GitHub Copilotin käyttö ja arviointi

Tehtävä 1: Metodien kirjoittaminen

Käytin GitHub Copilotia nopeuttamaan `Calculator`-luokan kirjoittamista. Prosessi eteni seuraavasti.

1. **Metodien rungot:** Aloitin kirjoittamalla metodin nimen, esimerkiksi `public void add(int number)`. Copilot tunnisti heti luokan kontekstin ja ehdotti metodin sisällöksi `this.currentSum += number;`.

2. **Logiikan ohjaaminen kommenteilla:** Negatiivisten lukujen käsittelyä varten kirjoitin metodin yläpuolelle englanninkielisen kommentin: *// Adds a number. Throws exception if number is negative.* Tämän kommentin perusteella Copilot osasi automaattisesti lisätä `if (number < 0)` -tarkistuksen ja poikkeuksen heittämisen ennen yhteenlaskua.

3. **Silmukoiden muokkaus:** Main-metodia kirjoittaessa Copilot ehdotti ensin peräkkäisiä `System.out.println`-komentoja. Kokeilin kirjoittaa `for`-silmukan aloituksen, jolloin Copilot täydensi koodin käymään läpi lukulistan. Kun vaihdoin rakenteen `while`-silmukaksi, Copilot ehdotti korvaavaa koodia, joka toimi samalla logiikalla mutta eri silmukkarakenteella.

Tehtävä 2: Koodin selittäminen

Dokumentaatiota varten kopioin koodin Markdown-tiedostoon. Kirjoitin otsikon `## Class Explanation`, minkä jälkeen Copilot analysoi yllä olevan koodilohkon ja generoi listauksen luokan toiminnasta. Copilotin tuottama teksti oli ihan sopiva.

Arviointi: GitHub Copilotin soveltuvuus

Copilot tekoäly on ihan ok sovellus sen avulla voi tehä toimivia koodi sisältöjä mietimättä koodi rakenteesta yhtään. Joskus vasaus voi olla huono ja pitä aina tarkista toimiko se vai ei.

Hyödyt

- Nopeus: Rutiinikoodin tuottaminen on huomattavasti nopeampaa kuin käsin kirjoittaminen.
- Syntaksiapu: Copilot muistaa oikean syntaksin (esim. poikkeusten heittäminen tai try-catch-lohkot) usein paremmin kuin minä ulkoa, mikä vähentää googlailun tarvetta.
- Dokumentointi: Työkalu on yllättävän hyvä kääntämään koodin toimintalogiikan selkeäksi englanninkieliseksi tekstiksi.

Haitat ja riskit

- Sokea luottamus: On riski hyväksyä ehdotettu koodi lukematta sitä ajatuksella läpi. Esimerkiksi ilman kommenttiani Copilot olisi aluksi sallinut negatiiviset luvut, mikä oli vastoin tehtävänantoa.
- Oppimisen vaarantuminen: Aloittelijalle työkalu voi antaa valmiin ratkaisun liian helposti, jolloin oma ongelmanratkaisukyky ei kehity.
- Hienovaraiset virheet: Joskus ehdotukset näyttävät oikeilta, mutta niissä voi olla pieniä logiikkavirheitä (esim. silmukan rajat väärin), jotka huomaa vasta testatessa.