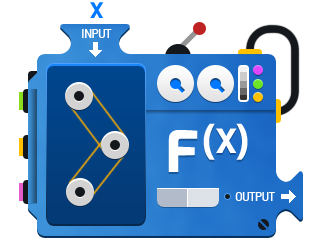
**Lesson 12. Methods**

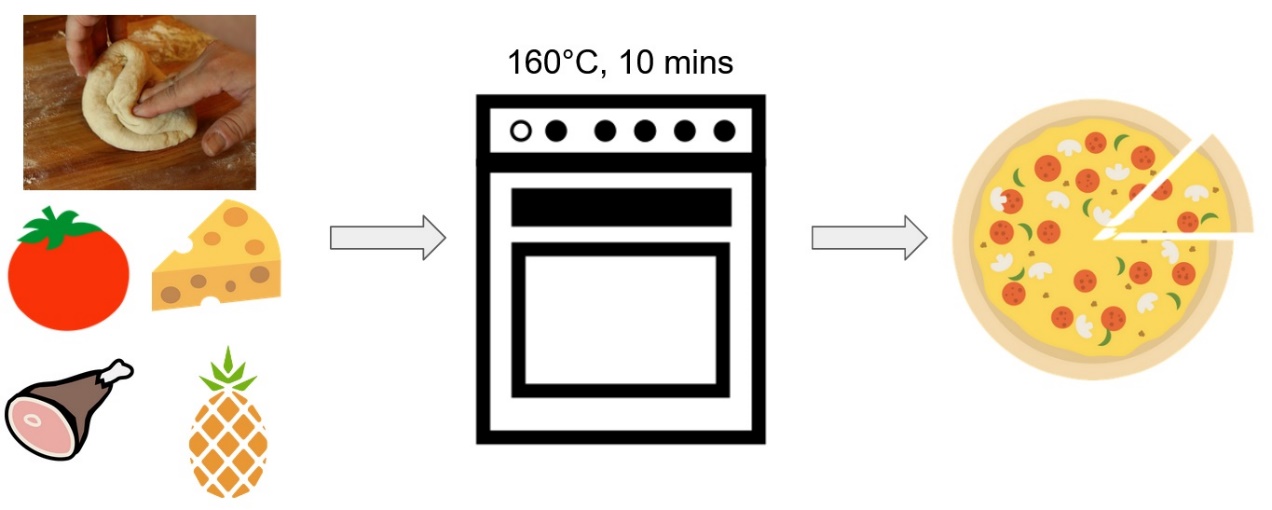
Одне із головних правил програмування – «повторне використання коду». Це означає, що алгоритм, який повинен бути виконаний кілька раз, не повинен бути скопійований, а повинен бути використаний ще раз.

Щоб наші програми більше відповідали даному правилу ми познайомимось із методами. Що воно таке ті методи, як і коли їх створювати, як називати, як оголосити параметри? Давайте з’ясовувати разом!

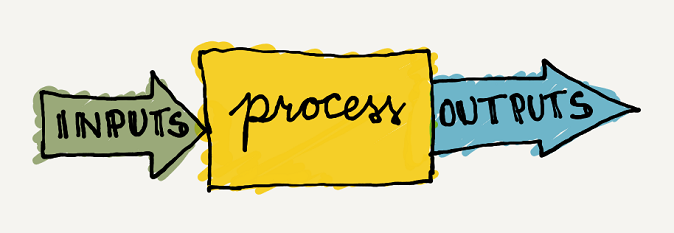
**What is a method?**

**Метод** – це блок коду, який можна викликати (використовувати повторно). Кожного разу, коли метод викликається, виконується код, який у ньому записано.

Метод може мати *вхідні дані* та/або *вихідні дані*. Метод можна уявити як пристрій, який отримує вхідні матеріали, обробляє їх і видає готовий продукт (результат).



У випадку із методом, вхідні та вихідні матеріали – це просто дані, які надаються методу і отримуються внаслідок його виконання.



Метод повинен являти собою окрему завершену операцію, тобто виконувати якусь одну коротку функцію. Наприклад «приготувати піцу». При цьому дана операція повинна складатись із викликів менших функцій («нарізати сир», «розкатати тісто», «зварити соус» тощо).

Таким чином код структурується, його більші частини складаються з менших, його стає легше розуміти і легше прочитати.

Клас може мати велику кількість методів. До цього часу ми оголошували тільки один із них – метод **main**.

**How to use methods?**

**main** зазвичай є першим методом, про який ви дізнаєтесь, коли починаєте програмувати на Java, оскільки він є точкою входу для виконання програми Java. Цей метод може містити код для виконання або виклику інших методів, і його можна розмістити в будь-якому класі, який є частиною програми. Більш складні програми зазвичай мають клас, який містить лише метод **main**.

Оголошення інших методів має схожий запис (метод *sayHello* – рядки 6 – 8):



У запису створення нового методу можемо визначити такі частини:

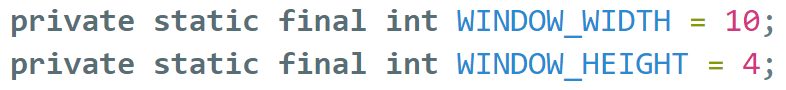
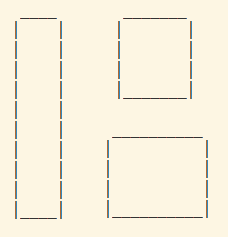
* ***private static*** – модифікатори доступу (поки що будемо користуватись саме таким, а під час роботи із класами розглянемо, які можуть бути інші і що взагалі вони означають);
* ***void***– тип значення, яке буде повернуто методом у програму у результаті його виконання (це може бути будь-який тип, або void, що буквально означає відсутність даних, які повертає метод);
* ***sayHello*** – ім’я методу, за яким його можна викликати;
* () – пара круглих дужок призначається для опису в них вхідних даних, які повинен отримувати метод (якщо метод не потребує вхідних даних дужки залишаються пустими);
* {} – пара фігурних дужок визначає тіло методу, або ж код, який повинен виконуватись при його виклику (замість одної дії, може бути певний алгоритм).

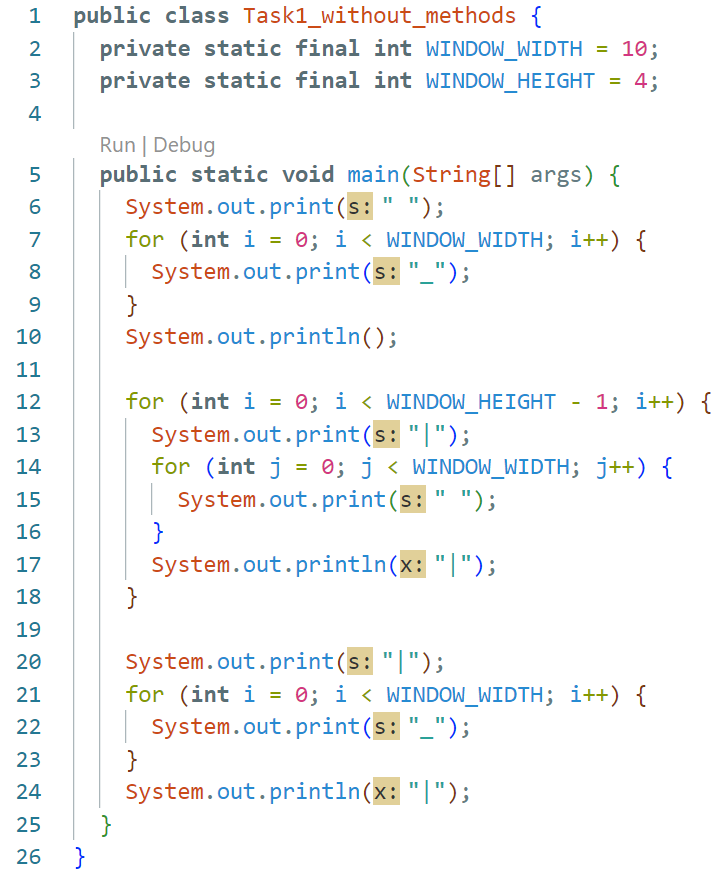
У даному занятті ми не будемо розглядати методи, які повертають значення у програму, тобто всі вони матимуть тип *void*. Але далі ми обов’язково розберемось і з ними також.

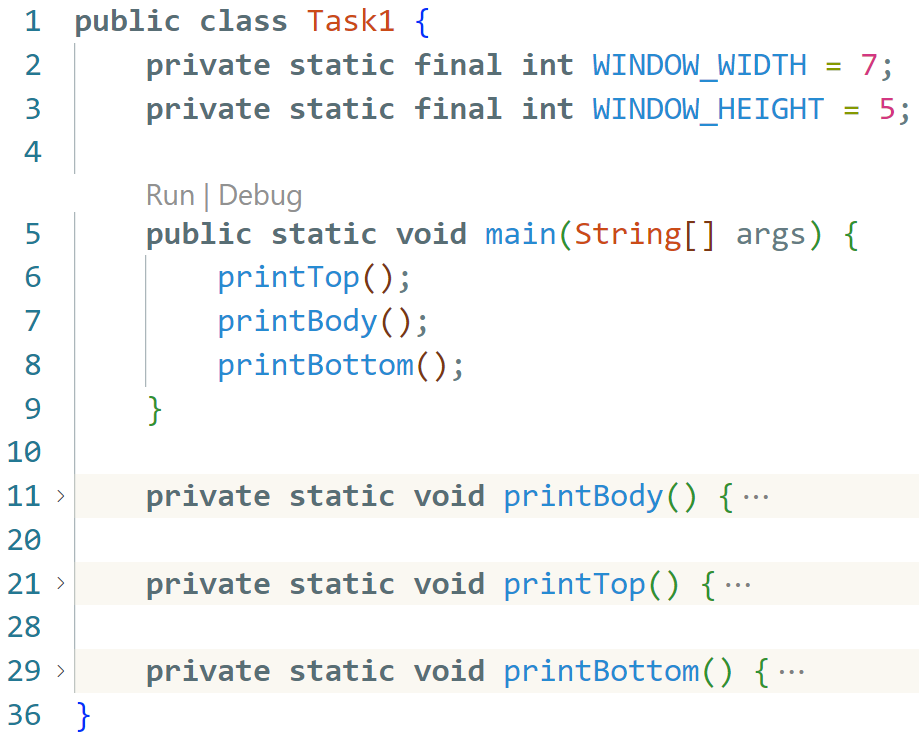
**Tasks**

**Завдання 1**. Створіть програму, що буде виводити у консоль віконечко (рамку) як показано на рисунку нижче.

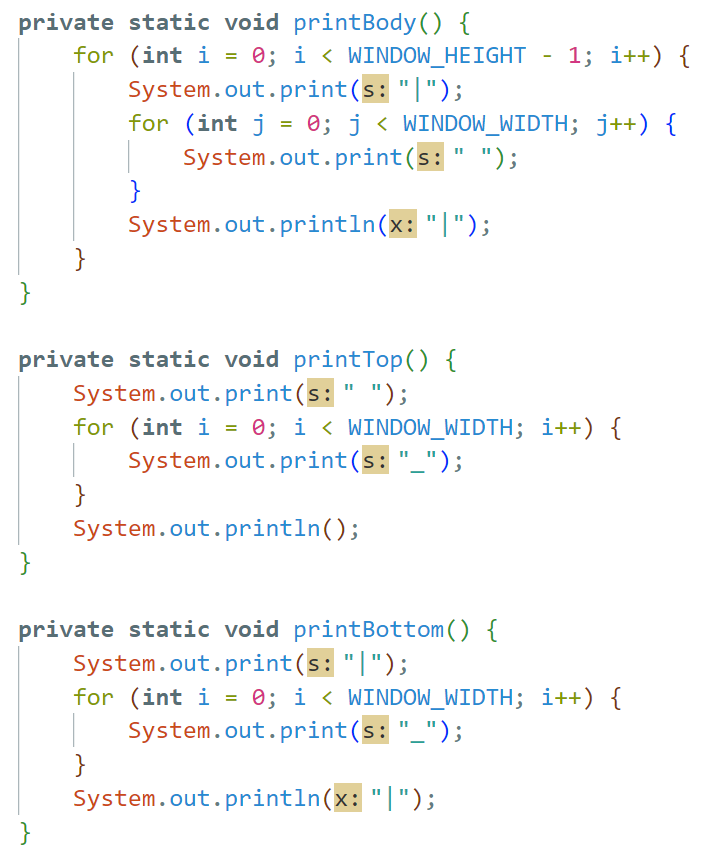
Ширина і довжина віконечка повинна бути задана константами:



Приклад 1.Виконання задачі без використання методів:

Приклад 2.Виконання задачі за допомогою методів:

На скріншоті вгорі код методів згорнуто для зручності перегляду. Нижче показано код кожного метода окремо.



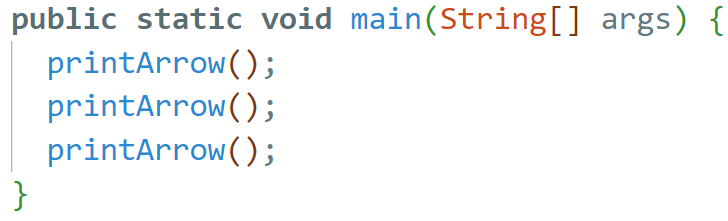
Запитання. Проаналізуйте код прикладів і дайте відповідь на питання, чому код із використанням методів вважатиметься кращим?

**New things**

**Завдання 1.** Створіть метод, що буде виводити у консоль стрілку. Довжина повинна задаватись константою класу:



Тестові дані:



Очікуваний результат:

