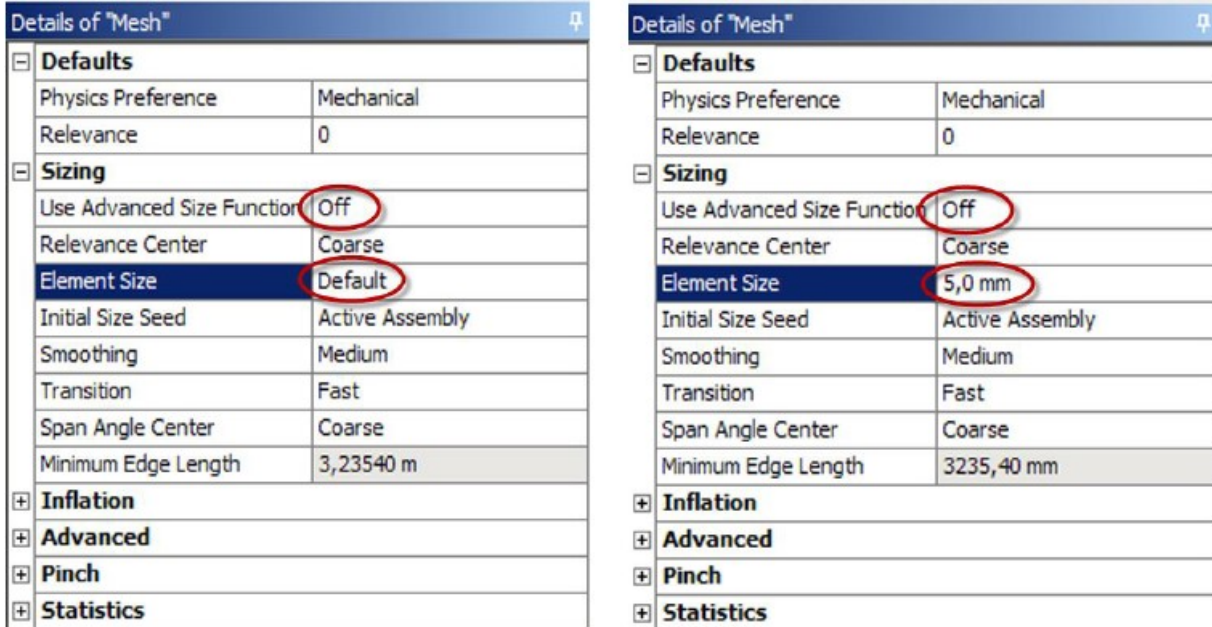


Рисунок 4.9 – Меню додаткових налаштувань розміру елемента

Налаштування вихідного розміру сітки (**Initial Size Seed**) вказує розмір сітки (**Assembly, Part,...**), який використовується на початку генерування сітки.

Налаштування згладжування (**Smoothing**) є активним при вимкненій функції додаткових налаштувань розміру елемента (**Use Advanced Size Function**). Згладжування сітки налаштовується для покращення якості елементів. Значення згладжування (**Low, Medium, High**) керують кількістю ітерацій згладжування.

Методи побудови сіток

Методи побудови сіток доступні через контекстне меню компонента **Mesh** у дереві **Outline** (рис. 4.10).

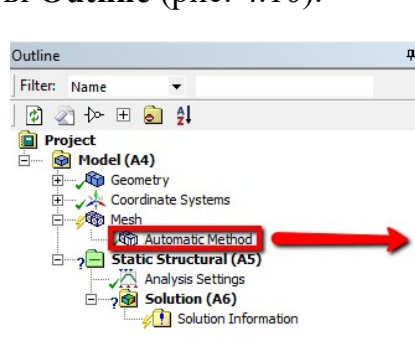


Рисунок 4.10 – Меню вибору методів побудови сітки

Критерії якості сіток

Якісна дискретизація вихідного об'єкта – ключовий фактор точності розв'язання задачі. Розділ **Quality** вікна властивостей (рис. 4.11) побудованої сітки дозволяє розраховувати різноманітні метрики якості і за ними робити висновок про якість побудованої сітки.

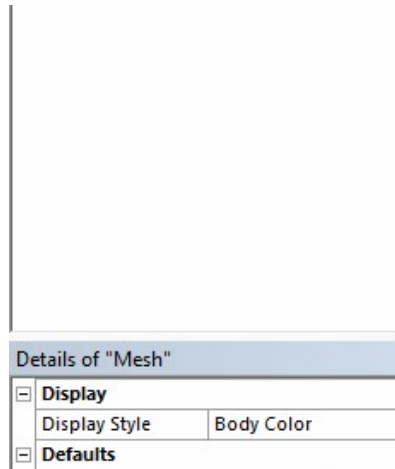


Рисунок 4.11 – Меню вибору метрик якості

Розглянемо деякі основні метрики якості.

- **Element Quality.** Може приймати значення від 0 (низька якість) до 1 (висока якість). Для 2D випадку критерій засновано на відношенні площі елемента до суми квадратів довжин ребер елемента. Для 3D випадку – на відношенні об'єму елемента до квадратного кореня суми квадратів довжин ребер елемента у третьому степені.

- **Aspect Ratio.** Визначає відношення максимальної і мінімальної довжин сторін елемента. Найкращим співвідношенням є Aspect Ratio=1, однак, для багатьох задач можливо більш високе значення.

Завдання до лабораторної роботи

Для кожної з геометричних моделей лабораторної роботи №2 згенерувати сітку скінченних елементів.

? Контрольні запитання

1. Опишіть процес розв'язання задачі в ANSYS.
2. Як відбувається побудова скінченно-елементної сітки в ANSYS?
3. Яким чином задаються зовнішні навантаження та граничні умови?
4. У чому суть процесу моделювання?
5. Як визначається адекватність моделі? Чим вона характеризується?

Лабораторна робота №5. Розрахунок конструкцій при статичних навантаженнях

Мета: опанувати методику розрахунку пластин в ANSYS.

Теоретичні відомості

Під **пластиною** будемо розуміти тіло, яке обмежено двома площинами. Зазвичай, відстань між площинами вважається малою. Найчастіше пластини у конструкціях зазнають згину.

Статичне навантаження – це таке навантаження, значення якого змінюється відносно повільно, так, що можна нехтувати залежністю такого навантаження від часу.

Для розрахунку конструкцій на статичні навантаження необхідно обрати шаблон **Static Structural** (рис. 5.1).

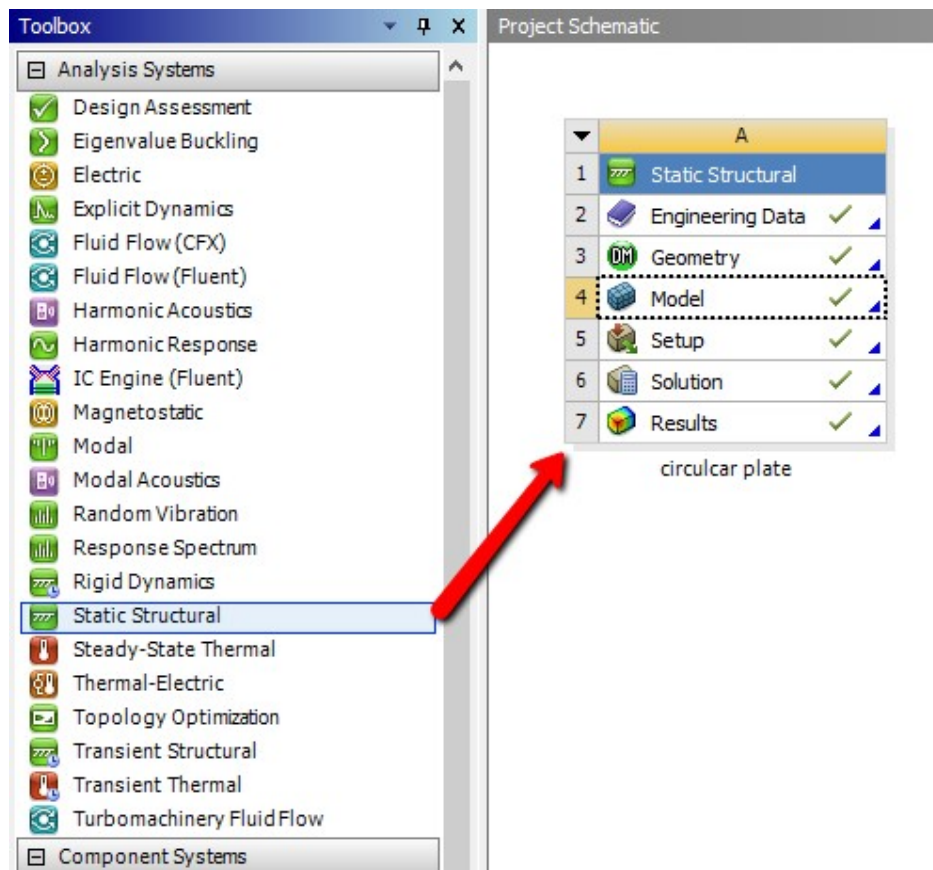


Рисунок 5.1 – Вибір шаблону для розрахунку статичних навантажень

Після послідовного заповнення полів **Engineering Data** (визначення матеріалів конструкції), **Geometry** (побудова геометричної моделі) та **Model** (побудова сітки скінченних елементів) необхідно визначити умови закріплення та навантаження, які діють на конструкцію.

Для визначення умов закріплення та зовнішніх навантажень використовується програмне забезпечення **Ansys Mechanical**, тобто, теж саме, що і для побудови скінченної сітки (рис. 5.2).

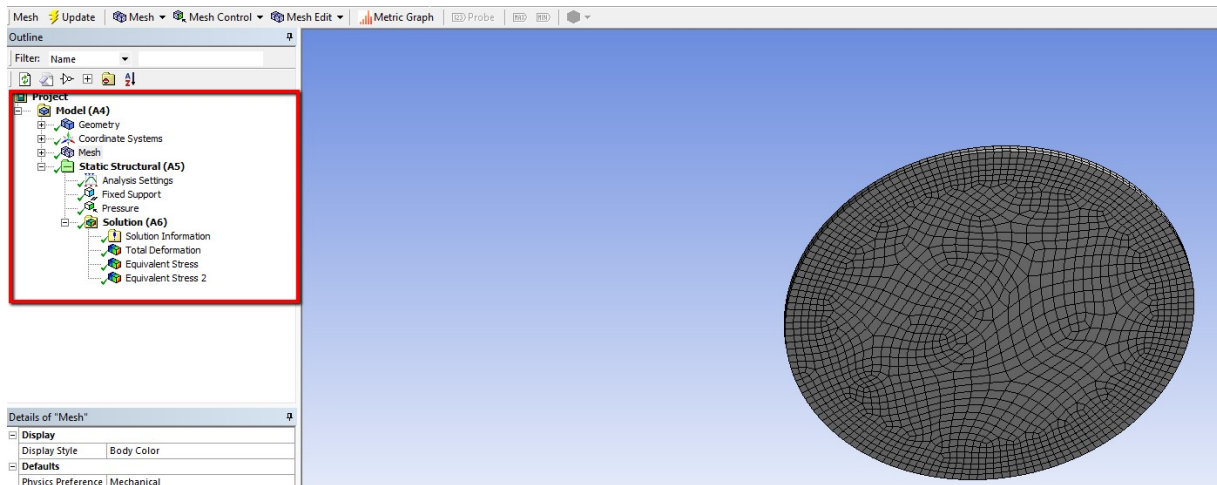


Рисунок 5.2 – Конструкція з сіткою скінченних елементів

Для визначення умов закріплення та зовнішніх навантажень, необхідно у дереві проектів обрати гілку **Static Structural**. Після цього, у контекстному меню та панелі інструментів активуються пункти меню **Loads** для визначення навантажень та **Supports** для визначення способів закріплення (рис. 5.3).

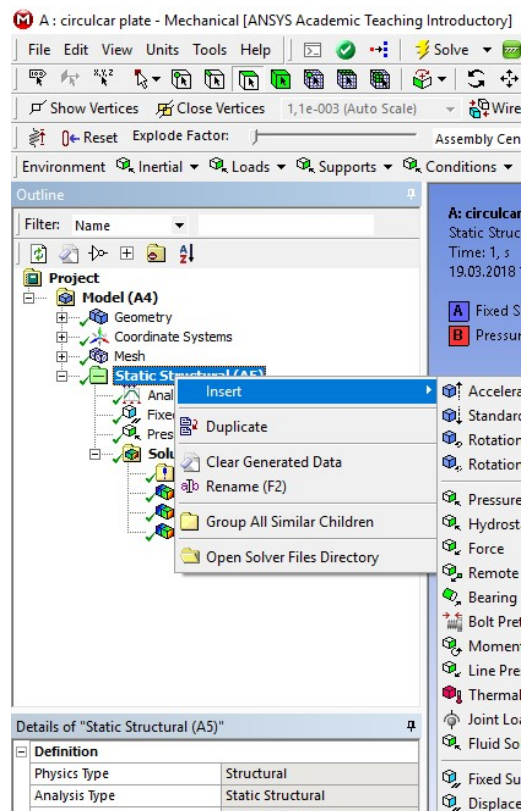


Рисунок 5.3 – Вибір навантажень та способів закріплення конструкції

Наприклад, для розв'язання задачі вигину жорстко закріпленої по контуру круглої пластини під дією поперечного тиску, необхідно виконати такі дії.

1. Якщо у дереві проекту програми **Ansys Mechanical** немає гілки **Static Structural**, обрати тип задачі, що буде розв'язуватись (рис. 5.4).

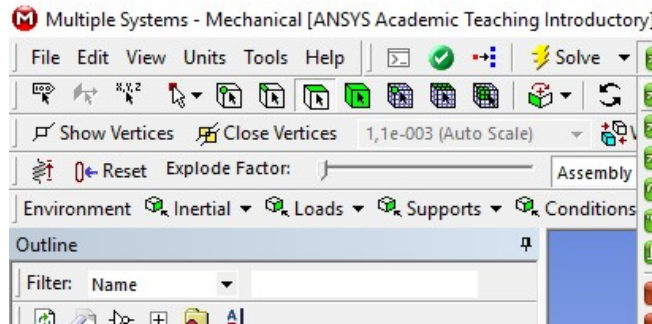


Рисунок 5.4 – Вибір типу задачі

2. Визначити спосіб закріплення. Для цього, слід обрати пункт **Fixed Support** у меню **Supports** та обрати поверхню, до якої слід застосувати умови закріплення (рис. 5.5).

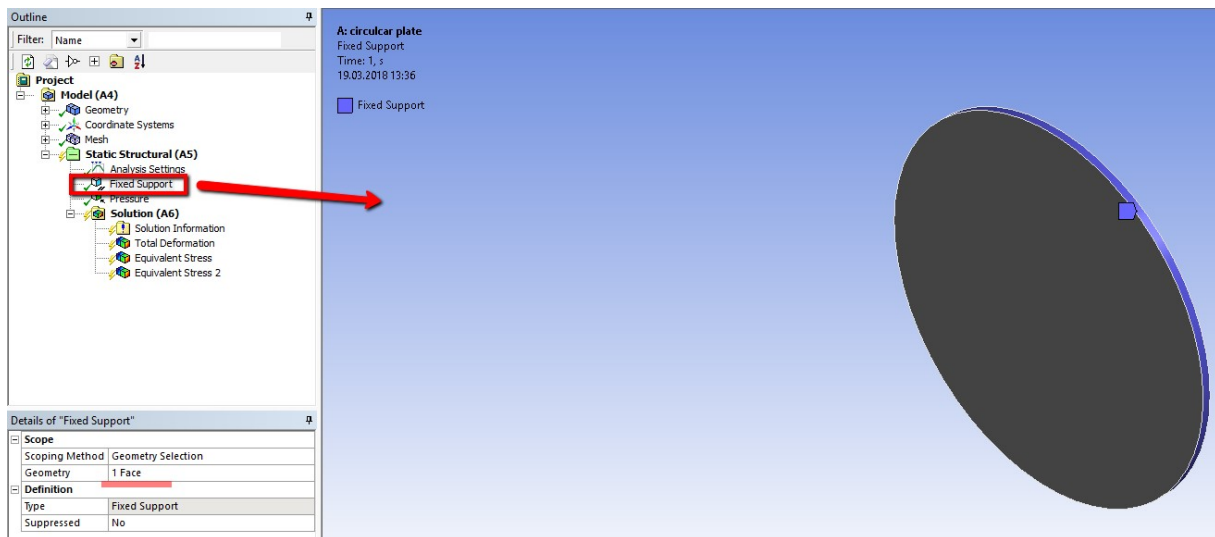


Рисунок 5.5 – Вибір типу закріплення

3. Визначити тип на величину зовнішнього навантаження. Для цього, у меню **Loads** слід обрати **Pressure** та визначити величину тиску у Паскалях (рис. 5.6)

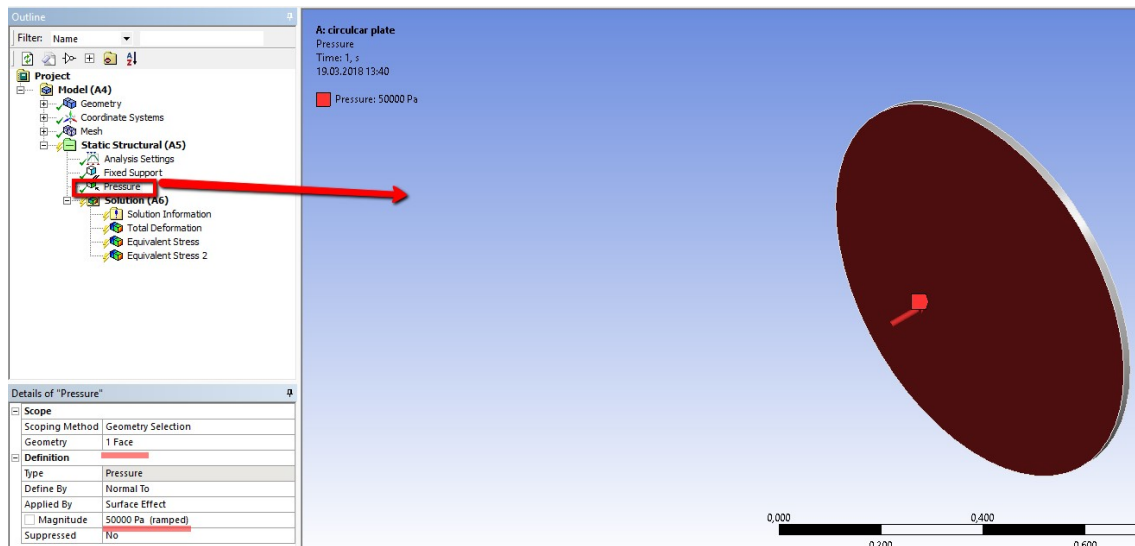


Рисунок 5.6 – Визначення зовнішнього навантаження на конструкцію

4. Розв'язати задачу. Для цього потрібно викликати розв'язувач з панелі інструментів. Процес розв'язання відбувається автоматично. Після вдалого розв'язання у дереві проекту з'явиться гілка **Solution** (рис. 5.7).

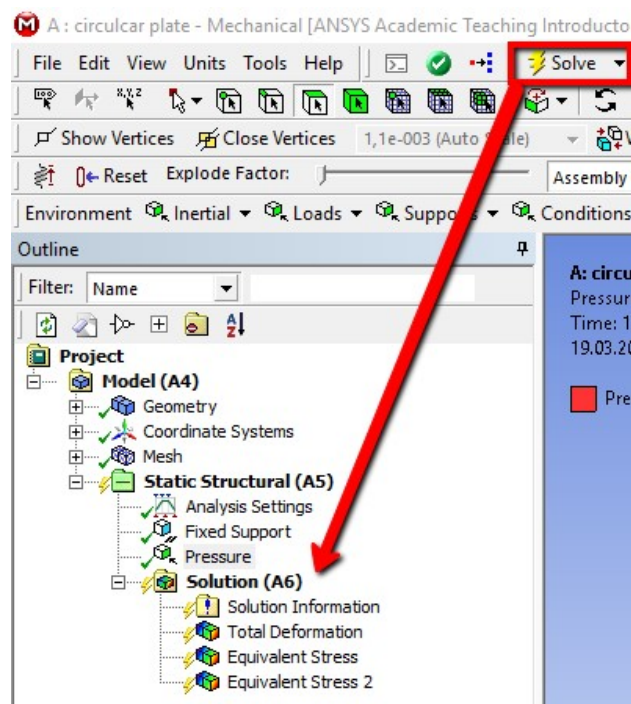


Рисунок 5.7 – Меню розв'язку задачі

5. Для візуалізації результатів обчислень, додайте у дерево проекту один з пунктів (**Deformation, Strain, Stress**) контекстного меню або панелі інструментів, які доступні після вибору гілки **Solution** (рис. 5.8).

6. Відобразити графічно результати обчислень. Для цього потрібно виконати команду **Evaluate All Results** для доданого у дерево проекту типу розв'язку (рис. 5.9).

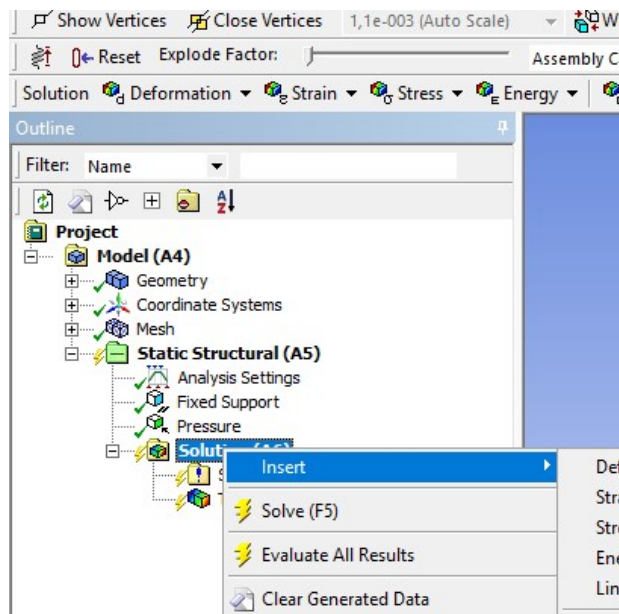


Рисунок 5.8 – Додання у дерево проекту візуалізації напруження, тиску, деформації

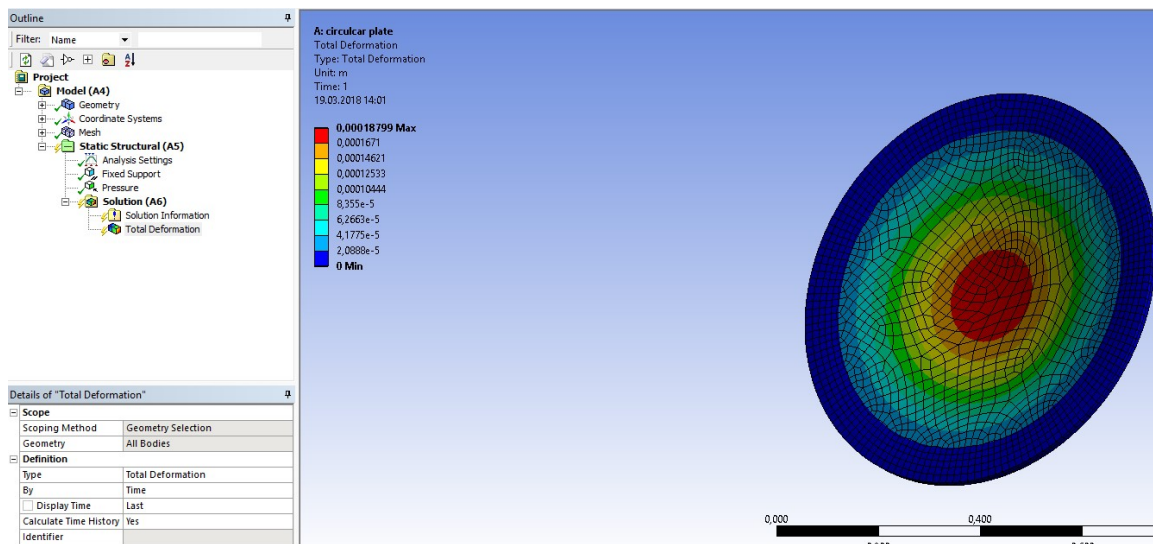


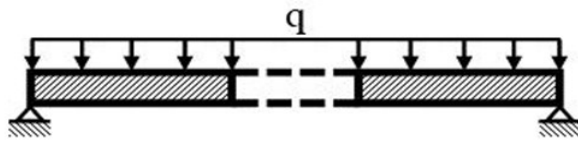
Рисунок 5.9 – Візуалізація отриманого розв'язку

Завдання до лабораторної роботи

1. Розв'язати задачу згину кільцевої пластини під дією поперечного тиску інтенсивності q (схема опертя згідно варіанта). Параметри пластини: товщина $h = 18 \cdot 10^{-3}$ м, радіус зовнішнього контуру $R_a = 0,4$ м, радіус внутрішнього контуру $R_b = 0,2$ м. Матеріал пластини – Structural Steel (Young's Modulus $E = 2E+11$ МПа, Poisson's Ratio $\nu = 0,3$). Вивести на екран переміщення точок пластини (Total Deformation).

Будемо розглядати наступні схеми граничних умов:

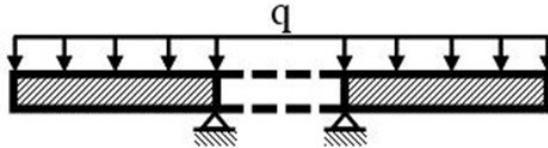
а) шарнірне опертя зовнішнього контуру, вільний внутрішній контур;



б) затиснення зовнішнього контуру, вільний внутрішній контур;



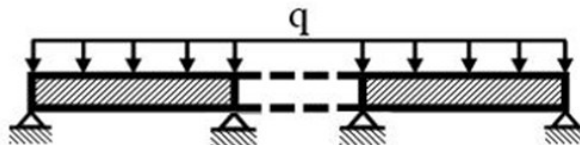
в) вільний зовнішній контур, шарнірне опертя внутрішнього контуру;



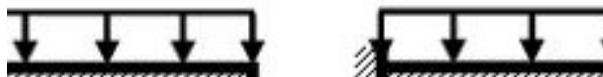
г) вільний зовнішній контур, затиснення внутрішнього контуру;



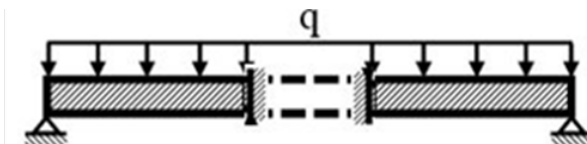
д) шарнірне опертя обох контурів;



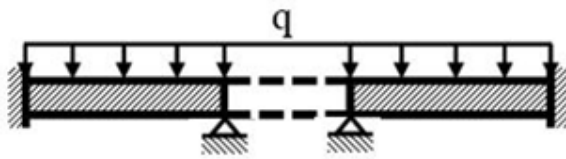
е) затиснення обох контурів;



ж) шарнірне опертя зовнішнього контуру, затиснення внутрішнього контуру;



з) затиснення зовнішнього контуру, шарнірне опертя внутрішнього контуру;



№ варіанта	q , МПа	схема опертя
1	0,05	а
2	0,07	а

3	0,09	а
4	0,11	а
5	0,05	б
6	0,07	б
7	0,09	б
8	0,11	б
9	0,05	в
10	0,07	в
11	0,09	в
12	0,11	в
13	0,05	г
14	0,07	г
15	0,09	г
16	0,11	г
17	0,05	д
18	0,07	д
19	0,09	д
20	0,11	д
21	0,05	е
22	0,07	е
23	0,09	е
24	0,11	е
25	0,05	ж
26	0,07	ж
27	0,09	ж
28	0,11	ж
29	0,05	з
30	0,07	з
31	0,09	з
32	0,11	з

? Контрольні запитання

1. Які основні етапи розв'язання задачі із використанням САПР?
2. У чому особливість систем класу CAE (Computer Aided Engineering)?
3. Яке призначення препроцесора?
4. Які основні функції процесора?
5. Які основні функції постпроцесора?

Лабораторна робота №6. Геометричне моделювання у програмі FreeCAD

Мета: вивчити основні засоби геометричного моделювання програми FreeCAD.

Теоретичні відомості

FreeCAD – це вільна САПР з відкритим вихідним кодом, яка може використовуватись для 2D та 3D моделювання.

Графічний інтерфейс має типовий для таких систем вигляд (рис. 6.1) та містить такі панелі інструментів:

- вікно відображення геометричної моделі (1);
- дерево проекту, яке містить всі об'єкти моделі та операції, що виконуються над об'єктами (2);
- редактор властивостей об'єктів моделі (3);
- вікно виведення, в якому виводяться повідомлення системи, помилки та ін. (4);
- консоль Python, де командами мови Python дублюються дії користувача та є можливість інтерактивного виконання команд (5).

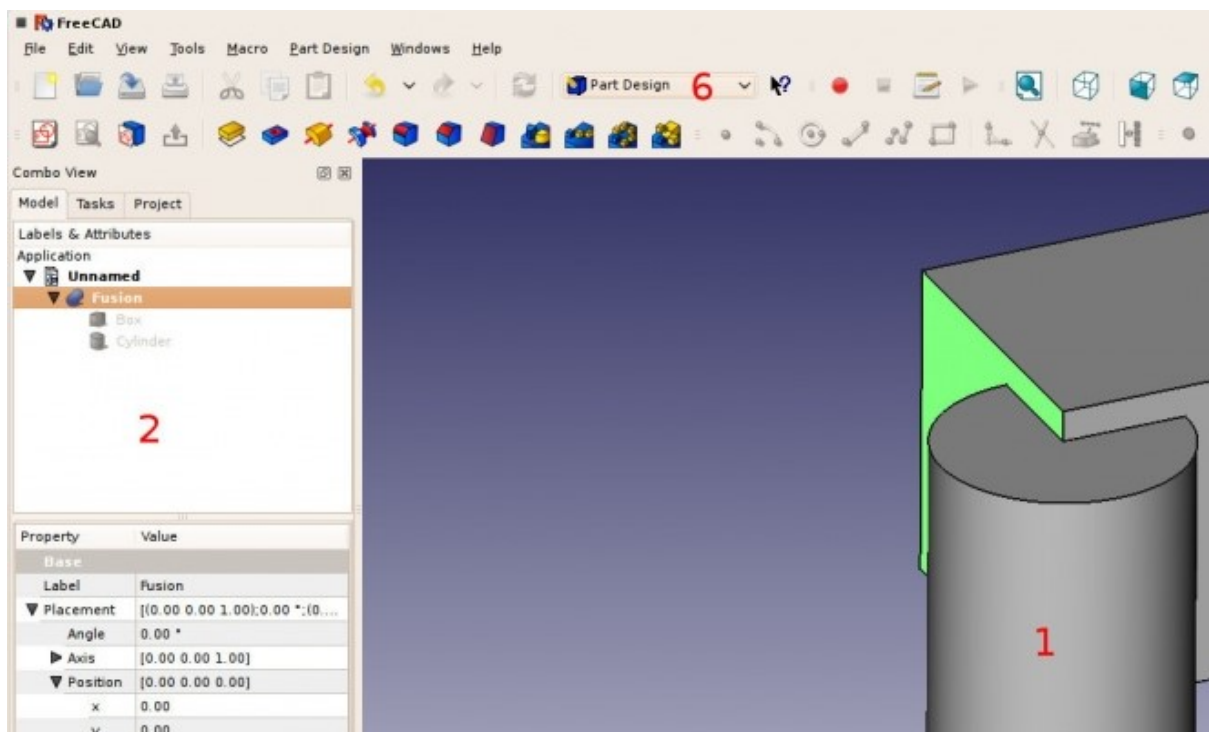


Рисунок 6.1 – Інтерфейс програми FreeCAD

Графічний інтерфейс користувача адаптується у залежності від того, який модуль (workbench) системи обрано для роботи. Деякі з модулів реалізовано на базі відкритої платформи Open Cascade Technology або CAS.CADE. Ця платформа, яку реалізовано на мові C++, використовується

для побудови на її основі 3D CAD, CAM та CAE систем. Також доступна оболонка CAS.CADE на Python (Python OCC, <http://www.pythonocc.org>).

Модуль **Part** (рис. 6.2) призначено для 3D моделювання деталей та виконання операцій над ними.

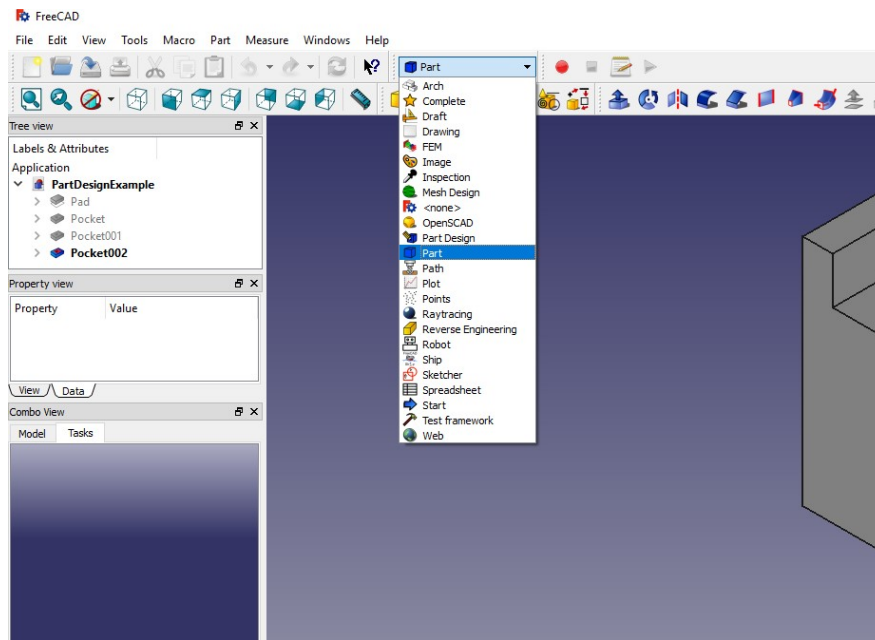


Рисунок 6.2 – Інтерфейс модуля **Part**

Для побудови геометричної моделі використовуються графічні примітиви, перелік яких міститься у меню **Part/Primitives** або на панелі інструментів (рис. 6.3).

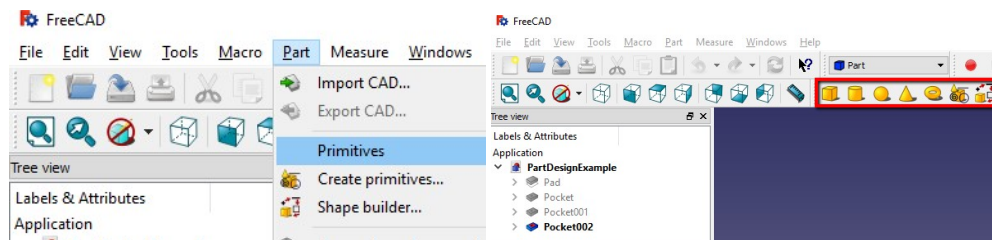




Рисунок 6.3 – Меню графічних примітивів

Алгоритм побудови геометричної моделі можна представити у вигляді послідовності етапів: вибір необхідних примітивів, параметризація примітивів, застосування операцій перетворення або логічних операцій.

Основні операції модуля наведено у таблиці 6.1.

Таблиця 6.1 Основні операції програми FreeCAD

Операція	Назва	Опис
	Booleans	Булеві операції для двох заданих форм
	Union	Об'єднання деяких форм

	Common	Перетин двох заданих форм
	Cut	Вирізання двох форм
	Connect	Об'єднання меж двох об'єктів
	Embed	Вбудова одного тонкостінного об'єкта в інший
	Cutout	Соворює виріз в одному тонкостінному об'єкті за формою іншого об'єкта
	Extrude	Видавлення плоских граней об'єктів
	Revolve	Створення твердого тіла обертанням іншого тіла (не суцільного) навколо обраної вісі
	Mirror	Відзеркалення обраний об'єкт на задану площину
	Fillet	Скруглення кутів (ребер) обраного об'єкту
	Chamfer	Створення фаски (скосу) ребер обраного об'єкту
	Ruled Surface	Загортання площини
	Loft	Лінійне поєднання однієї поверхні з іншою
	Sweep	Нелінійне поєднання однієї чи більше поверхонь
	3D Offset	Створення площини на деякій відстані від об'єкту
	Thickness	Вирізання тіла за допомогою чотирьох площин
	Section	Створення перерізу, перетинаючи об'єкт площиною
	Cross sections	Створення декількох перерізів
	Check geometry	Перевірка геометрії обраного об'єкту на помилки

Так, наприклад, операції (Connect, Embed, Cutout) частіше всього застосовуються для поєднання тонкостінних об'єктів, таких як труби, в один об'єкт.

☞ Завдання до лабораторної роботи

1. Виконати геометричне моделювання об'єктів з другої лабораторної роботи (завдання 1 – 3) у модулі **Part**.
2. Завдання 4 з лабораторної роботи реалізувати через консоль Python у модулі **Part**.

? Контрольні запитання

1. Які вільні програми та бібліотеки існують для створення CAD\CAE систем?
2. У чому полягає принцип побудови інтерфейсу системи FreeCAD?
3. Які модулі доступні для моделювання у системі FreeCAD?
4. Які дії потрібно виконати при геометричному моделюванні?
5. Які основні операції над об'єктами доступні у модулі Part системи FreeCAD?

Лабораторна робота №7. Створення сценаріїв засобами програми FreeCAD

Мета: опанувати методи написання та виконання сценаріїв у FreeCAD.

Теоретичні відомості

Для автоматизації деяких задач, що можуть повторюватися, або для опису досить складного технічного об'єкту (рис. 7.1) у системах CAD/CAE зазвичай передбачається можливість використовувати сценарії (скрипти).

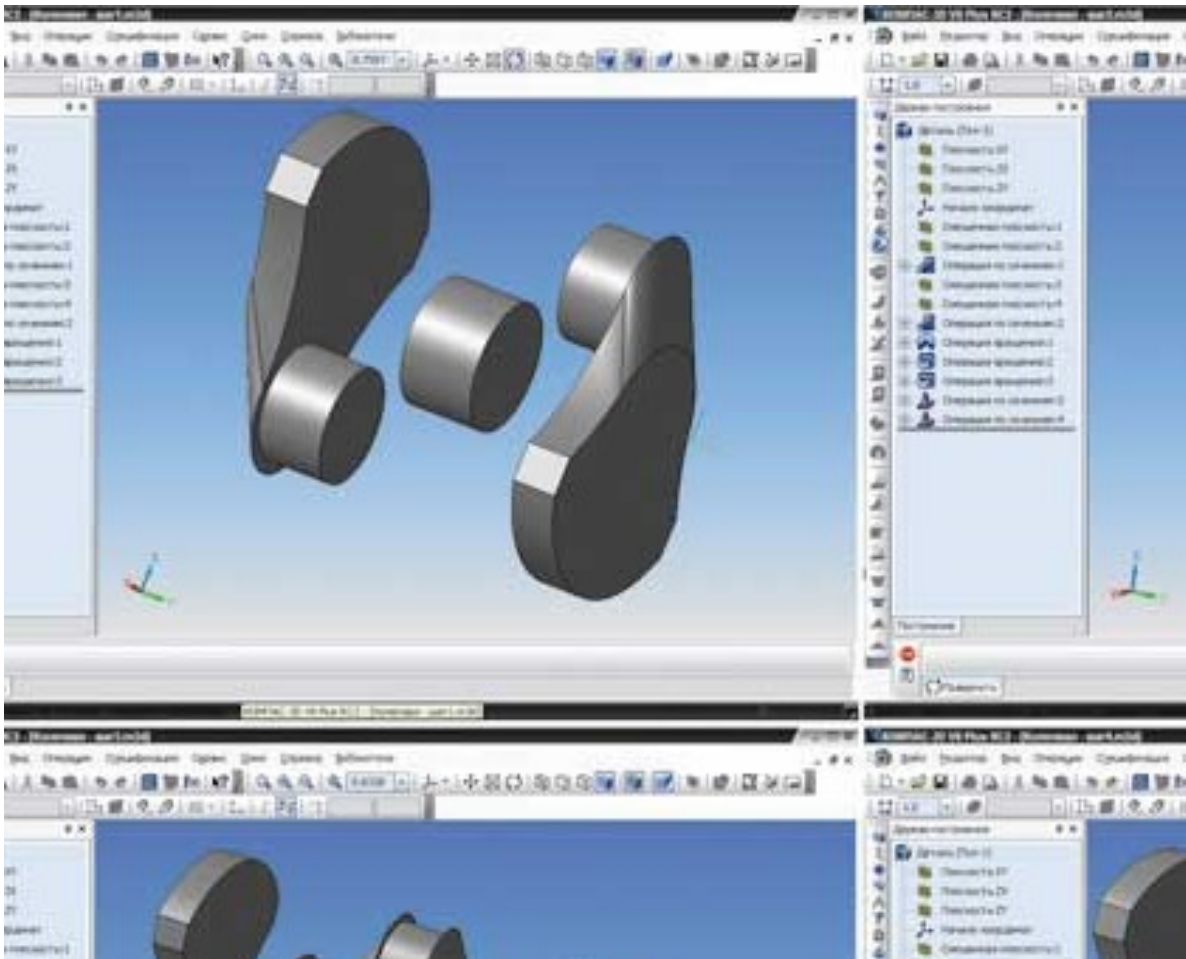


Рисунок 7.1 – Фрагмент колінчатого вала

У системі FreeCAD для цієї задачі використовується мова програмування Python. Для інтерактивного покрокового виконання команд сценарію використовується консоль у модулі **Part** (рис. 7.2)

```
Python console
Python 2.7.8 (default, Nov 17 2014, 20:37:05) [MSC v.1800 64 bit
Type 'help', 'copyright', 'credits' or 'license' for more inform
>>> import WebGui
>>> from StartPage import StartPage
>>> WebGui.openBrowserHTML(StartPage.handle(),App.getResourceDir
>>> App.newDocument("Unnamed")
>>> App.setActiveDocument("Unnamed")
>>> App.ActiveDocument=App.getDocument("Unnamed")
>>> Gui.ActiveDocument=Gui.getDocument("Unnamed")
```

Рисунок 7.2 – Фрагмент скрипта Python у модулі Part

Також, сценарії можна створити шляхом запису дій, які виконує користувач у графічному інтерфейсі, а потім за необхідністю редагувати (рис. 7.3).

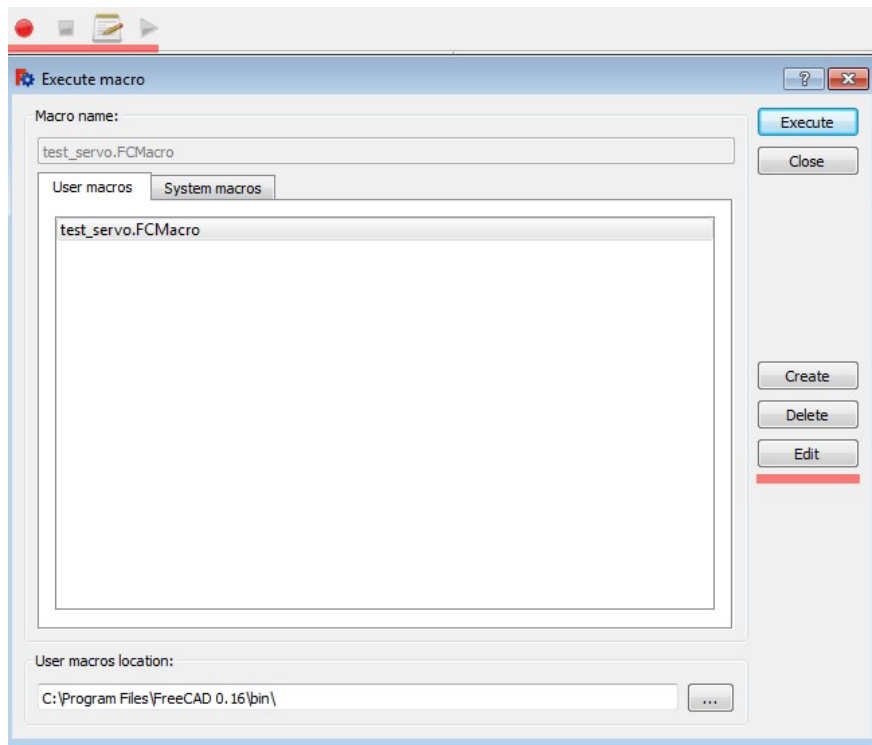


Рисунок 7.3 – Вікно редагування сценарію

Переважна більшість об’єктів відноситься до двох модулів:

- **App** модуля, який містить функції створення документів, файлів, об’єктів, підключення додаткових модулів;
- **Gui** модуля, який містить функції доступу та керування об’єктами графічного інтерфейсу.

Всі дії системи дублюються у відповідному документі. Він містить геометричну модель та може бути збережений у зовнішній файл. Один документ зазвичай містить як **App**, так і **Gui** об’єкти.

Далі наведемо деякі фрагменти коду Python:

- для доступу до об'єктів геометричної моделі даного документа:

```
myDocument = App.ActiveDocument
```

- для створення нового документа:

```
myDocument = App.newDocument("Document Name")
```

- для доступу до **Gui** об'єктів даного документа:

```
myGuiDocument = Gui.ActiveDocument
```

Всі модулі, які доступні у системі з графічного інтерфейсу, можна використовувати і при написанні сценаріїв. Наприклад, для створення об'єктів модуля Part:

```
import Part
cube = Part.makeBox(2, 2, 2)
Part.show(cube)
```

Додання об'єктів у документ:

```
myMesh =
FreeCAD.ActiveDocument.addObject("Mesh::Feature", "myMeshName")
myPart =
FreeCAD.ActiveDocument.addObject("Part::Feature", "myPartName")
```

Приклад 7.1. Створення відрізка:

```
import Part, PartGui
doc=App.activeDocument()
# add a line element to the document and set its
points
l=Part.Line()
l.StartPoint=(0.0, 0.0, 0.0)
l.EndPoint=(1.0, 1.0, 1.0)
doc.addObject("Part::Feature", "Line").Shape=l.toShape()
doc.recompute()
```

Приклад 7.2. Створення кола в активному документі:

```
import Part
```

```

doc = App.activeDocument()
c = Part.Circle()
c.Radius=10.0
# create a document with a circle feature
f = doc.addObject("Part::Feature", "Circle")
# Assign the circle shape to the shape property
f.Shape = c.toShape()
doc.recompute()

```

Приклад 7.3. Створення вала з прикріпленою пластиною з отворами:

```

import Part
from FreeCAD import Vector
plate = Part.makeBox(40,40,5,Vector(-20,-20,0))
hole1= Part.makeCylinder(1.5,5,Vector(-15,-15,0))
hole2= Part.makeCylinder(1.5,5,Vector(-15,15,0))
hole3= Part.makeCylinder(1.5,5,Vector(15,15,0))
hole4= Part.makeCylinder(1.5,5,Vector(15,-15,0))
faceplate = plate.cut(hole1)
faceplate = faceplate.cut(hole2)
faceplate = faceplate.cut(hole3)
faceplate = faceplate.cut(hole4)
motorbody=Part.makeCylinder(17.5,60,Vector(0,0,5))
shaft = Part.makeCylinder(3.175,15,Vector(0,0,-15))
servo = motorbody.fuse(faceplate)
servo = servo.fuse(shaft)
Part.show(servo)

```

Більш докладні приклади сценаріїв для створення геометричних моделей та роботи з системою наведено за посиланням https://www.freecadweb.org/wiki/Code_snippets

✍ Завдання до лабораторної роботи

Створити сценарій на Python геометричної моделі колінчатого вала (рис. 7.1).

? Контрольні запитання

1. Що таке скриптова мова програмування? Наведіть приклади.
2. Яке призначення модулів **App** та **Gui**?
3. Які методи інтегровано у модуль **Part**?
4. Які є методи у модулі **Draft**?
5. Які основні команди операцій над моделями?

Лабораторна робота №8. Геометричне моделювання засобами бібліотеки PythonOCC

Мета: вивчити методи геометричного моделювання бібліотеки PythonOCC.

Теоретичні відомості

Бібліотека PythonOCC (<http://www.pythonocc.org>) є оболонкою для бібліотеки OpenCascade Community Edition library, яку реалізовано на C++ та використовують для розробки CAD, CAM та CAE систем.

Інсталяція бібліотеки у середовищі Anaconda (Python2.7, Python3.6) виконується в Anaconda Prompt за допомогою команд:

```
conda config --append channels DLR-SC
conda config --append channels pythonocc
conda config --append channels oce
conda install pythonocc-core
```

Після інсталяції можна перевірити коректність імпортування бібліотеки та поточну версію (рис. 8.1).

```
Python 3.6.3 [Anaconda custom (64-bit)] (default, Oct 15 2017, 03:27:45)
v.1900 64 bit (AMD64)
Type "copyright", "credits" or "license" for more information.

IPython 6.1.0 -- An enhanced Interactive Python.

In [1]: import OCC

In [2]: OCC.VERSION
Out[2]: '0.17.3'
```

Рисунок 8.1 – Поточна версія бібліотеки, яку імпортовано

Вихідні коди та детальний опис кожного класу бібліотеки містяться на офіційному сайті на сторінках **Download** та **Documentation>API Documentation** відповідно. Зокрема, у папці **\examples** вихідного коду є приклади побудови геометричних моделей для різних технічних об'єктів.

Приклад 8.1. Програма тестування підключень основних бібліотек.

```
"""
\examples\core_helloworld.py
The very first pythonocc example. This uses to be
the script used to check the following points:

- pythonocc installation is correct, i.e. pythonocc
modules are found and properly imported
```

- a GUI manager is installed. Whether it is wxpython or pyqt/pyside, it's necessary to display a 3d window

- the rendering window can be initialized and set up, that is to say the graphic driver and OpenGL works correctly.

If this example run on your machine, that means you're ready to explore the wide pythonocc world and run all the other examples.

```
"""  
  
from OCC.Display.SimpleGui import init_display  
from OCC.BRepPrimAPI import BRepPrimAPI_MakeBox  
  
display, start_display, add_menu,  
add_function_to_menu = init_display()  
my_box = BRepPrimAPI_MakeBox(10., 20., 30.).Shape()  
  
display.DisplayShape(my_box, update=True)  
start_display()
```

Перший рядок прикладу 8.1 імпортує `init_display` функцію, яка відповідає за використання вбудованого GUI бібліотеки `pythonocc`:

```
from OCC.Display.SimpleGui import init_display.
```

Далі імпортується клас, який відповідає за налаштування властивостей геометричного примітиву паралелепіпед:

```
from OCC.BRepPrimAPI import BRepPrimAPI_MakeBox.
```

Тут модуль `BRepPrimAPI` (Boundary Representation Primitive API) містить методи створення таких графічних примітивів, як сфера, тор, конус та ін.

Далі, проводиться ініціалізація функції виведення. За умовчанням, функція `init_display` звертається до бібліотек графічного інтерфейсу на базі бібліотеки Qt (PyQt, PySide):

```
display, start_display, add_menu,  
add_function_to_menu = init_display()
```

Клас `BRepPrimAPI_MakeBox` ініціалізується параметрами ширини, довжини та висоти паралелепіпеда:

```
my_box = BRepPrimAPI_MakeBox(10., 20., 30.).Shape()
```

Після цього, об'єкт `my_box` передається методу візуалізатора:

```
display.DisplayShape(my_box, update=True)
```

та викликається функція візуалізації:

```
start_display()
```

Отже, основними поняттями при роботі з бібліотекою PythonOCC є графічні примітиви, операції над примітивами та способи їх відображення.

Приклад 8.2. Створення та відображення поверхні.

```
# \examples\core_topology_prism.py
from __future__ import print_function

from OCC.gp import gp_Pnt, gp_Vec
from OCC.GeomAPI import GeomAPI_PointsToBSpline
from OCC.TColgp import TColgp_Array1OfPnt
from OCC.BRepBuilderAPI import
BRepBuilderAPI_MakeEdge
from OCC.BRepPrimAPI import BRepPrimAPI_MakePrism

from OCC.Display.SimpleGui import init_display
display, start_display, add_menu,
add_function_to_menu = init_display()

def prism():
    # the bspline profile
    array = TColgp_Array1OfPnt(1, 5)
    array.SetValue(1, gp_Pnt(0, 0, 0))
    array.SetValue(2, gp_Pnt(1, 2, 0))
    array.SetValue(3, gp_Pnt(2, 3, 0))
    array.SetValue(4, gp_Pnt(4, 3, 0))
    array.SetValue(5, gp_Pnt(5, 5, 0))
    bspline = GeomAPI_PointsToBSpline(array).Curve()
    profile =
BRepBuilderAPI_MakeEdge(bspline).Edge()

    # the linear path
    starting_point = gp_Pnt(0., 0., 0.)
    end_point = gp_Pnt(0., 0., 6.)
    vec = gp_Vec(starting_point, end_point)
```

```

        path = BRepBuilderAPI_MakeEdge(starting_point,
end_point).Edge()

        # extrusion
        prism = BRepPrimAPI_MakePrism(profile,
vec).Shape()

        display.DisplayShape(profile, update=False)
        display.DisplayShape(starting_point,
update=False)
        display.DisplayShape(end_point, update=False)
        display.DisplayShape(path, update=False)
        display.DisplayShape(prism, update=True)

if __name__ == '__main__':
    prism()
    start_display()

```

У прикладі 8.2, виконується така перевірка:

```
if __name__ == '__main__'
```

Тобто код буде виконуватися тільки при умові того, що даний модуль запущено як програму, але забороняється його виконувати, якщо код модуля буде імпортовано.

Приклад 8.3. Логічні операції над графічними примітивами.

```

# \examples\core_topology_boolean.py

import sys
import time

from OCC.BRepAlgoAPI import BRepAlgoAPI_Fuse,
BRepAlgoAPI_Common, BRepAlgoAPI_Section,
BRepAlgoAPI_Cut
from OCC.BRepBuilderAPI import
BRepBuilderAPI_MakeFace, BRepBuilderAPI_Transform
from OCC.BRepPrimAPI import BRepPrimAPI_MakeBox,
BRepPrimAPI_MakeWedge, BRepPrimAPI_MakeSphere,
BRepPrimAPI_MakeTorus
from OCC.Display.SimpleGui import init_display
from OCC.gp import gp_Vec, gp_Ax2, gp_Pnt, gp_Dir,
gp_Pln, gp_Trnsf

```

```

    display, start_display, add_menu,
add_function_to_menu = init_display()

    def translate_topods_from_vector(brep_or_iterable,
vec, copy=False):
    '''
        translate a brep over a vector
        @param brep:    the Topo_DS to translate
        @param vec:    the vector defining the
translation
        @param copy:    copies to brep if True
    '''
    trns = gp_Trnsf()
    trns.SetTranslation(vec)
    brep_trns =
BRepBuilderAPI_Transform(brep_or_iterable, trns, copy)
    brep_trns.Build()
    return brep_trns.Shape()

    def fuse(event=None):
    display.EraseAll()
    box1 = BRepPrimAPI_MakeBox(2, 1, 1).Shape()
    box2 = BRepPrimAPI_MakeBox(2, 1, 1).Shape()
    box1 = translate_topods_from_vector(box1,
gp_Vec(.5, .5, 0))
    fuse_shp = BRepAlgoAPI_Fuse(box1, box2).Shape()
    display.DisplayShape(fuse_shp)
    display.FitAll()

    def common(event=None):
    # Create Box
    axe = gp_Ax2(gp_Pnt(10, 10, 10), gp_Dir(1, 2,
1))
    Box = BRepPrimAPI_MakeBox(axe, 60, 80,
100).Shape()
    # Create wedge
    Wedge = BRepPrimAPI_MakeWedge(60., 100., 80.,
20.).Shape()
    # Common surface
    CommonSurface = BRepAlgoAPI_Common(Box,
Wedge).Shape()

```

```

display.EraseAll()
ais_box = display.DisplayShape(Box)
ais_wedge = display.DisplayShape(Wedge)
display.Context.SetTransparency(ais_box, 0.8)
display.Context.SetTransparency(ais_wedge, 0.8)
display.DisplayShape(CommonSurface)
display.FitAll()

def slicer(event=None):
    # Param
    Zmin, Zmax, deltaZ = -100, 100, 5
    # Note: the shape can also come from a
shape
    # selected from InteractiveViewer
    if 'display' in dir():
        shape = display.GetSelectedShape()
    else:
        # Create the shape to slice
        shape = BRepPrimAPI_MakeSphere(60.).Shape()
    # Define the direction
    D = gp_Dir(0., 0., 1.) # the z direction
    # Perform slice
    sections = []
    init_time = time.time() # for total time
computation
    for z in range(Zmin, Zmax, deltaZ):
        # Create Plane defined by a point and the
        # perpendicular direction
        P = gp_Pnt(0, 0, z)
        Pln = gp_Pln(P, D)
        face = BRepBuilderAPI_MakeFace(Pln).Shape()
        # Computes Shape/Plane intersection
        section_shp = BRepAlgoAPI_Section(shape,
face)
        if section_shp.IsDone():
            sections.append(section_shp)
    total_time = time.time() - init_time
    print("%.3fs necessary to perform slice." %
total_time)

display.EraseAll()
display.DisplayShape(shape)
for section_ in sections:
    display.DisplayShape(section_.Shape())

```

```

display.FitAll()

def section(event=None):
    torus = BRepPrimAPI_MakeTorus(120, 20).Shape()
    radius = 120.0
    sections = []
    for i in range(-3, 4):
        # Create Sphere
        sphere = BRepPrimAPI_MakeSphere(gp_Pnt(26 *
3 * i, 0, 0), radius).Shape()
        # Computes Torus/Sphere section
        section_shp = BRepAlgoAPI_Section(torus,
sphere, False)
        section_shp.ComputePCurveOn1(True)
        section_shp.Approximation(True)
        section_shp.Build()
        sections.append(section_shp)

    display.EraseAll()
    display.DisplayShape(torus)
    for section_ in sections:
        display.DisplayShape(section_.Shape())
    display.FitAll()

def cut(event=None):
    # Create Box
    Box = BRepPrimAPI_MakeBox(200, 60, 60).Shape()
    # Create Sphere
    Sphere = BRepPrimAPI_MakeSphere(gp_Pnt(100, 20,
20), 80).Shape()
    # Cut: the sphere is cut 'by' the box
    Cut = BRepAlgoAPI_Cut(Sphere, Box).Shape()
    display.EraseAll()
    ais_box = display.DisplayShape(Box)
    display.Context.SetTransparency(ais_box, 0.8)
    display.DisplayShape(Cut)
    display.FitAll()

def exit(event=None):
    sys.exit()

```

```
if __name__ == '__main__':
    add_menu('topology boolean operations')
    add_function_to_menu('topology boolean
operations', fuse)
    add_function_to_menu('topology boolean
operations', common)
    add_function_to_menu('topology boolean
operations', cut)
    add_function_to_menu('topology boolean
operations', section)
    add_function_to_menu('topology boolean
operations', slicer)
    add_function_to_menu('topology boolean
operations', exit)
    start_display()
```

✍ Завдання до лабораторної роботи

Створити засобами бібліотеки PythonOCC геометричну модель колінчастого валу (рис. 7.1).

? Контрольні запитання

1. Які основні класи бібліотеки PythonOCC?
2. Що таке геометричний примітив?
3. Які основні логічні операції над графічними примітивами доступні у бібліотеці?
4. Які підходи до відображення створених моделей реалізовано у бібліотеці PythonOCC?
5. Які існують бібліотеки геометричного моделювання?

Україно-англійський словник найбільш вживаних термінів

Балка	beam
Вісь	axis
Вузол	knot
Геометрична модель	geometric model
Деформація	deformation
Довжина	length
Жорстка задача	rigid task
Жорстке закріплення	rigid fastening
Згин	bending
Зовнішнє навантаження	external loading
Ізотропність	isotropy
Кільцева пластина	annular plate
Конус	cone
Коефіцієнт Пуассона	quotient Puassona
Міцність	strength
Модуль пружності	the elasticity module
Наближення	approach
Навантаження	load, loading
Напружено-деформований стан	strains-deformed state
Напруження	stress
Обробка даних	data processing
Об'ємні сили	body loads
Отвір	mesh
Переміщення	displacement
Пластина	plate
Площина	plane
Поверхневі сили	surface loads
Поверхня	surface
Попередження	warning
Пружність	elasticity
Розтягнення	strain
Система автоматизованого проектування	computer-aided design system
Стійкість	rigidity, firmness
Стискання	compression
Стрижень	pivot
Сфера	sphere
Товщина	thickness
Шарнірне оперття	articulated abut
Ширина	width

Глосарій

Автоматизоване проектування – проектування, при якому окремі перетворення об'єкта й (або) алгоритму його функціонування або алгоритму процесу, а також описи різноманітними мовами здійснюються взаємодією людини та комп'ютера.

Балка – горизонтальний брус, закріплений на опорах, який зазнає деформацію згину.

Брус – тіло, в якого один розмір значно більший за два інших.

Граничні розміри – два гранично допустимих розміри, між якими має розташовуватись дійсний розмір деталі.

Деформація – здатність тіла змінювати форму та розміри під дією навантажень. Деформації можуть бути пружними, що зникають після закінчення дії сил, що її викликають, або пластичними, тобто залишковими, що не зникають.

Екструзія – процес отримання виробів методом видавлювання матеріалу крізь формувальний отвір у матриці.

Жорсткість – здатність конструкції та її елементів чинити опір змінюванню своїх первинних розмірів і форми.

Жорстке затиснення – тип затиснення, при якому виключається можливість переміщень та повороту балки.

Згин – вид деформації, що характеризується скривленням осі або середньої поверхні об'єкта, що деформується (балки, плити, оболонки тощо).

Зосереджене навантаження – навантаження, що діє на площинці, дуже малій порівняно із загальними розмірами деталі, та яке умовно вважається прикладеним у точці.

Ізотропність – однаковість фізичних властивостей матеріалу (теплопровідність, електропровідність, пружність, оптичні властивості тощо) в усіх напрямках.

Коефіцієнт Пуассона (коефіцієнт деформації) – абсолютна величина відношення відносної поперечної деформації до відносної подовжньої деформації.

Метод скінченних елементів – числова техніка знаходження розв'язків інтегральних та диференціальних рівнянь у частинних похідних (ДРЧП). Процес розв'язання побудовано або на повному усуненні диференціального рівняння для стаціонарних задач, або на розкладі ДРЧП в апроксимуючу систему звичайних диференціальних рівнянь, які потім розв'язуються використанням якої-небудь стандартної техніки, такої як метод Ейлера, Рунге-Кутти тощо.

Міцність – здатність конструкції, її частин і деталей витримувати граничне навантаження без руйнування.

Модуль Юнга – фізична величина, що характеризує властивості матеріалу опиратися розтягуванню, стисненню при пружній деформації.

Навантаження – силова взаємодія, що викликає змінювання напружено-деформованого стану ланок в механізмі. Навантаження, яке не змінюється в часі, називається статичним, а те, яке змінюється – динамічним.

Напруження – міра внутрішніх сил, що виникають в тілі під впливом зовнішніх діянь (навантажень, зміни температури).

Напружено-деформований стан – сукупність внутрішніх напружень і деформацій конструкції або її елементу, що виникають при дії на неї зовнішніх навантажень, температурних полів чи інших факторів.

Оболонка – це тіло, обмежене криволінійними поверхнями, які розташовані на близькій відстані одна від одної.

Опір матеріалів – наука про інженерні методи розрахунків на міцність, жорсткість і стійкість елементів машин і споруд.

Переміщення – вектор змінювання положення рухомої матеріальної точки відносно системи відліку.

Пластина – тіло, що має форму прямої призми або прямого циліндру, висота (товщина) якого є малою в порівнянні з розмірами основи.

Пружність – властивість тіла відновлювати свою форму та об'єм або тільки об'єм (для газу та рідини) по закінченні дії сил або з інших причин, що викликають деформацію тіла.

Розподілене навантаження – сили, які прикладено безперервно впродовж деякої довжини або площі конструкції.

Розтягнення – вид деформації стрижня або його частини під дією поздовжніх розтяжних зусиль. Розтягнення характеризується змінням довжини стрижня або його частини.

Система автоматизованого проектування – автоматизована система, яка реалізує інформаційну технологію виконання функцій проектування, представляє собою організаційно-технічну систему, яку призначено для автоматизації процесу проектування. Використовується абревіатура САПР.

Статика – розділ механіки, який займається вивченням умов рівноваги рідких, твердих, газоподібних тіл.

Стійкість – здатність конструкції або її елементів зберігати певну початкову форму пружної рівноваги.

Стискання – вид деформації стрижня (бруса) або його частини під дією поздовжніх стискаючих зусиль.

Стрижень – тонкий та довгий брус із прямолінійною віссю.

Ступінь свободи об'єкта – кількість параметрів (координат) для визначення розташування об'єкта у просторі.

Шарнірно-рухома опора – опора, що накладає заборону на переміщення в певному напрямку (перпендикулярно до опорної поверхні).

Шарнірно-нерухома опора – опора, що допускає тільки поворот балки відносно точки закріплення (шарніра).

Рекомендована література

1. Бруйка В.А., Фокин В.Г., Солдусова Е.А. Инженерный анализ в ANSYS WORKBENCH. Самара : СГТУ, 2010. 271 с.
2. Шингель Л.П. Системы автоматизированного проектирования. Решение задач прочностного анализа с использованием пакета программ ANSYS 12.1. Пермь : ПНИПУ, 2015. 53 с.
3. Каменских А.А. Реализация решения задач механики контактного взаимодействия в прикладном пакете ANSYS. Пермь : ПНИПУ, 2017. 65 с.
4. Пальчевський Б.О., Валецький Б.П., Вараніцький Т.Л. Системи 3D моделювання. Луцьк : ІВВ Луцького НТУ, 2016. 176 с.
5. Гришенцев А.Ю., Гурьянов А.В., Кузнецова О.В. Математическое обеспечение в системах автоматизированного проектирования. Санкт-Петербург : Университет ИТМО, 2017. 88 с.
6. Гришенцев А.Ю., Гурьянов А.В., Тушканов Е.В. Виртуализация и программное обеспечение в системах автоматизированного проектирования. Санкт-Петербург : Университет ИТМО, 2017. 60 с.
7. Huei-Huang Lee. Finite Elements Simulations with ANSYS Workbench 12. Taiwan : SDC Publications, 2010. 592 p.
8. Литовка Ю.В. Получение оптимальных проектных решений и их анализ с использованием математических моделей. Тамбов : ТГТУ, 2006. 160 с.
9. Малюх В.Н. Введение в современные САПР. Москва : ДМК Пресс, 2010. 192 с.
10. Наумчук О.М. Основи систем автоматизованого проектування: інтерактивний комплекс навчально-методичного забезпечення Рівне : НУВГП, 2008. 136 с.
11. Норенков И.П. Автоматизированное проектирование. Москва : МГТУ им. Н.Э. Баумана, 2000. 188 с.
12. Перельмутер А.В., Сливкер В.И. Расчётные модели сооружений и возможность их анализа. Москва : «SCAD Soft», 2011. 732 с.
13. Алямовский А.А. SolidWorks. Компьютерное моделирование в инженерной практике. Санкт-Петербург : БХВ-Петербург, 2005. 740 с.
14. Басов К.А. Графический интерфейс комплекса ANSYS. Москва : ДМК Пресс, 2006. 241 с.

Навчально-методичне видання
(українською мовою)

Кудін Олексій Володимирович
Кривохата Анастасія Григорівна

ОСНОВИ АВТОМАТИЗОВАНОГО ПРОЕКТУВАННЯ СКЛАДНИХ ОБ'ЄКТІВ І СИСТЕМ

Методичні рекомендації до виконання лабораторних робіт
для здобувачів ступеня вищої освіти бакалавра
спеціальності «Інженерія програмного забезпечення»
освітньо-професійної програми «Програмна інженерія»

Рецензент *С.М. Гребенюк*
Відповідальний за випуск *А.О. Лісняк*
Коректор *А.Г. Кривохата*