Programmeren met AI Copilot

Lesplan voor MBO/HBO - Informatica

Duur: 120 minuten

Omschrijving

In deze les leren studenten hoe ze GitHub Copilot en andere AI-codeerassistenten kunnen gebruiken om hun programmeervaardigheden te verbeteren. Ze ontdekken hoe AI kan helpen bij het schrijven, begrijpen en debuggen van code, terwijl ze tegelijkertijd kritisch leren nadenken over de gegenereerde suggesties.

Leerdoelen

Na deze les kunnen studenten:

- AI-codeerassistenten effectief configureren en gebruiken
- Effici^«nt code schrijven met behulp van AI-suggesties
- AI-gegenereerde code kritisch evalueren en verbeteren
- De beperkingen en ethische aspecten van AI-geassisteerd coderen begrijpen

Benodigde materialen

- Computers met VS Code of een andere ondersteunde IDE ge^flnstalleerd
- GitHub Copilot toegang (studententoegang is gratis)
- Alternatieve AI-codeerassistenten voor vergelijking:
  * Amazon CodeWhisperer
  * Tabnine
  * ChatGPT/Claude voor codeervragen
- Voorbeeldprojecten en codeeropgaven (bijgevoegd)

Voorbereiding

Docent:

- Zorg dat alle AI-tools vooraf zijn ge^flnstalleerd en getest
- Bereid voorbeelden voor van goede en slechte AI-suggesties
- Cre^«er opdrachten van verschillende moeilijkheidsgraden

Studenten:

- GitHub-account aanmaken (als ze dit nog niet hebben)
- GitHub Student Developer Pack aanvragen voor gratis Copilot-toegang
- VS Code of een andere ondersteunde IDE installeren

Lesopbouw

1. Introductie (15 minuten)

- Wat zijn AI-codeerassistenten en hoe werken ze?
- Demonstratie van GitHub Copilot in actie
- Discussie: Hoe verandert AI het programmeeronderwijs?

2. Setup en configuratie (20 minuten)

- Installatie en activatie van GitHub Copilot
- Configuratie van voorkeuren en instellingen
- Eerste tests om te controleren of alles werkt

3. Basisvaardigheden (25 minuten)

- Oefening 1: Code aanvullen met Copilot
  * Schrijf een commentaar met wat je wilt bereiken
  * Laat Copilot suggesties doen
  * Evalueer en pas de suggesties aan
- Oefening 2: Functies implementeren
  * Schrijf alleen de functiesignatuur en documentatie
  * Laat Copilot de implementatie genereren
  * Beoordeel de kwaliteit van de gegenereerde code

4. Gevorderde technieken (30 minuten)

- Prompt engineering voor betere code-suggesties
- Gebruik van Copilot voor het debuggen van bestaande code
- Complexe algoritmen implementeren met AI-assistentie
- Vergelijking van verschillende AI-codeerassistenten

5. Projectopdracht (20 minuten)

- Studenten werken aan een klein project naar keuze:
  a. Webapplicatie met formuliervalidatie
  b. Datavisualisatie script
  c. API-client voor een publieke API
- Ze gebruiken Copilot om sneller te ontwikkelen
- Docent loopt rond voor ondersteuning

6. Reflectie en discussie (10 minuten)

- Wat zijn de sterke en zwakke punten van AI-codeerassistenten?
- Hoe beˆflnvloedt dit het leerproces van programmeren?
- Ethische overwegingen: copyright, plagiaatdetectie, etc.

Voorbeeldopdrachten

Basisopdracht: Tekstverwerking

Schrijf het volgende commentaar en laat Copilot de implementatie genereren:

```
// Functie die een string neemt en het volgende doet:
// 1. Alle woorden in hoofdletters zet
// 2. Spaties vervangt door underscores
// 3. Het resultaat teruggeeft
// Naam van de functie: formatText
```

Gevorderde opdracht: API-client

Vraag Copilot om een client te maken voor de OpenWeatherMap API:

```
// Maak een klasse WeatherClient die de OpenWeatherMap API gebruikt
// De klasse moet de volgende functionaliteit hebben:
// - Constructor die API key accepteert
// - Methode getCurrentWeather(city) die het huidige weer ophaalt
// - Methode getForecast(city, days) die een weersvoorspelling ophaalt
// - Foutafhandeling voor ongeldige steden of API-problemen
```

Debugging opdracht:

Geef studenten buggy code en laat ze Copilot gebruiken om de problemen op te sporen:

```
// Deze functie zou de som van alle even getallen in een array moeten berekenen
// maar er zitten meerdere bugs in. Gebruik Copilot om ze te vinden en op te lossen.
function sumEvenNumbers(numbers) {
  let sum;
  for (let i = 0; i <= numbers.length; i++) {
    if (numbers[i] % 2 = 0) {
      sum += numbers[i]
    }
  }
}
```

Beoordelingscriteria

Studenten worden beoordeeld op:
- Effectief gebruik van AI-codeerassistenten (25%)
- Kwaliteit en correctheid van de gegenereerde code (25%)
- Kritische evaluatie en verbetering van AI-suggesties (25%)
- Begrip van de onderliggende programmeerconcepten (25%)

Tips voor docenten

- Benadruk dat AI een hulpmiddel is, geen vervanging voor begrip
- Moedig studenten aan om te experimenteren met verschillende prompts
- Bespreek hoe AI kan helpen bij het leerproces door code te verklaren
- Wijs op het belang van het begrijpen van de gegenereerde code
- Integreer codereviews om kritisch denken te stimuleren

Uitbreidingsmogelijkheden

Voor gevorderde studenten of vervolgactiviteiten:
- Vergelijkende analyse van verschillende AI-codeerassistenten
- Onderzoek naar hoe AI codeerkwaliteit kan verbeteren
- Ontwikkeling van custom prompts voor specifieke programmeertaken
- Integratie van AI in bestaande ontwikkelworkflows

Aanvullende bronnen

- GitHub Copilot documentatie: https://docs.github.com/copilot
- "AI-Assisted Programming" door Sarah Johnson (2024)
- Online cursus: "Mastering GitHub Copilot" op Pluralsight
- YouTube-kanaal: "AI Coding Tutorials" voor praktische voorbeelden

Reflectievragen voor studenten

- Hoe verandert AI-assistentie je manier van programmeren?
- Welke taken kan AI goed overnemen en welke blijven menselijk vakmanschap?
- Hoe zorg je ervoor dat je conceptueel begrip behoudt terwijl je AI gebruikt?
- Wat zijn de ethische implicaties van het gebruik van AI-gegenereerde code?
- Hoe zie je de toekomst van programmeren met AI-assistentie?