Introductie Python en Linux - Introductiedag Informatiekunde

I. Linux terminal commando's

- Start de Linux terminal
- Voer na de Linux terminal prompt (**$**) de volgende commando's in om een georganiseerde file-structure te beginnen:
    1. Maak een nieuwe directory (map) aan met `$ mkdir <directory naam>`
    2. Navigeer naar deze map met `$ cd <directory naam>`
    3. Maak in deze map een tekstbestand aan met `$ nano myfile.txt`
       Dit opent de text editor Nano, een minimalistische text editor die standaard op Ubuntu is geïnstalleerd.
    4. Typ hier ongeveer 3 regels aan tekst. Druk op **Enter** ⏎ om naar de volgende regel te gaan.
    5. Sla het bestand op met **Ctrl + o** gevolgd door **Enter** ⏎ en sluit nano af met **Ctrl + x**
    6. Print nu je tekst naar de terminal met `$ cat myfile.txt`
    7. Tel het aantal woorden en regels in het bestand met `$ wc myfile.txt`
       wc laat het aantal regels, woorden en karakters zien in het bestand. Controleer of de aantallen kloppen.
    8. Veel commando's hebben een gebruiksaanwijzing die je kan vinden met `$ man <command naam>`. Bekijk de manual van wc met `$ man wc`
       Sluit dit venster af door **q** in te toetsen.
    9. Je kan vorige commands terugvinden door te scrollen met de pijltjes. Gebruik het pijltje omhoog om `$ cat myfile.txt` terug te vinden.
    10. Zoek een specifiek woord uit het bestand terug met
        `$ grep --color -i -w <zoekwoord> myfile.txt`
    11. Gebruik nu `$ man grep` om uit te vinden wat de opties `-i` en `-w` deden met het grep commando.
    12. Maak een nieuw tekstbestand aan in dezelfde map, en vergeet niet om dit bestand een andere naam te geven. Zet een woord in dit bestand en sla dit op. Ga nu weer terug naar de terminal
    13. Gebruik nu `$ ls` om alle bestanden in de huidige map weer te geven.
    14. Om bestanden te verwijderen kan je `$ rm <filename>` gebruiken. Verwijder het bestand dat je zojuist gemaakt hebt, maar stop halverwege de bestandsnaam met typen en druk op **Tab** ⇄. Tab ⇄ is de autocomplete van de Linux terminal! Tab ⇄ kan je gebruiken om commands en lange bestandsnamen aan te vullen.
    15. Gebruik `$ cd` zonder directory naam als argument om terug te keren naar de home directory. Sluit de terminal af met **Ctrl + d**.

II. Python commando's in de interactive Python shell

- Start de Linux terminal
- Start de interactive Python shell met `$ python3`. Hiermee wordt de interpreter van Python 3 gestart.
- Het is gelukt als de Linux terminal prompt (**$**) verandert in de prompt van de Python shell (**>>>**)
- Voer de volgende commando's in, en noteer de output die je krijgt. Probeer foutmeldingen te begrijpen, deze helpen je in het begrijpen van wat er precies gebeurt.

```
>>> 12 + 12
>>> “12” + “12”
>>> ‘12’ + “12”
>>> ‘12’ + ‘12’
>>> “12 + 12”
>>> “12” + 12
>>> “2” * “5”
>>> “2” * 5
>>> 2 * 5

>>> print(12)
>>> print(“12”)
>>> print(12 + 12)
>>> print(“De directory’s!”)
>>> print(‘De directory’s!’)
>>> print(‘De directory\’s!’)
>>> print(“De directory\’s!”)
>>> print(‘up\down’)
>>> print(‘up\\down’)
>>> print(‘up\\\\down’)

>>> var1 = 2
>>> var2 = “5”
>>> var1 + var2
>>> var1 + int(var2)
>>> str(var1) + var2
>>> var1 = var2
>>> var1 + var2

>>> movie = “Shrek”
>>> movie + “is een film.”
>>> movie + “ is een film.”
>>> print(movie, “is een film.”)
>>> movie = “Madagascar”
>>> “Ik vind “ + movie + “ niet leuk.”
```

```
>>> question = “Vind jij “ + movie + “ leuk ?”
```
>>> answer = input(question) # typ ‘nee’ en druk op Enter ⏎
```
>>> answer
>>> movie = “Shrek”
>>> question = “Vind jij “ + movie + “ leuk ?”
```
>>> answer = input(question) # typ ‘ja!’ en druk op Enter ⏎
```
>>> answer

>>> print(“100 + 200 = “, 110 + 200)
>>> print(10 * 20)
>>> print(10 ** 20)
>>> print(7/3)
>>> print(7//3)
>>> print(10//3)

>>> def hello_world():
...     print(“Hello World!”) # Let goed op de indentatie!
...     print(“Dit is mijn eerste Python functie!”)
...     print(“Functies zijn ideaal om blokken code vaker uit te
   voeren.”)
...
>>> hello_world()
>>> hello_world()

>>> def add(item1, item2):
...         print(item1 + item2)
...
>>> add(“Python”, “3”)
>>> add(“1”, “2”)
>>> add(1, 2)

>>> def compose():
...     word1 = input(“Vul een woord in: “)
...     word2 = input(“Vul nog een woord in: “)
...     comp1 = word1 + word2
...     comp2 = word2 + word1
...     print(f”De samenstellingen zijn {comp1} en {comp2}”)
...
>>> compose()
```

```
>>> def add():
...     x = input(“Vul een nummer in: “)
...     y = input(“Vul nog een nummer in: “)
...     z = x + y
...     print(f“De som van {x} en {y} is {z}.”)
...
>>> add()
```

Je krijgt hier waarschijnlijk niet de output die je verwacht. Hoe kan je dit oplossen?

- Verlaat de Python shell met `>>> exit()`

III. Tab indentation in nano aanpassen

- Maak of bewerk het .nanorc bestand met `$ nano ~/.nanorc`
- Voeg de volgende regel aan het bestand toe: `set tabsize 4`
- Sla het bestand op en restart alle instanties van nano.

IV. Python scripts opslaan

- Maak een nieuw bestand aan met nano en noem het hello_world.py
- In het bestand schrijf je de volgende code:

```
def hello_world():
    name = input(“Vul je naam in: “)
    print(f“Hello World!\nThis is {name}!”)

hello_world()
```

- Sla het bestand op en ga terug naar de terminal
- In de terminal kan je de Python3 interpreter gebruiken om bestanden te interpreteren als python, bestanden als hello_world.py.
- Run het bestand hello_world.py als Python script met het Linux commando `$ python3 hello_world.py` (zorg dat je in de goede directory zit)
- Ga nu terug naar het bestand en pas de code aan naar het volgende:

```
def hello_world(first_name, last_name):
    print(f“Hello World! This is {first_name} {last_name}!”)

hello_world(“Guido”, “van Rossum”)
hello_world(“Bill”, “Gates”)
```

- Sla de code op en run deze opnieuw als python script.
- Sluit de terminal nu helemaal af
- Als je de terminal nu opnieuw opstart en naar dezelfde directory gaat als het hello_world.py bestand kan je deze gewoon weer opnieuw gebruiken!