Terug naar de Universiteit

Duiken in Wetenschappelijk Programmeren

Willem L. Middelkoop
8 feb. 2023

Deze maand ging ik terug naar de universiteit voor een speciale cursus wetenschappelijk programmeren. Met de vele recente ontwikkelingen in machine learning (in PowerPoint-taal "AI" genoemd), waaronder ChatGPT, voelde het als het ideale moment om me te verdiepen in Python programmeren. Ga met me mee op deze reis.

Waarom Python voor wetenschappelijk programmeren?

Python heeft enorme populariteit verworven binnen de wetenschappelijke gemeenschap, en terecht. De eenvoud en leesbaarheid maken het toegankelijk voor nieuwkomers, terwijl het de kracht behoudt om complexe taken aan te pakken. De veelzijdigheid van de taal stelt onderzoekers en wetenschappers in staat om algoritmen en modellen relatief eenvoudig te implementeren.

Een uitgebreid scala aan bibliotheken die zijn afgestemd op wetenschappelijk computergebruik, zoals NumPy, SciPy, Pandas en Matplotlib, versterkt Pythons positie in dit domein. Deze bibliotheken bieden uitgebreide functies en tools voor data-analyse, statistische modellering en visualisatie, waardoor het wetenschappelijk programmeringsproces wordt gestroomlijnd.

In vergelijking met andere programmeertalen, zoals JavaScript of Ruby, die vaak worden gebruikt in niet-wetenschappelijke contexten zoals webontwikkeling, heeft Python een eenvoudigere syntaxis. Deze eigenschap stelt gebruikers in staat zich te concentreren op het probleem in kwestie in plaats van te worstelen met taalcomplexiteiten. Bovendien maakt Pythons uitgebreide wetenschappelijke ecosysteem het een geschiktere keuze voor onderzoekstoepassingen.

Bekijk de volgende twee codevoorbeelden, één in C++ en de andere in Python 3. Elk programma voert identieke taken uit: het opvragen van de naam en geboortedatum van de gebruiker, het uitvoeren van basisinvoervalidatie en uiteindelijk het onthullen van de huidige leeftijd van de gebruiker.

```cpp
#include <iostream>
#include <string>
#include <sstream>
#include <ctime>

bool validate_date(const std::string &date_string, int &day, int &month, int &year) {
    std::istringstream iss(date_string);
    char delimiter;
    if (iss >> day >> delimiter >> month >> delimiter >> year) {
        return true;
    }
    return false;
}

int calculate_age(int day, int month, int year) {
    time_t now = time(0);
    tm *ltm = localtime(&now);

    int current_year = 1900 + ltm->tm_year;
    int current_month = 1 + ltm->tm_mon;
    int current_day = ltm->tm_mday;

    int age = current_year - year - ((current_month < month) || (current_month == month && current_day
< day));
    return age;
}

int main() {
    std::string name, dob_string;
    int day, month, year;

    std::cout << "Please enter your name: ";
    std::getline(std::cin, name);

    do {
        std::cout << "Please enter your date of birth (dd-mm-yyyy): ";
        std::getline(std::cin, dob_string);

        if (!validate_date(dob_string, day, month, year)) {
            std::cout << "Invalid date format. Please try again." << std::endl;
        }
    } while (!validate_date(dob_string, day, month, year));

    int age = calculate_age(day, month, year);
    std::cout << name << ", your age is " << age << " years." << std::endl;

    return 0;
}
```

C++ voorbeeldcode: Bepalen van de leeftijd van de gebruiker

```python
from datetime import datetime

def get_user_input(prompt):
    while True:
        user_input = input(prompt).strip()
        if user_input:
            return user_input

def validate_date(date_string):
    try:
        date_obj = datetime.strptime(date_string, '%d-%m-%Y')
        return date_obj
    except ValueError:
        return None

def calculate_age(dob):
    today = datetime.now()
    age = today.year - dob.year - ((today.month, today.day) < (dob.month,
dob.day))
    return age

name = get_user_input("Please enter your name: ")
dob_string = ""

while not dob_string:
    dob_string = get_user_input("Please enter your date of birth (dd-mm-yyyy): ")
    dob = validate_date(dob_string)
    if not dob:
        dob_string = ""
        print("Invalid date format. Please try again.")

age = calculate_age(dob)
print(f"{name}, your age is {age} years.")
```

Python 3 voorbeeldcode: Bepalen van de leeftijd van de gebruiker

Men kan gemakkelijk de contrasterende syntaxis tussen de twee talen waarnemen, waarbij Python meer lijkt op alledaags Engels. Dit aspect van Python verbetert vaak de leesbaarheid en het begrip voor veel gebruikers.

Een andere factor die bijdraagt aan het succes van Python in wetenschappelijk programmeren is de bloeiende gemeenschap. Onderzoekers en ontwikkelaars uit verschillende disciplines delen kennis, bieden ondersteuning en dragen bij aan open-sourceprojecten, waardoor een omgeving van samenwerking en continu leren wordt bevorderd.

De Cursus

Gedurende de cursus werden de absolute basisprincipes van programmeren verkend door problemen uit verschillende wetenschappelijke domeinen aan te pakken. Na voltooiing werd een solide begrip van programmeerprincipes verworven, met het vermogen om deze toe te passen in verschillende vakgebieden en projecten.

De Universiteit van Amsterdam op de Science Park campus - vrienden van de blog zullen mijn fiets herkennen

Het herhalen van de basisprincipes van programmeren, zelfs voor ervaren programmeurs (zoals ikzelf), kan enorm nuttig zijn. Na verloop van tijd is het niet ongebruikelijk dat programmeurs gewoonten of snelkoppelingen ontwikkelen die mogelijk niet voldoen aan de beste praktijken. Door terug te keren naar de basis, kun je eventuele schadelijke gewoonten afleren en een sterke basis in programmeerprincipes herstellen. Bovendien betekent de snelle aard van de technologie-industrie dat er voortdurend innovaties en updates opduiken. Door de basis te herhalen, kunnen ervaren programmeurs op de hoogte blijven van de nieuwste ontwikkelingen en methodologieën. Dit proces van herevaluatie biedt ook een mogelijkheid om programmeren met een schone lei te benaderen.

[image]

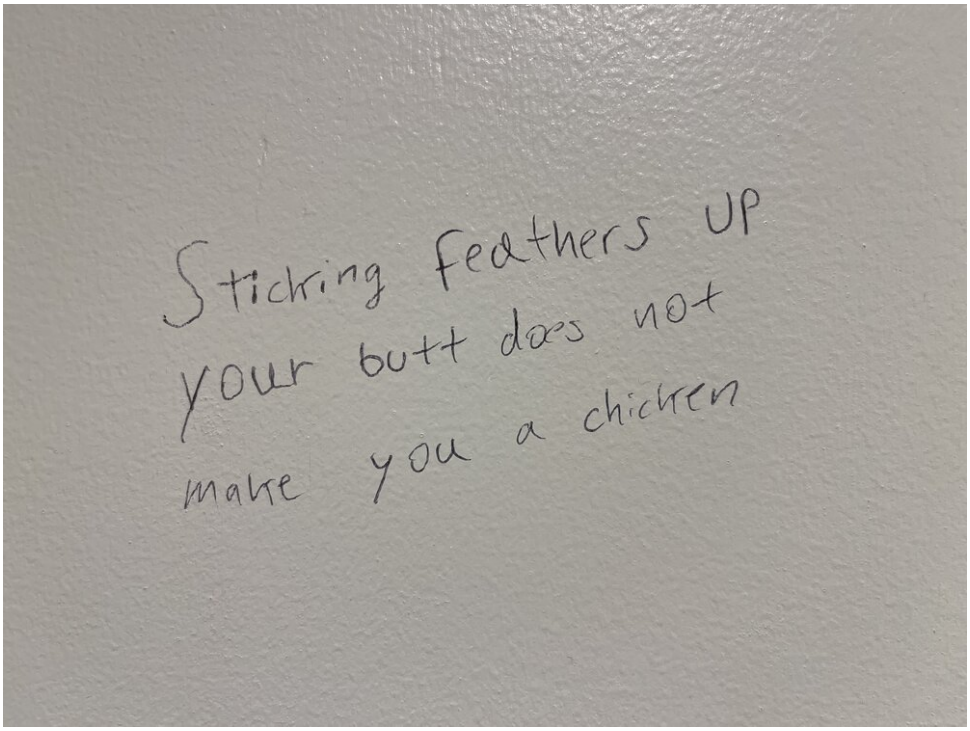
De cursus bestond uit meerdere modules, beginnend met Niveau 1, waar een keuze werd gemaakt tussen ALGORITMES, die zich richtte op het opsplitsen van intuïtieve problemen in stappen die een computer kon begrijpen, en GETALLEN, een wiskundig georiënteerde module die zich verdiepte in getaleigenschappen zonder dat voorkennis van wiskunde vereist was. In Niveau 2 was de keuze tussen TEKST, die zich concentreerde op natuurlijke taalverwerking en sentimentanalyse van tweets, en NUMERIEKE INTEGRATIE, waar belangrijke technieken werden aangeleerd voor het bepalen van oppervlakten onder functies wanneer traditionele integratie niet volstond. Niveau 3 omvatte het werken met BIG-DATA, het analyseren van grote datasets en weerpatronen in Nederland. Een optioneel bonusniveau, BEWEGING, bood een simulatie van het graven van een tunnel door de planeet en verkende natuurkundige problemen die met computerondersteuning oplosbaar waren.

Ik neem deel aan een programmeerles op de Universiteit (klasgenoten geblurd om privacyredenen)

[image]

Conclusie

Ongeacht iemands leeftijd of wijsheid, er is altijd ruimte om iets nieuws te leren, vooral met de vooruitgang in machine learning zoals GPT4. Bovendien biedt teruggaan naar de universiteit een verfrissende verandering, die een volledig andere context presenteert in vergelijking met de commerciële of professionele sfeer waarin ik normaal gesproken opereer. Omarm de mantra: blijf hongerig, blijf dwaas!



De wekelijkse lab sessies bieden een heerlijke mogelijkheid om je codeeravonturen op de rails te houden, zodat je altijd een helpende hand van anderen hebt wanneer je vastloopt.