ΜΕΜ104 Γλώσσα Προγραμματισμού Ι

5ο φυλλάδιο ασκήσεων

2 Δεκεμβρίου 2020

1. Γράψτε ένα πρόγραμμα το οποίο διαβάζει από τον χρήστη ακέραιους αριθμούς και τους αποθηκεύει στη λίστα L. Στη συνέχεια, κατασκευάζει τις λίστες small και large, με την πρώτη να περιέχει τα στοιχεία της λίστας L τα οποία είναι μικρότερα ή ίσα από τον μέσο όρο των στοιχείων της λίστας L, και τη δεύτερη με τα υπόλοιπα στοιχεία. Τέλος, αποθηκεύει στη μεταβλητή result το μήκος της μεγαλύτερης από τις λίστες large και small.

```
L = eval(input('Enter list of integers: '))
mo = sum(L) / len(L)

small = []
large = []

for i in L:
  if i <= mo:
    small.append(i)
  else:
    large.append(i)
```

Η συμπερίληψη λίστας είναι ιδιαίτερα εύκολη για την κατασκευή των `small` και `large`:

```
small = [i for i in L if i <= mo]
large = [i for i in L if i >  mo]
```

2. Γράψτε ένα πρόγραμμα το οποίο διαβάζει από τον χρήστη μια λίστα L αποτελούμενη από συμβολοσειρές, και στη συνέχεια κατασκευάζει τη λίστα result αποτελούμενη από τον αριθμό των φωνηέντων κάθε συμβολοσειράς στη λίστα L.

```
L = eval(input('Enter list of strings: '))
result = []

for s in L:
  k = 0
  for c in s:
    if c in 'aeiouAEIOU':
      k += 1
  result.append(k)
```

3. Γράψτε ένα πρόγραμμα το οποίο διαβάζει μια λίστα L αποτελούμενη από ακέραιους αριθμούς $k_1, k_2, ..., k_n$ και στη συνέχεια κατασκευάζει τη λίστα result η οποία αποτελείται από συμβολοσειρές $s_1, s_2, ..., s_n$, όπου s_i είναι ο χαρακτήρας '<' ή '>' ή '=', ανάλογα αν το πρώτο ψηφία του k_i είναι μικρότερο από το τελευταίο, μεγαλύτερο από το τελευταίο, ή ίσο με το τελευταίο.

```
L = eval(input('Enter list of integers: '))
result = []
```

```
for n in L:
  first = last = n % 10
  while n > 0:
    last = n % 10
    n //= 10

  if first > last:
    result.append('>')
  elif first == last:
    result.append('=')
  else:
    result.append('<')
```

4. Γράψτε ένα πρόγραμμα το οποίο διαβάζει συμβολοσειρές και τις αποθηκεύει στη λίστα L. Στη συνέχεια κατασκευάζει τη λίστα result η οποία αποτελείται από τις ίδιες συμβολοσειρές αλλά με ανεστραμμένη τη σειρά των χαρακτήρων τους και όπου κάθε φωνήεν έχει αντικατασταθεί από τον χαρακρτήρα 'x'.

```
L = eval(input('Enter list of strings: '))
result = []

for s in L:
  t = ''
  for c in s:
    if c in 'aeiouAEIOU':
      t += 'x'
    else:
      t += c
  result.append( t[::-1] )
```

5. Οι αριθμοί κυκλοφορίας των αυτοκινήτων της πόλης του Ηρακλείου είναι της μορφής 'HZZ-xyzw', όπου x, y, z, w είναι ψηφία. Γράψτε ένα πρόγραμμα το οποίο αποθηκεύει στη λίστα L πινακίδες αυτοκινήτων και στη συνέχεια στη λίστα result εκείνες τις πινακίδες για τις οποίες ο αριθμός xyzw είναι μεγαλύτερος ή ίσος του 1000 και πολλαπλάσιο του 133 ή του 331. Το πρόγραμμά σας θα πρέπει να αγνοεί πινακίδες οι οποίες δεν ξεκινούν με το πρόθεμα 'HZZ'.

```
L = eval(input('Enter list of license plates: '))
result = []

for s in L:
  if s[:3] != 'HZZ':
    continue

  num = int(s[4:])
  if num >= 100 and (num % 133 == 0 or num % 331 == 0):
    result.append(s)
```

6. Γράψτε μια συνάρτηση reciprocalProduct που θα παίρνει όρισμα μια λίστα `L = [x1, x2, ..., xn]` και θα υπολογίζει το γινόμενο `1/x1 * 2/x2 * ... * n/xn`. Γράψτε ένα πρόγραμμα που να διαβάζει μια λίστα και χρησιμοποιεί τη συνάρτηση reciprocalProduct για να υπολογίζει το γινόμενο.

```
def reciprocalProduct(L):
  p = 1
```

```
  for i, x in enumerate(L):
    p *= (i+1) / x
  return p

L = eval(input('Enter list of numbers: '))
result = reciprocalProduct(L)
```

7. Γράψτε μια συνάρτηση occurences που θα παίρνει όρισμα ένα string s και μια λίστα χαρακτήρων letters. Η συνάρτηση θα πρέπει να φτιάχνει και να επιστρέφει μια λίστα που σε κάθε θέση θα έχει το πλήθος των εμφανίσεων του αντίστοιχου χαρακτήρα από την λίστα letters μέσα στο string s. Γράψτε ένα πρόγραμμα που να διαβάζει ένα string και μια λίστα, καλεί με αυτά τη συνάρτηση occurences και αποθηκεύει το αποτέλεσμα στη μεταβλητή result.

```
def occurences(s, letters):
  L = []
  for let in letters:
    L.append( s.count(let) )
  return L

s = input('Enter string: ')
letters = eval(input('Enter list of characters: '))

result = occurences(s, letters)
```

8. Γράψτε μια συνάρτηση positiveFirst που θα παίρνει όρισμα μια λίστα L με θετικούς και αρνητικούς ακέραιους. Η συνάρτηση θα πρέπει να φτιάχνει και να επιστρέφει μια λίστα που θα έχει τους θετικούς ακέραιους ακολουθούμενους από τους αρνητικούς ακέραιους της αρχικής λίστας (με την ίδια σειρά που δίνονται στην L). Γράψτε ένα πρόγραμμα που να διαβάζει μια λίστα, καλεί τη συνάρτηση positiveFirst και αποθηκεύει το αποτέλεσμα στη μεταβλητή result.

```
def positiveFirst(L):
  M = []
  for e in L:
    if e >= 0:
          M.append(e)
  for e in L:
    if e < 0:
          M.append(e)
  return M

L = eval(input('Enter list of integers: '))
result = positiveFirst(L)
```

9. Γράψτε μια συνάρτηση sort3orMore που θα παίρνει όρισμα μια λίστα από λίστες ακεραίων L. Η συνάρτηση θα πρέπει να φτιάχνει και να επιστρέφει μια νέα λίστα με όλες τις λίστες της L, με τη διαφορά οτι οι λίστες με 3 ή περισσότερα στοιχεία θα είναι ταξινομημένες. Γράψτε ένα πρόγραμμα που να διαβάζει μια λίστα, καλεί τη συνάρτηση sort3orMore και αποθηκεύει το αποτέλεσμα στη μεταβλητή result.

```
def sort3orMore(L):
  M = []
  for sublist in L:
        if len(sublist) >= 3:
          M.append( sorted(sublist) )
```

```
        else:
          M.append( sublist )
  return M

L = eval(input('Enter list of lists: '))
result = sort2orMore(L)
```

10. Γράψτε μια συνάρτηση sumLastColumn που θα παίρνει όρισμα μια λίστα από λίστες ακεραίων L που περιγράφει ένα δισδιάστατο πίνακα. Η συνάρτηση θα πρέπει να υπολογίζει και να επιστρέφει το άθροισμα των στοιχείων της τελευταίας στήλης του πίνακα. Γράψτε ένα πρόγραμμα που να διαβάζει μια λίστα, καλεί τη συνάρτηση sumLastColumn και αποθηκεύει το αποτέλεσμα στη μεταβλητή result.

```
def sumLastColumn(L):
  summ = 0
  for sublist in L:
    summ += sublist[-1]
  return summ

L = eval(input('Enter list of lists: '))
result = sumLastColumn(L)
```