ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ
ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΠΟΛΙΤΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
ΤΟΜΕΑΣ ΔΟΜΟΣΤΑΤΙΚΗΣ

Ακαδ. Έτος: 2021-2022

Πέμπτη, 2/6/2022

Μάθημα: Προγραμματισμός Η/Υ

Διδάσκοντες: Ν.Δ. Λαγαρός (Καθηγητής), Α. Στάμος (ΕΔΙΠ), Χ. Φραγκουδάκης (ΕΔΙΠ)
Αμβ. Σαββίδης (Δρ)

Παραδείγματα για την 10η παράδοση – Εισαγωγή στο web programming

1. Υπολογισμός οπλισμού δοκού

Το μηχανικό ποσοστό οπλισμού ω δοκού δίνεται σε σχέση με την ανηγμένη ροπή μ_{sd}:

$$\omega = 0.84\left(1-\sqrt{1-2.4\,\mu_{sd}}\right)$$

Να συνταχθεί πρόγραμμα σε python το οποίο να εμφανίζει σε ιστοσελίδα:
α. Πίνακα οπλισμού ω/μ_{sd} για ω=0, 0.01, ..., 0.35
β. Το αντίστοιχο διάγραμμα

Λύση

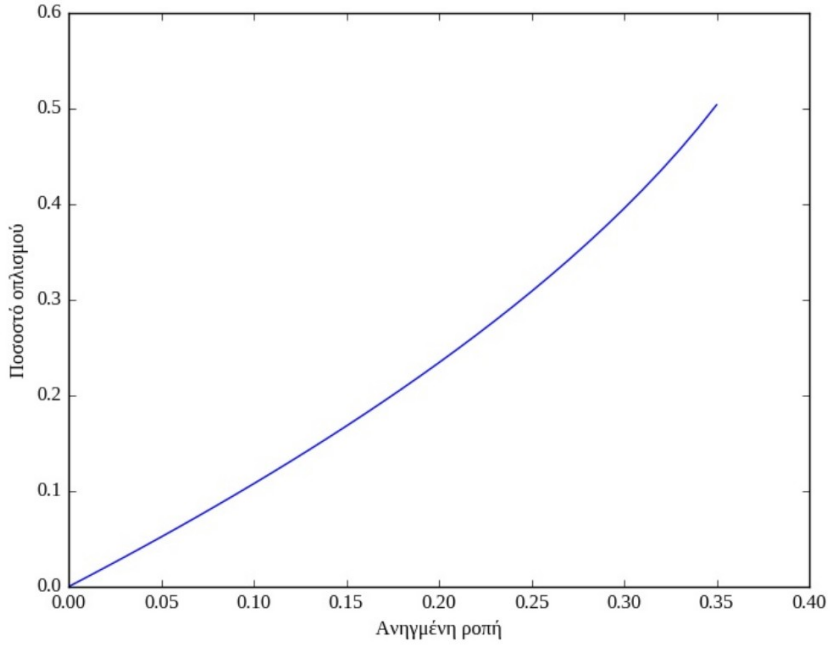
Αρχικά θα υπολογιστούν οι αριθμητικές τιμές των ω και μ_{sd}. Στη συνέχεια σε ένα πίνακα με 1 γραμμή και δύο στήλες θα τοποθετηθεί στην πρώτη στήλη ο πίνακας οπλισμού και στη δεύτερη το διάγραμμα, δηλαδή θα τοποθετηθεί πίνακας μέσα σε πίνακα.
Το διάγραμμα θα αποθηκευτεί σε αρχείο temp.jpg και αυτό θα εμφανιστεί στην ιστοσελίδα. Για λόγους ασφαλείας, η βιβλιοθήκη bottle επιτρέπει άνοιγμα αρχείων μόνο σε ένα φάκελλο που ορίζεται μέσω της συνάρτησης static_file().

```
from pathlib import Path
import numpy as np
from matplotlib import pyplot as plt
import matplotlib
from bottle import route, run, static_file

@route("/static/<fn>")
def server_files(fn):
    "The static files are in current directory."
    return static_file(fn, root=str(Path.cwd()))

@route('/')
def showMoments():
    "Make and show table and diagram of beam reinforcement."
    mu = np.arange(0, 0.36, 0.01)
    w = 0.84*(1-np.sqrt(1-2.4*mu))
    matplotlib.rc('font', family='serif')
    plt.figure()
    plt.plot(mu, w)
    plt.xlabel("Ανηγμένη ροπή")
    plt.ylabel("Ποσοστό οπλισμού")
    plt.savefig("temp.jpg")

    html = []
    html.append("<html>")
    html.append("<head><title>Οπλισμός δοκού</title></head>")
    html.append("<body>")
    html.append("""<table border=2 align="left">""")
    html.append("<tr>")
```

```
    html.append("<td>")
    html.extend(table_numbers(mu, w))
    html.append("</td>")

    html.append("<td>")
    html.append("""<img src="/static/temp.jpg" alt="image not found">""")
    html.append("</td>")

    html.append("</tr>")
    html.append("</table>")

    html.append("</body>")
    html.append("</html>")

    html = "\n".join(html)
    print(html)
    return html

def table_numbers(mu, w):
    "Return the table in html format."
    t = []
    t.append("""<table border=1 align="left" style="font-size:0.8em">""")
    t.append("<tr>")
    t.append("<th><div>Ανηγμένη</div><div>ροπή</div></th>")
    t.append("<th><div>Ποσοστό</div><div>οπλισμού</div></th>")
    t.append("</tr>")
    for mu1, w1 in zip(mu, w):
        t.append("<tr>")
        t.append("""<td align="center">{:.3f}</td>"""
            """<td align="center">{:.3f}</th>""".format(mu1, w1))
        t.append("</tr>")
    t.append("</table>")
    return t

run(host='localhost', port=8080)
```

Στον εσωτερικό πίνακα η γραμματοσειρά ορίστηκε με μέγεθος 0.8em, που σημαίνει το εξορισμού (default) μέγεθος της γραμματοσειράς επί 0.8, έτσι ώστε να χωρούν τα δεδομένα σε μία σελίδα. Το πρόγραμμα δημιουργεί την ιστοσελίδα localhost:8080/ η οποία φαίνεται παρακάτω:

Ανηγμένη ροπή	Ποσοστό οπλισμού
0.000	0.000
0.010	0.010
0.020	0.020
0.030	0.031
0.040	0.041
0.050	0.052
0.060	0.063
0.070	0.074
0.080	0.085
0.090	0.096
0.100	0.108
0.110	0.119
0.120	0.131
0.130	0.143
0.140	0.156
0.150	0.168
0.160	0.181
0.170	0.194
0.180	0.207
0.190	0.220
0.200	0.234
0.210	0.248
0.220	0.263
0.230	0.278
0.240	0.293
0.250	0.309
0.260	0.325
0.270	0.342
0.280	0.359
0.290	0.377
0.300	0.396
0.310	0.415
0.320	0.435
0.330	0.457
0.340	0.480
0.350	0.504



2. Υπολογισμός θερμοπερατότητας πολυστρωματικού επιφανειακού στοιχείου

Ο συντελεστής θερμοπερατότητας U πολυστρωματικού επιφανειακού στοιχείου που αποτελείται από n στρώσεις θερμικής αγωγιμότητας λ_i και πάχους d_i δίνεται από:

$$\frac{1}{U}=\sum_{i=1}^{n}\frac{d_i}{\lambda_i}$$

Να συνταχθεί πρόγραμμα σε python το οποίο να εμφανίζει σε ιστοσελίδα κατάλληλα πεδία στα οποία ο χρήστης μπορεί να εισάγει την θερμική αγωγιμότητα και το πάχος μέχρι και 9 στρώσεων. Στη συνέχεια με το πάτημα κομβίου να γίνεται ο υπολογισμός του συντελεστή θερμοπερατότητας U.

Δοκιμάστε το πρόγραμμά σας σε μπατική τοιχοποιία χωρίς μόνωση η οποία αποτελείται από 2 στρώσεις οπτόπλινθων πάχους 0.09cm αγωγιμότητας 0.64 W/(mK) και δύο στρώσεις σοβά πάχους 0.01 m με αγωγιμότητα 0.87 w/(mK).

Λύση

Αρχικά θα γίνει συνάρτηση που δημιουργεί φόρμα (form) με input widgets που δέχονται αριθμούς για την αγωγιμότητα και το πάχος στρώσης. Αυτά θα επαναληφθούν 9 φορές. Στο τέλος θα γίνει input widget για submit, που εμφανίζει σχετικό κομβίο. Στους αριθμούς θα μπουν όρια min=0.0001, max=100 και step=0.0001 (το step πρέπει να είναι τόσο γιατί αλλιώς το input δεν αφήνει να δώσουμε αυθαίρετο αριθμό). Ο χρήστης μπορεί να μην δώσει τιμές στις τελευταίες n στρώσεις (n<9). Πρέπει να γίνονται οι έλεγχοι:

α. Ο χρήστης πρέπει να δώσει τουλάχιστον μία στρώση.

β. Αν σε μία στρώση έχει δοθεί η αγωγιμότητα τότε πρέπει να δοθεί και το πάχος, και αντίστροφα.
γ. Αν έχει δοθεί μία στρώση k τότε πρέπει να έχουν δοθεί και οι προηγούμενες k-1.
Αν ο χρήστης έχει κάνει κάποιο λάθος τότε πρέπει να εμφανιστεί αυτό το λάθος και να εμφανιστεί πάλι η φόρμα με συμπληρωμένα τα στοιχεία που έχει ήδη δώσει ο χρήστης, για να τα διορθώσει. Αν ο χρήστης δεν έχει κάνει λάθος πρέπει να υπολογιστεί και να εμφανιστεί η θερμοπερατότητα και πάλι η φόρμα με συμπληρωμένα τα στοιχεία που έχει ήδη δώσει ο χρήστης, σε περίπτωση που θέλει να αλλάξει κάτι.
Τέλος χρήσιμο είναι να υπάρχει τρόπος διαγραφής των παλιών στοιχείων από τη φόρμα, για να δοθούν νέα. Αυτό γίνεται με ένα σύνδεσμο ο οποίος οδηγεί στην ιστοσελίδα "/" (δηλαδή στην ίδια συνάρτηση χωρίς τα ορίσματα).

```
import numpy as np
from bottle import route, run, request

@route('/')
def qerm(lams1=(), ds1=(), U=None, terr=""):
    "Get the data from the user."
    lams = [""]*9
    ds = [""]*9
    n = min(len(lams1), len(lams))
    lams[:n] = lams1[:n]
    n = min(len(ds1), len(ds))
    ds[:n] = ds1[:n]

    html = []
    html.append("<html>")
    html.append("<head><title>Υπολογισμός θερμοπερατότητας</title></head>")
    html.append("<body>")
    html.append("<h2>Υπολογισμός θερμοπερατότητας επιφανειακού στοιχείου</h2>")

    html.append('''<form action="/" method="post">''')
    for i in range(9):
        html.append("<b>Στρώση {:2d}:</b>".format(i+1))
        html.append('''  αγωγιμότητα (W/(m K)) <input name="lam" type="number"
min="0.0001" max="100.0" step="0.0001" value={} />'''.format(lams[i]))
        html.append('''  πάχος (m)             <input name="d"   type="number"
min="0.0001" max="100.0" step="0.0001" value={} />'''.format(ds[i]))
        html.append("<br><br>")
    html.append("<br>")
    html.append('''<a href="/"> Καθαρισμός </a>
        ''')
    html.append('''<input value="Υπολογισμός" type="submit" />''')
    html.append("</form>")
    if U is not None:
        html.append('''<font color="darkgreen">Θερμοπερατότητα U={:.3f} W/(m<sup>2</sup>
K) </font><br><br>'''.format(U))
    if terr != "":
        html.append('''<font color="darkred">''')
        html.append(terr)
        html.append("<br>Διορθώστε και προσπαθείστε πάλι.</font>")

    html.append("</body>")
    html.append("</html>")

    html = "\n".join(html)
    #print(html)
    return html
```

Οι χαρακτήρες   εμφανίζουν τον κενό χαρακτήρα (στην HTML δύο ή περισσότερα κενά είναι ισοδύναμα με 1 κενό). Στη συνέχεια δημιουργείται συνάρτηση η οποία διαβάζει τα δεδομένα από τα widgets, τα ελέγχει και αν είναι σωστά υπολογίζει τη θερμοπερατότητα U και θέτει το μήνυμα λάθους terr="". Αν δεν είναι σωστά, θέτει το μήνυμα λάθους στη μεταβλητή terr και θέτει U=None. Στη συνέχεια καλεί την προηγούμενη συνάρτηση qerm με ορίσματα τα δεδομένα, το αποτέλεσμα και το μήνυμα λάθους. Ας σημειωθεί ότι η βιβλιοθήκη bottle καλεί τη συνάρτηση qerm() χωρίς ορίσματα και για αυτό στα ορίσματα της qerm() έχουν τοποθετηθεί τιμές εξορισμού (default).

```
@route('/', method="POST")
def calc():
```

```
    "Compute the U-value."
    lams = request.forms.getall('lam')
    ds = request.forms.getall('d')
    print(lams)
    print(ds)
    nlams = len(lams)
    while nlams > 0:
        if lams[nlams-1].strip() != "": break
        nlams -= 1
    nds = len(ds)
    while nds > 0:
        if ds[nds-1].strip() != "": break
        nds -= 1
    for i in range(nlams):
        if lams[i].strip() == "":
            return qerm(lams, ds, None, "Δεν δόθηκε η αγωγιμότητα της στρώσης
{}.".format(i+1))
        lams[i] = float(lams[i].replace(",", "."))
    for i in range(nds):
        if ds[i].strip() == "":
            return qerm(lams, ds, None, "Δεν δόθηκε το πάχος της στρώσης
{}.".format(i+1))
        ds[i] = float(ds[i].replace(",", "."))
    if nds != nlams:
        return qerm(lams, ds, None, "Το πλήθος των αγωγιμοτήτων και των παχών πρέπει να
είναι το ίδιο.")
    if nlams == 0:
        return qerm(lams, ds, None, "Δεν δόθηκε καμμία στρώση.")
    s = 0.0
    for i in range(nlams):
        s += ds[i]/lams[i]
    U = 1/s
    return qerm(lams, ds, U, "")

run(host='localhost', port=8080)
```

Το πρόγραμμα δημιουργεί την ιστοσελίδα localhost:8080/ η οποία φαίνεται παρακάτω:

Υπολογισμός θερμοπερατότητας επιφανειακού στοιχείου

Στρώση 1:	αγωγιμότητα (W/(m K))	0,87	πάχος (m)	0,01
Στρώση 2:	αγωγιμότητα (W/(m K))	0,64	πάχος (m)	0,09
Στρώση 3:	αγωγιμότητα (W/(m K))	0,64	πάχος (m)	0,09
Στρώση 4:	αγωγιμότητα (W/(m K))	0,87	πάχος (m)	0,01
Στρώση 5:	αγωγιμότητα (W/(m K))		πάχος (m)	
Στρώση 6:	αγωγιμότητα (W/(m K))		πάχος (m)	
Στρώση 7:	αγωγιμότητα (W/(m K))		πάχος (m)	
Στρώση 8:	αγωγιμότητα (W/(m K))		πάχος (m)	
Στρώση 9:	αγωγιμότητα (W/(m K))		πάχος (m)	

Καθαρισμός Υπολογισμός

Θερμοπερατότητα U=3.287 W/(m^2 K)