ΦΥΛΛΟ ΕΡΓΑΣΙΑΣ – ΒΑΘΜΟΙ ΜΕ PYTHON

Θα χρησιμοποιήσεις το Προγραμματιστικό Περιβάλλον Geany για να γράφεις προγράμματα στη γλώσσα Python. Το Geany επιτρέπει την ανάπτυξη εφαρμογών σε διάφορες γλώσσες προγραμματισμού. Για την εγκατάστασή του, όπως και της γλώσσας Python, θα μιλήσουμε αργότερα. Προς το παρόν, είναι ήδη εγκατεστημένο στο λογαριασμό σου, οπότε βρες το και εκκίνησέ το.

Ας θεωρήσουμε το πρόβλημα του υπολογισμού του ετήσιου βαθμού σε ένα μάθημα στην Α΄Γάξη του λυκείου. Ο βαθμός αυτός προκύπτει ως μέσος όρος του βαθμού των τελικών γραπτών εξετάσεων (ας αναφερθούμε μόνο σε τέτοια μαθήματα) και του προφορικού βαθμού. Ο προφορικός βαθμός είναι ο μέσος όρος των βαθμών των δύο τετραμήνων, οι οποίοι είναι ακέραιοι στην 20βαθμη κλίμακα. Ο βαθμός των γραπτών εξετάσεων είναι ακέραιος στην 100βαθμη κλίμακα, και πρέπει να αναχθεί στην 20βαθμη. Ο τελικός βαθμός, προς το παρόν, έχει όσα δεκαδικά ψηφία χρειάζεται (πόσα;).

Δραστηριότητα 1: Ας θεωρήσουμε ότι στο παραπάνω πρόβλημα, οι βαθμοί των δύο τετραμήνων είναι 17 και 18, ενώ ο γραπτός βαθμός 97. Πληκτρολόγησε το πρώτο σου πρόγραμμα στην Python, συμπληρώνοντας τα κενά (...), όπως πρέπει.

```
terms = (... + ...) / 2
finals = ... / 5
grade = (... + ...)/ 2
```

Θυμήσου από το γυμνάσιο ότι terms, finals και grade είναι οι μεταβλητές του προγράμματός σου. Η terms παριστάνει τον προφορικό βαθμό του προβλήματος, η finals τον βαθμό των εξετάσεων και η grade τον τελικό βαθμό. Γιατί κάνουμε διαίρεση με το 5;

Αποθήκευσε στο φάκελο Python των Εγγράφων σου το πρώτο σου πρόγραμμα με όνομα αρχείου grades1.py. Με το πλήκτρο F5 εκτέλεσε το πρόγραμμά σου. Είδες κάποιο αποτέλεσμα; Ποιος είναι ο ετήσιος βαθμός; Κλείσε το Geany.

Όπως διαπιστώνεις, και όπως φαίνεται κάτω στο χώρο «Κατάσταση», με την εκκίνησή του, «ανοίχθηκε νέο αρχείο "χωρίς όνομα"». Την επόμενη φορά, αντί για αυτό θα ανοίγει/ουν με την εκκίνηση, όσα αρχεία είχες αφήσει ό ίδιος ανοιχτά την τελευταία φορά.

Όλα τα προγράμματά σου στην Python, φρόντισε να έχουν την προέκταση ονόματος .py. Παρατήρησε ότι μόλις είπες, με την αποθήκευση, ότι οι 3 γραμμές σου είναι σε γλώσσα Python, το Geany άρχισε να χρωματίζει τον κώδικά σου.

Δραστηριότητα 2: Άνοιξε το Geany. *Όπως βλέπεις, «θυμάται» ότι την τελευταία φορά δούλευες με το grades1.py και το έχει φορτωμένο.* Χρησιμοποίησε, όπως φαίνεται δίπλα, τη **συνάρτηση** print() της Python για να δεις τον τελικό βαθμό. Εκτέλεσε το πρόγραμμά σου. Κάπως καλύτερα; *Κάθε φορά που εκτελείς το πρόγραμμα αυτό αποθηκεύεται εκ νέου.*

```
terms = (... + ...) / 2
finals = ... / 5
grade = (... + ...)/ 2
print(grade)
```

Μια συνάρτηση είναι ένα αυτόνομο τμήμα προγράμματος που εκτελεί κάποιες λειτουργίες. Υπάρχουν συναρτήσεις έτοιμες στην Python και τις βιβλιοθήκες της, αλλά σε λίγο θα φτιάχνεις και τις δικές σου.

Ας περάσουμε τα δεδομένα του προβλήματος, τους τρεις βαθμούς, σε τρεις μεταβλητές, t1, t2 και finals, που την έχουμε ήδη. Συμπλήρωσε και τροποποίησε κατάλληλα. Δεν θα δεις καμία διαφορά στην εκτέλεση του προγράμματος. Αν θελήσεις τώρα να αλλάξεις κάποιο/α από τα τρία δεδομένα τι θα κάνεις; Αν στο πρώτο τετράμηνο θέλεις 16 και στο γραπτό 91, πού θα επέμβεις στο πρόγραμμα;

```
t1 = ...
t2 = ...
finals = ...

terms = (... + ...) / 2
finals = ... / 5
grade = (... + ...)/ 2
print(grade)
```

Δυστυχώς το πρόγραμμά σου εξακολουθεί να μην παρέχει διεπαφή στο χρήστη για να εισάγει τα δεδομένα που θέλει. Μόνο ο προγραμματιστής του -εσύ- μπορεί να μεταβάλλει τα δεδομένα, *μέσα* στον κώδικα του προγράμματος. Όμως τα προγράμματά μας χρειάζονται ρητή είσοδο δεδομένων. Ας διορθώσουμε, λοιπόν.

Δραστηριότητα 3: Θα χρησιμοποιήσεις τώρα τη συνάρτηση input() προκειμένου να εισάγονται δεδομένα κατά την εκτέλεση από το χρήστη. Αυτά θα αποθηκεύονται σε αντίστοιχες μεταβλητές. Δοκίμασέ την σε συνδυασμό με την print(), ώστε να την επαληθεύσεις, σε ένα διαφορετικό αρχείο. *Από το μενού Έγγραφο, διάλεξε Κλωνοποίηση. Στην καρτέλα με το νέο αρχείο «χωρίς όνομα» κάνε τις τροποποιήσεις που βλέπεις. Εκτέλεσε το νέο πρόγραμμα, αφού το ονομάσεις input-test.py* Μπορείς να καταλάβεις ότι με τον κατάλληλο χειρισμό των ορισμάτων των συναρτήσεων input() και print(), το πρόγραμμά μας γίνεται εύγλωττο πλέον προς το χρήστη, από βουβό που ήταν ως τώρα.

```
t1 = input()
print(t1)

t2 = input("Βαθμός 2ου Τετραμήνου: ")
print("Έδωσες", t2)

print("Βαθμός Εξετάσεων: ")
finals = input()
print(finals)

#terms = (t1 + t2) / 2
```

Η συνάρτηση input() διακόπτει τη ροή της εκτέλεσης του προγράμματος, περιμένει είσοδο από το πληκτρολόγιο του χρήστη, και την επιστρέφει στο πρόγραμμα, ως σειρά από τους χαρακτήρες που πληκτρολογήθηκαν –ως μια συμβολοσειρά, ή string.
Γενικότερα, οι συναρτήσεις έχουν το καλό να επιστρέφουν χρήσιμα δεδομένα στο πρόγραμμα όπου καλούνται.

Ορίσματα ή παραμέτρους μιας συνάρτησης ονομάζουμε αυτά που βλέπουμε μέσα στην παρένθεσή της. Ό,τι είναι το x για μια μαθηματική συνάρτηση f(x), είναι και τα ορίσματα στις συναρτήσεις στην Python: αυτά που χρησιμοποιούν για να επιτελέσουν τις λειτουργίες τους και να παράξουν ό,τι πρέπει να επιστρέψουν.

Η γραμμή με το χαρακτήρα # στην αρχή της είναι σαν να μην υπάρχει. Η Python την παρακάμπτει. Αφαίρεσε το #, και εκτέλεσε το πρόγραμμα. Φαίνεται ότι υπάρχει πρόβλημα. Η Python μας λέει ότι δεν μπορεί να κάνει αριθμητική διαίρεση συμβολοσειράς με ακέραιο. Οι αριθμοί που δίνει ο χρήστης ως βαθμούς, θεωρούνται από την Python απλοί χαρακτήρες. Το 19 με την input δεν είναι παρά η σειρά από τους δύο χαρακτήρες 1 και 9 στα αντίστοιχα πλήκτρα, και όχι ο αριθμός 19. Πρέπει να πούμε στην Python να μετατρέπει τη συμβολοσειρά στους αντίστοιχους ακεραίους. Αυτό θα κάνει η συνάρτηση int(). Για παράδειγμα η γραμμή κώδικα `finals = int(input())` θα μετατρέψει τη συμβολοσειρά που επιστρέφει η `input()` σε ακέραιο.

```
Traceback (most recent call last):
  File "input-test.py", line 11, in
<module>
    terms = (t1 + t2) / 2
TypeError: unsupported operand type(s)
for /: 'str' and 'int'
```

Τα σχόλια απευθύνονται στον ίδιο τον προγραμματιστή, ή όποιον διαβάσει το πρόγραμμα.

Δραστηριότητα 4: Ας κάνουμε το πρόγραμμά μας ορθότερο, πιο εύγλωττο και επικοινωνιακό, φροντίζοντας για τα κατάλληλα μηνύματα στις input(), τη μετατροπή των δεδομένων και την προσθήκη επεξηγητικού μηνύματος στην εμφάνιση του τελικού αποτελέσματος, του ετήσιου βαθμού. Αποθήκευσε την τελική μορφή του πρώτου σας προγράμματος στην Python, του grades1.py, και δοκίμασε την εκτέλεσή του με τους βαθμούς της αρεσκείας σου.

```
t1 = int(input("Βαθμός 1ου Τετραμήνου (1-20): "))
t2 = int(input("Βαθμός 2ου Τετραμήνου (1-20): "))
finals = int(input("Βαθμός Εξετάσεων     (0-100): "))

terms = (t1 + t2) / 2
finals = finals / 5
grade = (terms+finals)/2

print("Ετήσιος βαθμός                :", grade)
```

Κι αν δώσεις για βαθμό τετραμήνου το 21, ή εξετάσεων το 200;

Δραστηριότητα 5: Πριν ασχοληθούμε με τα ερωτήματα στην κατακλείδα της προηγούμενης δραστηριότητας, ας αναρωτηθούμε σχετικά με κάτι άλλο. Πόσες μεταβλητές χρειαζόμαστε σε ένα πρόγραμμα; Ένα από τα χαρακτηρι-

```
print("Ετήσιος βαθμός                :",
      ((int(input("Βαθμός 1ου Τετραμήνου (1-20): "))+
        int(input("Βαθμός 2ου Τετραμήνου (1-20): ")))/2+
        int(input("Βαθμός Εξετάσεων      (0-100): "))/ 5)/2
```

στικά της Python είναι η λακωνικότητά της. Συνδυάζοντας τα δύο αυτά θα μπορούσαμε να γράψουμε όλο μας το πρόγραμμα σε μία γραμμή, χωρίς καμία μεταβλητή, με διαδοχικές αντικαταστάσεις.

Συμβουλή: Μην το παρακάνεις! Καλό είναι να μπορείς να διαβάζεις τα προγράμματά σου με άνεση, και μετά από καιρό.

Κλωνοποίησε το πρόγραμμά σου και ονόμασε το αντίγραφο grades2.py. Θα δούμε την πιο χρήσιμη εντολή όλων των γλωσσών προγραμματισμού. Όλος ο πλούτος τους βασίζεται σε αυτήν, ο ίδιος ο υπολογιστής είναι αυτό το σπουδαίο κατασκεύασμα, εξαιτίας της: Η εντολή if καθορίζει την εκτέλεση εντολών μόνο εφόσον ικανοποιείται ή όχι μια συγκεκριμένη συνθήκη. Ας το δούμε στο παραπάνω απόσπασμα στο grades2.py που θα τροποποιήσετε. Καταρχάς η συνάρτηση range(1,21) επιστρέφει την ακολουθία των ακεραίων 1 έως 20. Θα αναφερθούμε σε αυτήν πιο αναλυτικά αργότερα. Ο τελεστής in ελέγχει αν η τιμή της t1 που επιστρέφει η input(), μέσω της int(), περιλαμβάνεται στην ακολουθία αυτή. Η έκφραση αυτή που ακολουθεί τη λέξη if αποκαλείται συνθήκη και αποτιμάται σε μία από δύο τιμές: ☐rue, αν ισχύει, ή False στην αντίθετη περίπτωση. Παρατηρήστε την άνω και κάτω τελεία μετά τη συνθήκη. Δηλώνει ότι η εντολή δεν τελείωσε, αλλά θα συνεχιστεί σε εσοχή στις επόμενες γραμμές. Αυτό που ακολουθεί σε εσοχή είναι αυτό που θα κάνει το πρόγραμμα εφόσον η συνθήκη αποτιμηθεί αληθής (☐rue). Θα εμφανιστεί το μήνυμα «Εντάξει». Ακολουθεί σε στοίχιση με την if η λέξη else και νέα άνω και κάτω τελεία. Δηλώνεται έτσι και πάλι ότι η εντολή δεν τε-

```
t1 = int(input("Βαθμός 1ου Τετραμήνου (1-20): "))
if t1 in range(1,21):
    print("Εντάξει.")
else:
    print("Έδωσες μη επιτρεπτό βαθμό.")
t2 = int(input("Βαθμός 2ου Τετραμήνου (1-20): "))
finals = int(input("Βαθμός Εξετάσεων      (0-100): "))
terms = (t1 + t2) / 2
finals = finals / 5
grade = (terms+finals)/2

print("Ετήσιος βαθμός                :", grade)
```

λείωσε, αλλά θα συνεχιστεί σε εσοχή στις επόμενες γραμμές. Αυτό που ακολουθεί σε εσοχή είναι αυτό που θα κάνει το πρόγραμμα εφόσον η συνθήκη του if αποτιμηθεί ψευδής (False).

Ακολουθεί σε στοίχιση με τις προηγηθείσες εντολές, η ανάθεση τιμής στην t2, και την finals. Ακριβώς επειδή δεν περιλαμβάνονται σε εσοχή, σημαίνει ότι δεν ανήκουν στην if αλλά εκτελούνται ασχέτως συνθηκών.

Είδαμε, λοιπόν, ότι μπορούμε να εμφανίσουμε ένα μήνυμα, ανάλογα με το αν η είσοδος από τον χρήστη για τη μεταβλητή t1 είναι συμβατή με αυτό που αναμένουμε. Και κάτι ανάλογο θα έπρεπε να κάνουμε για την t2 και τη finals. Ωστόσο, είναι αυτό αρκετό; *Εκτέλεσε και πάλι το πρόγραμμά σου*. Αυτό που θα θέλαμε είναι όταν η τιμή της t1 είναι μη έγκυρη, εκτός από το μήνυμα που θα ειδοποιεί σχετικά τον χρήστη, να *επαναλαμβάνεται* και πάλι η διαδικασία της εισόδου. Και αυτό να συνεχίζεται *όσο* ο χρήστης δεν ανταποκρίνεται με έγκυρη είσοδο. Αυτού του είδους την επανάληψη –να επαναλαμβάνεται κάτι όσο δεν ικανοποιείται μία συνθήκη, θα δούμε στη συνέχεια.

Διάφορες μορφές της εντολής if:

```
προηγούμενες εντολές
if συνθήκη:
    εντολές1
else:
    εντολές2
επόμενες εντολές
```

```
προηγούμενες εντολές
if συνθήκη:
    εντολές
επόμενες εντολές
```

```
προηγούμενες εντολές
if συνθήκη1:
    εντολές1
elif συνθήκη2:
    εντολές2
elif συνθήκη3:
    εντολές3
...
else:
    εντολέςN
επόμενες εντολές
```

```
προηγούμενες εντολές
if συνθήκη1:
    εντολές1
elif συνθήκη2:
    εντολές2
...
elif συνθήκηN:
    εντολέςN
επόμενες εντολές
```

```
Δεν τελειώσαμε (αφού τώρα ξεκινάμε…)
Όσο δεν τελειώσαμε
    πάρε το t1
    Αν το t1 είναι εντάξει
        τελειώσαμε
    αλλιώς
        μήνυμα λάθους
```

Δραστηριότητα 6: Εξακολουθούμε να εργαζόμαστε με το grades2.py. Ο κώδικας παρακάτω υλοποιεί αυτό που θέλουμε για την t1, και που σχηματικά φαίνεται δίπλα. Ενσωμάτωσέ τον και εκτέλεσε πάλι το πρόγραμμα. Όπως

```
Δεν τελειώσαμε (αφού τώρα ξεκινάμε…)
Όσο δεν τελειώσαμε
    πάρε το t1
    Αν το t1 είναι εντάξει
        τελειώσαμε
    αλλιώς
        μήνυμα λάθους
```

```
while συνθήκη:
    εντολές
```

```
ok=False
while not ok:
    t1 = int(input("Βαθμός 1ου Τετραμήνου (1-20): "))
    if t1 in range(1,21):
        ok = True
    else:
        print("Έδωσες μη επιτρεπτό βαθμό.")

t2 = int(input("Βαθμός 2ου Τετραμήνου (1-20): "))
finals = int(input("Βαθμός Εξετάσεων     (0-100): "))
terms = (t1 + t2) / 2
finals = finals / 5
grade = (terms+finals)/2

print("Ετήσιος βαθμός                :", grade)
```

βλέπεις, δεν θα προχωρήσει στη γραμμή που αφορά τη μεταβλητή t2, παρά μόνο αν η τιμή της t1 διαβαστεί στο επιτρεπτό εύρος. Δύο παρατηρήσεις:

α. Η εντολή while περιλαμβάνει και αυτή συνθήκη, όπως η εντολή if, εκτείνεται επίσης σε περισσότερες από μία γραμμές, και αυτό επιτυγχάνεται επίσης με την άνω και κάτω τελεία μετά τη συνθήκη, στο τέλος της αρχικής γραμμής, και με την εσοχή στη συνέχεια.

β. Σημαντικό ρόλο παίζει η λογική μεταβλητή ok, που δείχνει πότε η είσοδος τιμής για την t1 γίνεται αποδεκτή.

Τι θα κάνεις τώρα για την έγκυρη είσοδο τιμής για τις άλλες δύο μεταβλητές: t2 και finals; Μια λύση είναι να αντιγράψεις και να τροποποιήσεις κατάλληλα τον κώδικα της t1, δύο φορές για τις άλλες δύο μεταβλητές. Δοκίμασέ το, η αντιγραφή κι επικόλληση λειτουργεί ως συνήθως στο Geany.

```
ok=False
while not ok:
    t1 = int(input("Βαθμός 1ου Τετραμήνου (1-20): "))
    if t1 in range(1,21):
        ok = True
    else:
        print("Έδωσες μη επιτρεπτό βαθμό.")

ok=False
while not ok:
    t2 = int(input("Βαθμός 2ου Τετραμήνου (1-20): "))
    if t2 in range(1,21):
        ok = True
    else:
        print("Έδωσες μη επιτρεπτό βαθμό.")

ok=False
while not ok:
    finals = int(input("Βαθμός Εξετάσεων     (0-100): "))
    if finals in range(101):
        ok = True
    else:
        print("Έδωσες μη επιτρεπτό βαθμό.")

terms = (t1 + t2) / 2
finals = finals / 5
grade = (terms+finals)/2

print("Ετήσιος βαθμός                :", grade)
```

Όμως, εδώ έχεις την ευκαιρία να φτιάξεις την πρώτη σου συνάρτηση!

Οι λογικές μεταβλητές μπορούν να πάρουν μία από δύο μόνο τιμές: True ή False, είτε απευθείας με ανάθεση, όπως στο παράδειγμά μας η ok, είτε από το αποτέλεσμα μιας λογικής έκφρασης, μιας έκφρασης, δηλαδή, που περιέχει λογικούς ή/και συγκριτικούς τελεστές και οδηγεί στην αποτίμηση Ισχύει/Δεν Ισχύει, True/False.

Μια συνθήκη δεν είναι παρά μια λογική έκφραση. Μπορεί να περιλαμβάνει λογικούς τελεστές, όπως ο not, ή/και τελεστές σύγκρισης ή …, όπως ο in.

Η range(start, stop, step) επιστρέφει μια ακολουθία τιμών από την τιμή start μέχρι την stop-1, με βήμα step. Το start εννοείται 0, και το step 1, αν παραλειφθούν.

Δραστηριότητα 7: Κλωνοποίησε το πρόγραμμά σου και ονόμασε το αντίγραφο grades3.py. Σε αυτό θα δουλέψεις στη συνέχεια. Θα φτιάξεις μια συνάρτηση με το όνομα getGrade() και ένα όρισμα το term, που θα μπορεί να πάρει τιμή 1, 2 ή 3, για το 1ο ή 2ο τετράμηνο και τις εξετάσεις, αντίστοιχα. Μια συνάρτηση ξεκινά με την επικεφαλίδα της, που περιέχει τη λέξη def, το όνομα που θέλουμε και τα ορίσματα σε παρένθεση. Η επικεφαλίδα τελειώνει με άνω και κάτω τελεία, οπότε, καλά το υποθέτεις, όλο το υπόλοιπο, το σώμα της, θα είναι σε εσοχή:

```
def getGrade(term):
```

Ανάλογα με την τιμή του term, δύο πράγματα θα αλλάζουν: το μήνυμα που θα συνοδεύει την input() που διαβάζει το βαθμό του χρήστη, και το επιτρεπόμενο εύρος τιμών για αυτόν. Θα το πετύχουμε με μια εντολή if:

```
def getGrade(term):
    if term==1 or term==2:
        message="Βαθμός "+str(term)+"ου Τετραμήνου (1-20): "
        r=range(1,21)
    elif term==3:
        message="Βαθμός Εξετάσεων     (0-100): "
        r=range(101)
```

Ο συγκριτικός τελεστής ισότητας δηλώνεται με ==.

Ο λογικός τελεστής or -νωρίτερα είδαμε και τον not- επιτελεί διάζευξη, και δίνει αποτέλεσμα True αν ένα τουλάχιστον όρισμά της είναι True.

Η συνάρτηση str() μετατρέπει το όρισμά της σε αλφαριθμητική τιμή.

Ο τελεστής + σε αλφαριθμητικά επιτελεί συνένωση.

Η παραπάνω if υπολόγισε τις σωστές τιμές για το μήνυμα message και το εύρος r, που είμαστε έτοιμοι να χρησιμοποιήσουμε:

```
    ok=False
    while not ok:
        grade = int(input(message))
        if grade in r:
            ok = True
        else:
            print("Έδωσες μη επιτρεπτό βαθμό.")
    return grade
```

Η συνάρτηση επιστρέφει την έγκυρη τιμή βαθμού που διάβασε με την εντολή return. Είσαι έτοιμος να συμπληρώσεις το πρόγραμμά σου, μετά τη συνάρτηση, όπου και θα την καλείς τρεις φορές:

```
t1 = getGrade(1)
t2 = getGrade(2)
finals = getGrade(3)

terms = (t1 + t2) / 2
finals = finals / 5
grade = (terms+finals)/2

print("Ετήσιος βαθμός              :", grade)
```

Αποθήκευσε και εκτέλεσε το πρόγραμμά σου. Με τα αρχικά δεδομένα της δραστηριότητας 1 -17, 18 και 97- εξακολουθείς να παίρνεις το αποτέλεσμα 18,45. Αυτό δεν είναι ορθό για το πρόβλημά μας. Στο λύκειο ο τελικός βαθμός στρογγυλοποιείται στο πρώτο δεκαδικό ψηφίο· το 18,45 πρέπει να στρογγυλοποιείται στο 18,5.

Δραστηριότητα 8: Κλωνοποίησε το πρόγραμμά σου και ονόμασε το αντίγραφο grades4.py. Σε αυτό θα δουλέψεις στη συνέχεια. Θα φτιάξεις μια συνάρτηση με το όνομα round() και δύο ορίσματα: number και digits. Η round() θα στρογγυλοποιεί το number σε έναν αριθμόμε digits το πλήθος δεκαδικά ψηφία. Δεν είναι δύσκολο:

```
def myround(number,digits):
    return(int(number*10**digits+0.5)/10**digits)
```

Άλλαξε τώρα την τιμή της grade στο κύριο πρόγραμμά σου, και εκτέλεσέ το:

```
grade = myround((terms+finals)/2,1)
```