python scripts

- New Page
- New Page
- New Page

New Page

```
"""
Praktijkleren Examenopdracht – Python
Onderwerp: Kentekencheck automatiseren
Auteur: <jouw naam>
Datum: <datum>


Dit script automatiseert het opzoeken van voertuiggegevens
op basis van een kenteken via RDW Open Data.
"""


import requests



def vraag_kenteken():
    """
    Vraagt maximaal 3 keer om een geldig kenteken.
    Retourneert het kenteken of None.
    """
    pogingen = 0  # variabele


    while pogingen < 3:  # lus met limiet
        kenteken = input("Voer een kenteken in (6 tekens): ")
        kenteken = kenteken.upper().replace("-", "").strip()


        if len(kenteken) == 6:  # beslissing
            return kenteken
        else:
            pogingen += 1
            print(f"Ongeldig kenteken. Poging {pogingen} van 3.")


    print("Te veel foutieve pogingen. Programma stopt.")
    return None
```

```
def haal_voertuiggegevens_op(kenteken):
    """
    Haalt voertuiggegevens op via RDW Open Data.
    """
    url = "https://opendata.rdw.nl/resource/m9d7-ebf2.json"
    params = {"kenteken": kenteken}


    try:
        response = requests.get(url, params=params, timeout=10)
        response.raise_for_status()
        return response.json()
    except requests.RequestException:
        return None



def toon_gegevens(data):
    """
    Toont voertuiggegevens op een leesbare manier.
    """
    if not data:  # beslissing
        print("Geen voertuiggegevens gevonden.")
        return


    voertuig = data[0]
    print("\nVoertuiggegevens:")
    print("Kenteken:", voertuig.get("kenteken"))
    print("Merk:", voertuig.get("merk"))
    print("Type:", voertuig.get("handelsbenaming"))
    print("Voertuigsoort:", voertuig.get("voertuigsoort"))
    print("Eerste toelating:", voertuig.get("datum_eerste_toelating"))



def main():
    """
    Hoofdprogramma
    """
    doorgaan = True  # variabele


    while doorgaan:  # lus
```

```
        kenteken = vraag_kenteken()
        if kenteken is None:
            break  # stoppen bij te veel fouten


        gegevens = haal_voertuiggegevens_op(kenteken)
        toon_gegevens(gegevens)


        keuze = input("\nNog een kenteken controleren? (j/n): ")
        if keuze.lower() != "j":  # beslissing
            doorgaan = False


    print("Programma afgesloten.")



main()
```

New Page

```
"""
Praktijkleren Examenopdracht – Python
Onderwerp: Subnetcalculator
Auteur: <jouw naam>
Datum: <datum>


Dit script automatiseert het berekenen van netwerkgegevens
op basis van een IP-adres en subnetmasker.
"""


def vraag_ip_adres():
    """
    Vraagt maximaal 3 keer om een geldig IPv4-adres.
    """
    pogingen = 0


    while pogingen < 3:
        ip = input("Voer een IP-adres in (bijv. 192.168.1.10): ")
        delen = ip.split(".")


        if len(delen) == 4 and all(d.isdigit() and 0 <= int(d) <= 255 for d in delen):
            return ip
        else:
            pogingen += 1
            print(f"Ongeldig IP-adres. Poging {pogingen} van 3.")


    print("Te veel foutieve pogingen.")
    return None



def vraag_subnetmasker():
    """
    Vraagt een subnetmasker in CIDR-notatie.
    """
```

```
    masker = input("Voer subnetmasker in (CIDR, bijv. 24): ")


    if masker.isdigit() and 0 <= int(masker) <= 32:
        return int(masker)
    else:
        print("Ongeldig subnetmasker.")
        return None



def bereken_hosts(cidr):
    """
    Berekent het aantal hosts op basis van CIDR.
    """
    host_bits = 32 - cidr
    hosts = (2 ** host_bits) - 2
    return hosts



def main():
    """
    Hoofdprogramma
    """
    ip = vraag_ip_adres()
    if ip is None:
        return


    cidr = vraag_subnetmasker()
    if cidr is None:
        return


    hosts = bereken_hosts(cidr)


    print("\nResultaat:")
    print("IP-adres:", ip)
    print("Subnetmasker: /" + str(cidr))
    print("Aantal mogelijke hosts:", hosts)



main()
```

New Page

```
"""
Praktijkleren Examenopdracht – Python
Onderwerp: Logbestand analyseren
Auteur: <jouw naam>
Datum: <datum>


Dit script analyseert een logbestand en telt het aantal foutmeldingen.
"""


def lees_logbestand(bestandsnaam):
    """
    Leest een logbestand en geeft de regels terug als een lijst.
    """
    try:
        with open(bestandsnaam, "r") as bestand:
            regels = bestand.readlines()
            return regels
    except FileNotFoundError:
        print("Bestand niet gevonden.")
        return None



def tel_fouten(regels):
    """
    Telt het aantal regels waarin het woord 'ERROR' voorkomt.
    """
    fout_teller = 0  # variabele


    for regel in regels:  # lus
        if "ERROR" in regel:  # beslissing
            fout_teller += 1


    return fout_teller
```

```
def main():
    """
    Hoofdprogramma
    """
    bestandsnaam = "logbestand.txt"  # variabele


    regels = lees_logbestand(bestandsnaam)
    if regels is None:
        return


    fouten = tel_fouten(regels)


    print("Analyse resultaat:")
    print("Aantal regels in bestand:", len(regels))
    print("Aantal foutmeldingen:", fouten)



main()
```