Oefententamen Programmeermethoden NA
Universiteit Leiden — Informatica

Het tentamen bestaat uit 4 opgaven verdeeld over 3 pagina's. De duur van het tentamen is 3 uur. De te behalen punten (totaal 100) staan tussen haakjes bij de opgaven.

Bij alle functies moeten de variabelen (constanten eventueel uitgezonderd) als parameter of lokale variabele voorkomen. Gebruik `____` om één niveau van inspringen aan te duiden. Twee keer dit karakter betekent dus twee keer inspringen. Denk aan de dubbele punten!

Veel succes!

---

1. (35 punten) In een lijst `A` staan `len(A)` (gegarandeerd $\geq$ `1`) gehele getallen > 0.

a. (4) Schrijf een Python-functie `hoe(A,X)` die bepaalt hoe vaak het gehele getal `X` in de lijst `A` voorkomt en dit als returnwaarde teruggeeft.

b. (4) Schrijf een Python-functie `fac(k)` die met behulp van één `for`-loop de waarde van $\texttt{k}! = \texttt{k} \cdot (\texttt{k}-1) \ldots 2 \cdot 1$ uitrekent en dit als returnwaarde teruggeeft. Neem aan dat het gehele getal `k` groter dan 0 is. Gebruik *geen* recursie.

c. (6) Schrijf een Booleaanse Python-functie `controle(A)` die het volgende controleert voor de lijst `A`. In de elementen met even lijst-index > 0 moet een getal staan dat hooguit index keer voorkomt, en in de elementen met oneven lijst-index moet de faculteit van de index staan. (Bijvoorbeeld, met `len(A)` > 5: als `A[4]` 7 is, mag 7 hooguit 4 keer voorkomen, en `A[5]` moet $5! = 120$ bevatten.) Als dit klopt, geeft de functie `True`, en anders `False`. De functie moet stoppen zodra het antwoord bekend is. Gebruik **a** en **b**.

d. (7) Schrijf een Python-functie `busort(A)` die de lijst `A` met behulp van *bubblesort oplopend* sorteert, waarbij het algoritme stopt zodra er in een ronde geen verwisselingen waren.

e. (4) Hoeveel vergelijkingen tussen lijst-elementen doet de methode van **d** minimaal en maximaal, uitgedrukt in de lengte van de lijst (`n = len(A)`)? Geef voor beide situaties een rijtje getallen dat dit aantal realiseert.

f. (10) Schrijf een Python-functie `langste(A)` die de lengte uitrekent van een langste direct opeenvolgende serie gelijke getallen in de lijst `A`. De functie geeft als returnwaarde zowel de lengte van de gevonden serie als de lijst-index van het laatste element van een dergelijke serie; als er meer van dergelijke series zijn: degene met de hoogste index. De functie wordt aangeroepen als volgt: `lengte, laatste = langste(A)`. Voor de lijst `1 7 1 1 4 7 7 3` zou het antwoord `(2, 6)` zijn: `lengte` moet `2` worden en `laatste` `6`.

2. (20 punten)

a. (4) Bij een functie kun je te maken hebben met *locale* en *globale* variabelen. Verder onderscheiden we *formele* en *actuele* parameters. Leg deze vier begrippen duidelijk uit.

b. (6) Gegeven een Python-programma met daarin de volgende twee functies:

```
def gerard(x, y):
    global u
    i = 42
    u += 1
    for i in range(1, int(y+1)):
        x = x + y
        print x
    y = 0
    return x
```

```
def harry(v, w):
    global u
    if v < w:
        w = gerard(v, w)
    else:
        w = gerard(w, v)
    print w
    return v + w
    u += 1
```

Verder zijn de globale variabelen `s`, `t` en `u` gegeven (`s` en `t` van type `float`, `u` van type `int`). Wat is dan de uitvoer van het volgende stukje programma (leg je antwoord duidelijk uit):

```
s, t, u = 2.1, 3.1, 41
print harry(s, t)
print "{0}, {1}, {2}".format(s, t, u)
```

c. (5) Als we in de functie `gerard` de regel `global u` weghalen, runt het programma dan nog? En hoe zit dat voor de functie `harry`?

d. (5) Geef een eenvoudige uitdrukking in `x` en `y` voor de waarde die `gerard(x,y)` retourneert.

3. (10 punten) Gegeven een NumPy array `M` zoals hiernaast weergegeven. Schrijf voor de hieronder gegeven slices van `M` de juiste expressie die resulteert in die slice.

```
array([[14, 15, 16, 17],
       [18, 19, 20, 21],
       [22, 23, 24, 25],
       [26, 27, 28, 29],
       [30, 31, 32, 33],
       [34, 35, 36, 37]])
```

a. (2) `array([30, 31, 32, 33])`

b. (2) `array([28, 32, 36])`

c. (2)
```
array([[28, 29],
       [32, 33],
       [36, 37]])
```

d. (2)
```
array([[14, 15, 16, 17],
       [22, 23, 24, 25],
       [30, 31, 32, 33]])
```

e. (2) Geef de NumPy-expressie die een 1-dimensionale array oplevert met daarin de som voor elke rij van `M`. Resultaat: `array([ 62,  78,  94, 110, 126, 142])`.

4. (35 punten) Gegeven is een `m` bij `n` (beide > 0) NumPy array `H` met gehele getallen ≥ 0. Hierbij geeft `H[i,j]` de hoogte ter plekke (i, j) aan (met $0 \leq i < m$ en $0 \leq j < n$). Hiernaast staat een voorbeeld met $m = 4$ en $n = 5$. De variabelen `m` en `n` zijn als globale "constantes" gedefineerd en hoeven bij deze opgave niet doorgegeven te worden als parameter.

3	0	4	2	0
7	2	4	9	6
1	2	0	5	1
6	9	3	1	0

a. (8) Schrijf een Python-functie `aflagen(H,i,j,p,q)` die alle array-elementen > 0 in het array `H` in de rechthoek met linksboven (i, j) en rechtsonder (p, q), met één aflaagt. Neem aan dat $0 \leq i < p < m$ en $0 \leq j < q < n$. (Met `aflagen(H,0,0,m-1,n-1)` wordt het hele array behandeld.) Vergeet niet de functie-parameters in te vullen!

b. (8) Schrijf een Python-functie `gemiddeld(H,i,j)` die het gemiddelde van de vier (aan de randen drie of twee) hoogtes van direct horizontaal of verticaal aangrenzende buren van (i, j) berekent en deze waarde als integer teruggeeft. Neem aan dat $0 \leq i < m$ en $0 \leq j < n$. Het gemiddelde moet op de gebruikelijk manier worden afgerond; zo moet `gemiddeld(H,2,4)` in het voorbeeld $(6 + 5 + 0)/3 = 11/3 \to 4$ opleveren.

c. (9) Schrijf een Python-functie `maximum(H)` die het grootste getal uit `H` bepaalt, en vier returnwaarden teruggeeft: het grootste getal, het aantal keren dat dit maximum voorkomt en de coördinaten (rij, kolom) van het grootste getal. De functie wordt aangeroepen als `grootste, aantal, i0, j0 = maximum(H)`. In het voorbeeld zou het antwoord 9 zijn (met `aantal` gelijk aan 2), waarbij `i0` bijvoorbeeld gelijk aan 3 wordt en `j0` gelijk aan 1 (of 1 respectievelijk 3: als het maximum meerdere keren voorkomt, mag je zelf kiezen).

d. (10) Schrijf een Python-functie `egaliseer (H,drempel)` die het array `H` als volgt "egaliseert". Trek telkens van het (of beter: een) grootste getal van het array 1 af, totdat het maximum minstens `drempel` (een gegeven geheel getal > 0 en $\leq m \cdot n$) keer voorkomt. Het aantal benodigde stappen moet worden teruggegeven (type `int`). In het voorbeeld, met `drempel` gelijk aan 4, is dat 7 keer: dan zijn de 9, 9 en 7 alle drie 6 geworden; het maximum 6 komt dan trouwens 5 keer voor. Gebruik **c**.