Πρόγραμμα επεξεργασίας στοιχείων μαθητών

Θέματα: Επανάληψη, Έλεγχος τιμών, Λίστα, Tuple, Συναρτήσεις

Θέλουμε πρόγραμμα που να διαβάζει ανώνυμα ύψος και ηλικία άγνωστου αριθμού μαθητών και να αποθηκεύει τα ζεύγη των δύο ακέραιων αριθμών σε tuple το οποίο να προστίθεται σε λίστα.

Επειδή το πλήθος των μαθητών είναι άγνωστο, ο αλγόριθμος να σταματά, δίνοντας 9999 ή στο ύψος ή στην ηλικία.

Για αποφυγή λαθών πρέπει να γίνεται έλεγχος εάν το ύψος είναι από 100 μέχρι 250, και η ηλικία από 5 έτη έως 20. Αν δεν τηρούνται αυτά τα κριτήρια, να βγαίνει μήνυμα λάθους και το πρόγραμμα να ζητάει ξανά.

Όταν τελειώσει η εισαγωγή στοιχείων και έχουν αποθηκευτεί σε λίστα τα tuples με τα ύψη και τις ηλικίες των μαθητών, το πρόγραμμα πρέπει να ψάχνει την λίστα και να:

1. υπολογίζει και εμφανίζει μέγιστο ύψος μαθητή,
2. υπολογίζει και εμφανίζει πόσοι είναι 16 ετών και πάνω από 1,80 και
3. υπολογίζει και εμφανίζει τον μέσο όρο ύψους όλων

Να τροποποιηθεί το πρόγραμμα, έτσι ώστε η εισαγωγή στοιχείων και η δημιουργία της λίστας να γίνεται αυτόματα, με τυχαία στοιχεία με την χρήση random generator.

Η γενική δομή του προγράμματος με ψευδοκώδικα υλοποιείται με ένα κλασικό while:

```
δημιουργία κενής λίστας

εισαγωγή ύψους #1

while(ύψος != 9999): #2
     εισαγωγή ηλικίας
     if(ηλικία == 9999):
        break
     προσθήκη στη λίστα
     εισαγωγή ύψους #3

υπολογισμοί και εκτυπώσεις ζητούμενων
```

Η πρώτη υλοποίηση γίνεται χωρίς ελέγχους ορθότητας τιμών. Η επανάληψη βασίζεται στην τιμή της μεταβλητής height και σταματάει όταν το height γίνει 9999. Επειδή ζητείται το πρόγραμμα να σταματάει εναλλακτικά όταν το ίδιο γίνεται με το age, εάν ο χρήστης δώσει 9999 στο age, η τελευταία τιμή που έχει εισαγάγει στο height χάνεται, δεν προστίθεται στη λίστα και η επανάληψη σταματάει μέσω break.

```
#version 1

stu=[]

height=int(input('Height: ')) #1
while(height!=9999): #2
    age=int(input('Age: '))
    if(age==9999):
        break
    stu.append((height, age))
    height=int(input('Height: ')) #3

print(stu)
```

Δείγμα:

```
================= RESTART: C:/v1.py =================
Height: 130
Age: 7
Height: 172
Age: 14
Height: 150
Age: 10
Height: 190
Age: 9999
[(130, 7), (172, 14), (150, 10)]
>>>
```

Επειδή όμως υπάρχουν κανόνες σχετικά με την εισαγωγή ύψους και ηλικίας, πρέπει να γίνεται έλεγχος χρησιμοποιώντας while(True), έτσι ώστε να μην προχωράει το πρόγραμμα εάν έχουν εισαχθεί λάθος στοιχεία και να συνεχίζει να ζητάει μέχρι να τηρηθούν οι κανόνες. Επιπλέον, πρέπει σε περίπτωση που ο χρήστης δώσει 9999 σε ένα από αυτά τα πεδία, το πρόγραμμα να σταματάει, να επεξεργάζεται τη λίστα και να προχωράει στην διαδικασία των υπολογισμών.

Η δεύτερη έκδοση του προγράμματος ελέγχει μόνο την ορθότητα του ύψους. Αυτό γίνεται ενισχύοντας την αρχικοποίηση #1 και το βήμα #3 με while(True), if και break.

```
# version 2

stu=[]

while(True): #1
    height=int(input('Height: '))
    if((height >= 100 and height <=250) or height==9999):
        break
    print('height error!')

while(height!=9999): #2
    age=int(input('Age: '))
    if(age==9999):
        break
    stu.append((height, age))
    while(True): #3
        height=int(input('Height: '))
        if((height >= 100 and height <=250) or height==9999):
            break
        print('height error!')

print(stu)
```

Δείγμα:

```
================= RESTART: C:/v2.py =================
Height: 130
Age: 7
Height: 90
height error!
Height: 172
Age: 14
Height: 150
Age: 10
Height: 190
Age: 9999
[(130, 7), (172, 14), (150, 10)]
```

Προσθέτοντας τον έλεγχο και για την ηλικία, έχουμε την έκδοση 3, η οποία έχει τρία while(True):

```
# version 3

stu=[]

while(True): #1
    height=int(input('Height: '))
    if((height >= 100 and height <=250) or height==9999):
        break
    print('height error!')

while(height!=9999): #2

    while(True):
        age=int(input('Age: '))
        if((age>=5 and age<=20) or age==9999):
            break
        print('age error!')

    if(age==9999):
        break
    stu.append((height, age))

    while(True): #3
        height=int(input('Height: '))
        if((height >= 100 and height <=250) or height==9999):
            break
        print('height error!')

print(stu)
```

Δείγμα:

```
================= RESTART: C:/v3.py =================
Height: 130
Age: 4
age error!
Age: 7
Height: 172
Age: 14
```

```
Height: 150
Age: 10
Height: 190
Age: 999
age error!
Age: 9999
[(130, 7), (172, 14), (150, 10)]
>>>
```

Υπάρχει επανάληψη κώδικα, επειδή το #1 και το #3 είναι ακριβώς ίδια. Το κεντρικό πρόγραμμα μπορεί να μικρύνει τυποποιώντας τις διαδικασίες εισόδου μέσω συναρτήσεων. Με αυτόν τον τρόπο μπορούμε να απαλείψουμε και τα break αντικαθιστώντας τα με return.

Οι συναρτήσεις απλοποιούν τον κεντρικό κώδικα και επιτρέπουν επαναχρησιμοποίηση των εντολών. Το while(True) δεν χρειάζεται να επαναληφθεί δεύτερη φορά για το βήμα #3. Αρκεί να καλέσουμε την συνάρτηση.

```
# version 4

# functions

def enterHeight():
    while(True):
        height=int(input('Height: '))
        if((height >= 100 and height <=250) or height==9999):
            return height
        print('height error!')

def enterAge():
    while(True):
        age=int(input('Age: '))
        if((age>=5 and age<=20) or age==9999):
            return age
        print('age error!')

# program code

stu=[]

height=enterHeight() #1
while(height!=9999): #2
```

```
    age=enterAge()
    if(age==9999):
        break
    stu.append((height, age))
    height=enterHeight() #3

print(stu)
```

Δείγμα:

```
================= RESTART: C:/v4.py =================
Height: 130
Age: 4
age error!
Age: 7
Height: 172
Age: 14
Height: 150
Age: 10
Height: 190
Age: 999
age error!
Age: 9999
[(130, 7), (172, 14), (150, 10)]
>>>
```

Η πρόσβαση στα περιεχόμενα την λίστας γίνεται όπως έχουμε μάθει χρησιμοποιώντας δείκτες. Για παράδειγμα, έστω ότι η λίστα μας έχει τα παρακάτω περιεχόμενα:

Φαίνεται αμέσως ότι έχουμε στοιχεία για 5 μαθητές. Πέντε tuples σε μία λίστα:

```
>>> stu
[(150, 12), (175, 16), (150, 10), (145, 8), (183, 17)]
```

Κάθε tuple είναι προσβάσιμο μέσω του δείκτη της λίστας. Τα στοιχεία του πρώτου μαθητή είναι:

```
>>> stu[0]
(150, 12)
```

Τα στοιχεία του δεύτερου μαθητή είναι:

```
>>> stu[1]
(175, 16)
```

Το παρακάτω for μας δίνει πρόσβαση σε όλα τα tuples:

```
>>> for s in stu:
     print(s)


(150, 12)
(175, 16)
(150, 10)
(145, 8)
(183, 17)
>>>
```

Το ύψος του πρώτου μαθητή στη λίστα είναι:

```
>>> stu[0][0]
150
```

Η ηλικία του είναι:

```
>>> stu[0][1]
12
```

Προσπαθείστε να καταλάβετε γιατί παίρνουμε για δείκτη -1 και -5 τα παρακάτω αποτελέσματα:

```
>>> stu[-1][1]
17
>>> stu[-1][-1]
17
>>> stu[-5][-1]
12
>>> stu[-5][1]
12
>>>
```

Σε αυτό το σημείο μπορούν να υλοποιηθούν οι μετρήσεις που ζητάει η άσκηση επειδή η λίστα από tuples είναι έτοιμη. Οι μετρήσεις μπορούν να υλοποιηθούν επίσης με καινούργιες συναρτήσεις. Έχουμε πει ότι οι συναρτήσεις είναι καλύτερα να δηλώνονται στην κορυφή του προγράμματος. Οι συναρτήσεις μπορούν να επιστρέφουν τις ζητούμενες τιμές ή να τις τυπώνουν και να μην επιστρέφουν τίποτα. Επειδή οι συναρτήσεις επεξεργάζονται την λίστα μας θα πρέπει να περάσουμε την λίστα σαν παράμετρο.

Συνάρτηση 1: Υπολογίζει και εμφανίζει μέγιστο ύψος μαθητή

```
def highest(lst):
    highest=lst[0][0]
    for st in lst:
        if st[0]>highest:
            highest=st[0]
    return highest
```

Συνάρτηση 2: Υπολογίζει και εμφανίζει το αριθμό των ψηλών δεκαεξάρηδων μαθητών, δηλαδή των μαθητών που είναι 16 ετών και έχουν ύψος πάνω από 1.80.

```
def tall16(lst):
    cnt=0
    for st in lst:
        if st[0]>180 and st[1]==16:
            cnt+=1
    return cnt
```

3. Υπολογίζει και εμφανίζει τον μέσο όρο ύψους όλων

```
def averageHeight(lst):
    total=0
    for st in lst:
        total+=st[0]
    return total/len(lst)
```

Η κλήση των συναρτήσεων:

```
print('highest is', highest(stu))
print('16 year olds higher than 180', tall16(stu))
print('average height', averageHeight(stu))
```