# Μάντεψε τον Αριθμό

## Ενδεικτικές Απαντήσεις Φύλλου Εργασίας

10 Σεπτεμβρίου 2016
10:23

## Οδηγίες

Αρχικά, το πρόγραμμά μας θα δίνει κάποιες οδηγίες στον παίκτη σχετικά με το παιχνίδι.

1. Ξεκινήστε το πρόγραμμα με την εντολή:

   ```
   print("Θα σκεφτώ έναν αριθμό από το 1 μέχρι το 32.")
   ```

   Εκτελέστε το πρόγραμμα και δείτε τι συμβαίνει.

   *Εμφανίζεται στην οθόνη το μήνυμα:*

   ```
   Θα σκεφτώ έναν αριθμό από το 1 μέχρι το 32.
   ```

2. Βασιστείτε στην εντολή του προηγούμενου βήματος και συμπληρώστε το πρόγραμμα με μια ακόμα εντολή, έτσι ώστε να εμφανίζεται στην οθόνη το μήνυμα:

   ```
   Εσύ θα προσπαθήσεις να τον μαντέψεις.
   ```

   *Η εντολή που θα προσθέσουμε στο πρόγραμμα είναι:*

   ```
   print("Εσύ θα προσπαθήσεις να τον μαντέψεις.")
   ```

   Εκτελέστε ξανά το πρόγραμμα. Τα καταφέρατε;

   *Ναι, εμφανίζεται στην οθόνη και το δεύτερο μήνυμα.*

3. Το πρόγραμμά θα ορίζει τώρα τον μυστικό αριθμό. Έστω ότι επιλέγει το τυχερό 13. Προσθέστε την παρακάτω εντολή στο πρόγραμμα:

   ```
   secret = 13
   ```

   Η `secret` είναι μια *μεταβλητή* στην οποία αποθηκεύεται η τιμή του μυστικού αριθμού που επιλέγει το πρόγραμμά μας.

   Τώρα μπορούμε να αναφερόμαστε στον μυστικό αριθμό με το όνομα `secret`, χωρίς να έχει σημασία ποια είναι η τιμή του.

   Εκτελέστε το πρόγραμμα. Παρατηρείτε κάποια διαφορά στην εκτέλεση μετά την προσθήκη της παραπάνω εντολής;

   *Όχι, δεν υπάρχει διαφορά στην εκτέλεση του προγράμματος. Η ανάθεση τιμής σε ένα όνομα γίνεται από τον προγραμματιστή και δεν είναι "ορατή"*

*στον χρήστη. Ο χρήστης δεν γνωρίζει τα ονόματα και τις τιμές των μεταβλητών που χρησιμοποιούνται σε ένα πρόγραμμα. Ο προγραμματιστής επιλέγει να εμφανίσει την τιμή κάποιας μεταβλητής χρησιμοποιώντας το όνομά της μαζί με την* `print`.

## Γνωριμίες

Το πρόγραμμά μας θα ζητά από τον παίκτη το όνομά του και θα τον καλωσορίζει στο παιχνίδι.

4. *Προσθέστε* τις παρακάτω εντολές οπουδήποτε στο πρόγραμμα.

```
print("Πώς σε λένε;")
player = input()
```

Η `input()` επιστρέφει το κείμενο που πληκτρολόγησε ο χρήστης, δηλαδή επιστρέφει πάντα μια *αλφαριθμητική τιμή*. Στην συγκεκριμένη περίπτωση, χρησιμοποιούμε την `input()` για να διαβάσουμε την απάντηση του χρήστη, η οποία αποθηκεύεται στη μεταβλητή `player`.

Εκτελέστε το πρόγραμμα. Η μεταβλητή `player` παίρνει την τιμή που πληκτρολογείτε (αναλαμβάνοντας το ρόλο του χρήστη πλέον).

5. Συμπληρώστε το πρόγραμμα με την παρακάτω εντολή.

```
print("Γεια σου", player)
```

Εκτελέστε το πρόγραμμα σας 2–3 φορές, παίζοντας το ρόλο του χρήστη, πληκτρολογήστε κάθε φορά διαφορετικά ονόματα και παρατηρήστε τι εμφανίζει.

*Το πρόγραμμα εμφανίζει το μήνυμα* `Γεια σου` *ακολουθούμενο από το όνομα που πληκτρολόγησε κάθε φορά ο παίκτης.*

Ποια πιστεύετε ότι θα ήταν η διαφορά αν είχατε πληκτρολογήσει `print("Γεια σου player")`;

*Θα εμφανίζονταν στην οθόνη το μήνυμα* `Γεια σου player` *. Οτιδήποτε βρίσκεται μέσα σε εισαγωγικά αποτελεί μια αλφαριθμητική τιμή, δηλαδή κείμενο, και δεν αφορά ονόματα μεταβλητών.*

Ποια θα ήταν η διαφορά αν αντί της εντολής `player = input()` χρησιμοποιούσαμε την εντολή `player = "Μυρσίνη"`;

*Η μεταβλητή* `player` *θα ήταν καθορισμένη από τον προγραμματιστή και όχι από τον χρήστη και θα είχε σε κάθε εκτέλεση του προγράμματος την τιμή* `"Μυρσίνη"` *.*

Ποια θα ήταν η διαφορά αν στο βήμα 3 γράφαμε `secret = input()`, αντί για `secret = 13`;

*Ο χρήστης θα καθόριζε την τιμή της* `secret`*, ανάλογα με το τι θα πληκτρολογούσε σε κάθε εκτέλεση του προγράμματος. Η μεταβλητή* `secret` *δεν θα είχε απαραίτητα την τιμή* `13`.

## Φρέναρε λίγο

Το πρόγραμμά μας εμφανίζει όλα τα μηνύματα μαζεμένα. Θα δώσουμε μια ανάσα στον παίκτη, ώστε να προλαβαίνει να τα διαβάζει έχοντας λίγο χρόνο στη διάθεσή του, πριν του εμφανίσουμε το επόμενο μήνυμα.

Στο σημείο αυτό θα χρειαστούμε μια λειτουργία που δεν προσφέρεται από τις βασικές εντολές της Python. Για το σκοπό αυτό θα χρειαστεί να εισάγουμε στο πρόγραμμά μας μια βιβλιοθήκη και συγκεκριμένα τη βιβλιοθήκη `time`, η οποία παρέχει τη λειτουργικότητα που μας χρειάζεται. Οι βιβλιοθήκες είναι συλλογές από μικρά προγράμματα που μπορούμε να χρησιμοποιήσουμε στα προγράμματά μας.

6. *Προσθέστε στην αρχή του προγράμματος* την εντολή που ακολουθεί, για να *εισάγετε* τη βιβλιοθήκη `time`:

```
import time
```

7. Εισάγετε, ανάμεσα στις δύο πρώτες `print`, την παρακάτω εντολή:

```
time.sleep(1)
```

Με την εντολή αυτή χρησιμοποιείται η *συνάρτηση* `sleep` από τη βιβλιοθήκη `time`. Τί αποτέλεσμα έχει η προσθήκη αυτής της εντολής;

*Εισάγεται μια καθυστέρηση ανάμεσα στην εμφάνιση των δύο μηνυμάτων.*

Τί ρόλο παίζει η *παράμετρος* `1`; Διερευνήστε τι θα συμβεί αν χρησιμοποιήσουμε άλλον αριθμό στη θέση του `1`.

*Η παράμετρος* `1` *καθορίζει ότι θα υπάρχει μια καθυστέρηση ενός δευτερολέπτου. Αν χρησιμοποιήσουμε άλλον αριθμό στη θέση του* `1` *θα αλλάξει η περίοδος της χρονικής καθυστέρησης.*

## Ας παίξουμε

Ήρθε η ώρα να ξεκινήσει το πραγματικό παιχνίδι. Το πρόγραμμά θα ζητά από τον παίκτη έναν αριθμό και στη συνέχεια θα εμφανίζει μήνυμα αν βρήκε τον μυστικό αριθμό ή όχι.

8. *Προσθέστε* τις παρακάτω εντολές στο πρόγραμμα:

```
print("Δώσε αριθμό:")
number = int(input())
```

Τι διαφορά παρατηρείτε σε σχέση με τις εντολές που χρησιμοποιήσατε για να ζητήσετε το όνομα του παίκτη στο βήμα 4;

*Χρησιμοποιούμε την* `int()` *μαζί με την* `input()`*. Το αποτέλεσμα που επιστρέφεται από την* `input()`*, δηλαδή το κείμενο που πληκτρολόγησε ο χρήστης, δίνεται στη συνέχεια στην* `int()` *για να το επεξεργαστεί (δες σχετικά την απάντηση στην επόμενη ερώτηση).*

Ποια διαφορά πιστεύετε ότι έχει η τιμή του ονόματος του παίκτη από την τιμή του `number`;

*Η τιμή του* `number` *είναι αριθμητική και όχι αλφαριθμητική, όπως είναι το όνομα του παίκτη. Επειδή η τιμή που επιστρέφει η* `input()` *είναι αλφαριθμητική είναι απαραίτητο να την μετατρέψουμε σε ακέραιο αριθμό, κάτι που επιτυγχάνεται με τη βοήθεια της* `int()`.

9. Συμπληρώστε το πρόγραμμα, ώστε να εμφανίζει στον παίκτη τον αριθμό που πληκτρολόγησε, για παράδειγμα:

```
Έδωσες τον αριθμό 20
```

Αν δυσκολευτείτε, ανατρέξτε στο βήμα 5.

*Η εντολή που θα προσθέσουμε στο πρόγραμμα είναι:*

```
print("Έδωσες τον αριθμό", number)
```

10. *Προσθέστε* στο πρόγραμμα σας τις εντολές:

```
print("Σωστά!")
print("Λάθος...")
```

Τι *πιστεύετε* ότι θα εμφανίσει το πρόγραμμα αν το εκτελέσουμε τώρα;

*Θα εμφανίζονται και τα δύο μηνύματα. Και το* `Σωστά!` *και το* `Λάθος`*, αφού θα εκτελούνται σε κάθε περίπτωση και οι δύο εντολές.*

Εκτελέστε το πρόγραμμα. Επιβεβαιώνεται η απάντησή σας;

*Ναι, επιβεβαιώνεται.*

Σε ποια περίπτωση είναι επιθυμητό να εμφανίζεται το `"Σωστά!"`; Αντίστοιχα, σε ποια περίπτωση είναι επιθυμητό να εμφανίζεται το `"Λάθος..."`;

*Το μήνυμα* `Σωστά!` *πρέπει να εμφανίζεται στην περίπτωση που ο αριθμός που πληκτρολόγησε ο χρήστης είναι ίδιος με το μυστικό αριθμό. Στην αντίθετη περίπτωση πρέπει να εμφανίζεται το μήνυμα* `Λάθος`*.*

11. Θα τροποποιήσουμε το πρόγραμμά μας, έτσι ώστε να *ελέγχει* αν ο παίκτης μάντεψε τον μυστικό αριθμό και στη συνέχεια να εμφανίζει *μόνο* το κατάλληλο μήνυμα.

*Προσθέστε* τις παρακάτω εντολές:

```
if number == secret:
print("Σωστά!")
else:
print("Λάθος...")
```

Με το == ελέγχεται αν δύο τιμές είναι ίσες. Μην το συγχέετε με το = που χρησιμοποιείται για να δώσουμε τιμή σε μια μεταβλητή.

Εκτελέστε το πρόγραμμα. Παρατηρήστε το *σφάλμα* που εμφανίζει.

Στις περισσότερες γλώσσες προγραμματισμού δεν θα αντιμετωπίζατε κάποιο πρόβλημα, όμως η Python έχει μια "ευαισθησία": Οι εντολές που εκτελούνται στη μία ή στην άλλη περίπτωση πρέπει να

*ξεχωρίζουν*, πρέπει με κάποιον τρόπο να επισημανθεί ότι οι εντολές αυτές *ανήκουν* αντίστοιχα στην **if** και την **else**.

Στην Python, αυτό επιτυγχάνεται με τις *εσοχές*.

12. *Προσθέστε* 4 κενά πριν από τις δύο `print` και εκτελέστε το πρόγραμμα.

```
if secret == number:
    print("Σωστά!")
else:
    print("Λάθος...")
```

Εκτελέστε το πρόγραμμα σας δύο φορές. Στην πρώτη δώστε σωστά τον μυστικό αριθμό, ενώ στην επόμενη δώστε έναν διαφορετικό.

Εμφανίζεται το κατάλληλο μήνυμα σε κάθε περίπτωση;

*Ναι. Όταν ο αριθμός που δίνουμε είναι ίσος με το μυστικό, δηλαδή* `13`*, τότε το πρόγραμμα εμφανίζει το μήνυμα* `Σωστά` *, ενώ σε κάθε άλλη περίπτωση εμφανίζει το μήνυμα* `Λάθος` *.*

Σε ποια περίπτωση εκτελούνται οι εντολές της else;

*Όταν ο αριθμός που πληκτρολογεί ο χρήστης δεν είναι ίδιος με το μυστικό αριθμό.*

13. *Προσθέστε* στο πρόγραμμα μια `print`, η οποία θα εμφανίζει τον μυστικό αριθμό στην περίπτωση που ο παίκτης απαντήσει λάθος, όπως παρακάτω:

```
Ο μυστικός αριθμός ήταν ο 13.
```

*Η εντολή που θα προσθέσουμε στο πρόγραμμα είναι:*

```
    print("Ο μυστικός αριθμός ήταν ο", secret)
```

Εκτελέστε το πρόγραμμα δύο φορές. Στην πρώτη δώστε σωστά τον μυστικό αριθμό, ενώ στην επόμενη δώστε έναν διαφορετικό.

Εμφανίζει το μυστικό αριθμό *μόνο* στην περίπτωση που ο παίκτης δεν τον μαντέψει; Αν όχι, γιατί πιστεύετε ότι συμβαίνει αυτό;

*Η εντολή πρέπει να προστεθεί μέσα στην* **else***, επομένως πρέπει να προηγηθούν τέσσερα κενά, ώστε να δημιουργηθεί η κατάλληλη εσοχή. Σε διαφορετική περίπτωση θα εκτελείται ανεξάρτητα από τον αν ο παίκτης μαντέψει το μυστικό αριθμό.*

Αν ο μυστικός αριθμός εμφανίζεται είτε ο χρήστης τον μαντέψει, είτε όχι, τότε πιθανότατα δεν τοποθετήσατε τη νέα εντολή *μέσα* στην **else**, αλλά μετά από αυτή. Προσθέστε 4 κενά πριν την εντολή, για να υποδηλώσετε ότι κι αυτή ανήκει στην **else**.

14. Στο μήνυμα που εμφανίσατε προηγουμένως χρησιμοποιήσατε την μεταβλητή `secret`;

*Ναι.*

Αν απαντήσατε όχι, *τροποποιήστε* κατάλληλα την εντολή `print`, ώστε να χρησιμοποιεί τη `secret` και όχι το `13`.

Τι πλεονέκτημα πιστεύετε ότι έχει η χρήση της μεταβλητής `secret` στο μήνυμα, αντί της απευθείας εμφάνισης του αριθμού `13`;

*Αν δεν χρησιμοποιήσουμε τη μεταβλητή* `secret` *η* `print()` *θα εμφανίζει πάντα ακριβώς το ίδιο, σταθερό μήνυμα, ενώ αν χρησιμοποιήσουμε τη μεταβλητή* `secret` *η* `print()` *θα εμφανίζει την τιμή της, η οποία εδώ είναι* `13`*, αλλά θα μπορούσε να είναι διαφορετική.*

Δοκιμάστε να παίξετε 2–3 φορές ακόμα. Υπάρχει κάτι που σας ενοχλεί στον τρόπο που λειτουργεί το πρόγραμμα;

*Μετά από λίγες εκτελέσεις ο παίκτης θα αντιληφθεί ότι ο μυστικός αριθμός είναι πάντα το* `13`*.*

## Τυχαιότητα

Το παιχνίδι μας δεν αξίζει να το παίξεις πάνω από μία-δύο φορές, αν ο μυστικός αριθμός είναι πάντα το `13`. Αυτό που πραγματικά χρειαζόμαστε είναι να επιλέγεται κάθε φορά ένας διαφορετικός αριθμός.

Για τον σκοπό αυτό, θα χρειαστούμε τη βιβλιοθήκη `random`, όπως προηγουμένως είχαμε χρησιμοποιήσει τη βιβλιοθήκη `time`.

15. *Προσθέστε* στο πρόγραμμα την εντολή εισαγωγής της βιβλιοθήκης `random`.

```
import random
```

*Τροποποιήστε* την εντολή του βήματος 3, όπου ορίζεται η τιμή της μεταβλητής `secret`, ως εξής:

```
secret = random.randint(1,32)
```

Εκτελέστε το πρόγραμμα αρκετές φορές. Τί είδους τιμές παρατηρείτε ότι παίρνει η μεταβλητή `secret`;

*Η* `secret` *παίρνει τυχαίες τιμές.*

Ποιος πιστεύετε ότι είναι ο ρόλος των παραμέτρων `1` και `32`; Τί συμβαίνει αν δοκιμάσετε άλλες τιμές στη θέση τους, για παράδειγμα `33` και `64`; Αν χρειαστεί εκτελέστε πάλι το πρόγραμμα αρκετές φορές προκειμένου να απαντήσετε στην ερώτηση.

*Οι παράμετροι* `1` *και* `32` *καθορίζουν το διάστημα στο οποίο βρίσκεται η τυχαία τιμή. Αν χρησιμοποιήσουμε άλλους αριθμούς για παράδειγμα* `33` *και* `64`*, η τυχαία τιμή θα βρίσκεται μεταξύ των αριθμών* `33` *και* `64`*, συμπεριλαμβανομένων αυτών.*

16. Τί θα αλλάζατε στο πρόγραμμα όπως έχει μέχρι στιγμής; Πώς πιστεύετε ότι πρέπει να επεκταθεί για να γίνει πιο ενδιαφέρον;

*Ο παίκτης έχει μόνο μια ευκαιρία να μαντέψει το μυστικό αριθμό. Θα θέλαμε το πρόγραμμα να εκτελείται επαναληπτικά, ώστε να έχει περισσότερες ευκαιρίες στη διάθεσή του.*

## Γύρω-Γύρω Όλοι

17. *Προσθέστε* τη γραμμή που ακολουθεί *αμέσως πριν* από τις εντολές που προσθέσατε στο βήμα 8, όπου ζητείται από το χρήστη να μαντέψει τον μυστικό αριθμό.

```
while True:
```

*Προσθέστε* τέσσερα κενά μπροστά από *όλες τις εντολές που ακολουθούν* τη **while**, σηματοδοτώντας έτσι ότι αυτές οι εντολές *εμφωλεύονται* στη **while**, δηλαδή περιέχονται σε αυτήν.

Εκτελέστε το πρόγραμμα. Ποια αλλαγή παρατηρείτε ότι επιφέρει η χρήση της **while**;

*Οι εντολές που βρίσκονται μέσα στη **while** εκτελούνται ξανά και ξανά.*

Η εντολή του βήματος 15, που δίνει μια τυχαία τιμή στην μεταβλητή `secret`, βρίσκεται *πριν* από τη **while**. Ποια πιστεύετε ότι θα ήταν η διαφορά αν βρισκόνταν *μέσα* στη **while**;

*Σε κάθε γύρο του παιχνιδιού ο μυστικός αριθμός θα ήταν διαφορετικός, αφού η εντολή* `secret = random.randint(1,32)` *θα εκτελούνταν σε κάθε επανάληψη, αποδίδοντας κάθε φορά μια νέα τυχαία τιμή στο μυστικό αριθμό.*

Η εντολή του βήματος 13 που εμφανίζει τον μυστικό αριθμό στο χρήστη τώρα δημιουργεί πρόβλημα. Γιατί;

*Γιατί μετά την πρώτη αποτυχημένη προσπάθεια το πρόγραμμα εμφανίζει το μυστικό αριθμό στην οθόνη, επομένως στον επόμενο γύρο ο παίκτης θα γνωρίζει το μυστικό αριθμό και δεν θα χρειαστεί και πολλή προσπάθεια να τον μαντέψει!*

*Διαγράψτε* την εντολή που εμφανίζει τον μυστικό αριθμό.

Παίξτε το παιχνίδι μέχρι να μαντέψετε τον μυστικό αριθμό. Υπάρχει κάτι που σας ενοχλεί; Κάτι που φαίνεται να μη δουλεύει σωστά;

*Όταν ο παίκτης μαντέψει το μυστικό αριθμό το παιχνίδι συνεχίζεται, ενώ θα έπρεπε να σταματά.*

18. Η εντολή **break** διακόπτει την επανάληψη μέσα στην οποία βρίσκεται *αμέσως* μόλις εκτελεστεί. *Προσθέστε* την **break** στο σημείο που θεωρείτε κατάλληλο, έτσι ώστε το παιχνίδι να τερματίζεται όταν ο παίκτης μαντέψει τον αριθμό.

Παίξτε και πάλι το παιχνίδι μέχρι να μαντέψετε τον μυστικό αριθμό, για να επιβεβαιώσετε ότι τοποθετήσατε την **break** σε σωστό σημείο.

*Η εντολή **break** πρέπει να προστεθεί μέσα στην **if**, στην περίπτωση που ο παίκτης μαντέψει σωστά το μυστικό αριθμό, αφού τότε πρέπει να τερματιστεί η επανάληψη.*

```
    if number == secret:
        print("Σωστά!")
        break
```

```
    else:
        print("Λάθος...")
```

19. Για δοκιμαστικούς λόγους, κάτω από την **break** *προσθέστε* την εντολή:

```
print("Ζντονκ!")
```

Εκτελέστε πάλι το πρόγραμμα μέχρι να μαντέψετε τον αριθμό. Εμφανίζεται το μήνυμα `"Ζντονκ!"`; Γιατί πιστεύετε ότι συμβαίνει αυτό;

*Το μήνυμα δεν εμφανίζεται, αφού η εκτέλεση της εντολής* **break** *προκαλεί την άμεση έξοδο από την επανάληψη. Επομένως, οι εντολές που την ακολουθούν δεν θα εκτελεστούν ποτέ.*

*Αφαιρέστε* τώρα την εντολή που προσθέσατε.

20. Τί θα αλλάζατε στο πρόγραμμα όπως έχει μέχρι στιγμής; Πώς πιστεύετε ότι πρέπει να επεκταθεί για να γίνει πιο ενδιαφέρον;

*Προς το παρόν ο παίκτης μαντεύει στα τυφλά. Μια προσθήκη που θα τον βοηθούσε είναι να ενημερώνεται από το πρόγραμμα αν ο μυστικός αριθμός είναι μικρότερος ή μεγαλύτερος από αυτόν που έδωσε.*

## Επιλογές, Επιλογές

Θα επεκτείνουμε το πρόγραμμα έτσι ώστε να δίνει στο χρήστη περισσότερη πληροφορία: αντί να τον ενημερώνει απλά αν βρήκε τον μυστικό αριθμό ή όχι, θα τον κατευθύνει αν πρέπει να τον αναζητήσει ψηλότερα ή χαμηλότερα.

Για την επέκταση αυτή δεν αρκεί πια η απλή **if**–**else**, η οποία μπορεί να διακρίνει μόνο ανάμεσα σε δύο περιπτώσεις.

21. *Τροποποιήστε* την **if** που ελέγχει αν ο χρήστης βρήκε τον μυστικό αριθμό. Συμπληρώσετε την συνθήκη που λείπει:

```
    if secret == number:
        print("Σωστά!")
        break
    elif συνθήκη: # συμπληρώστε την συνθήκη
        print("Ο μυστικός αριθμός είναι μεγαλύτερος.")
    else:
        print("Ο μυστικός αριθμός είναι μικρότερος.")
```

*Η συνθήκη που θα συμπληρώσουμε είναι:*

```
elif secret > number :
```

*ώστε το πρόγραμμα να ελέγχει αν ο μυστικός αριθμός είναι μεγαλύτερος από τον αριθμό που έδωσε ο χρήστης.*

Εκτελέστε το πρόγραμμα και βεβαιωθείτε ότι λειτουργεί σωστά.

Σε ποια περίπτωση εκτελούνται οι εντολές της **else**; Γιατί πιστεύετε ότι δεν ελέγχουμε καμία συνθήκη σε αυτή την τρίτη περίπτωση;

*Οι εντολές της* **else** *εκτελούνται όταν ο μυστικός αριθμός είναι μικρότερος από τον αριθμό που έδωσε ο παίκτης. Δεν ελέγχουμε κάποια συνθήκη, αφού αυτή είναι η μοναδική, εναλλακτική περίπτωση.*

22. *Αναδιατάξτε* τις περιπτώσεις της **if** όπως φαίνεται παρακάτω. Και πάλι, θα πρέπει να συμπληρώσετε τις συνθήκες που ελέγχονται σε κάθε περίπτωση.

```
    if συνθήκη: # συμπληρώστε την συνθήκη
        print("Ο μυστικός αριθμός είναι μικρότερος.")
    elif συνθήκη: # συμπληρώστε την συνθήκη
        print("Ο μυστικός αριθμός είναι μεγαλύτερος.")
    else:
        print("Σωστά!")
        break
```

*Η συνθήκη που θα συμπληρώσουμε στην* **if** *είναι:*

```
if secret < number:
```

*Η συνθήκη που θα συμπληρώσουμε στην* **elif** *είναι:*

```
elif secret > number:
```

23. Τί θα αλλάζατε στο πρόγραμμα όπως έχει μέχρι στιγμής; Πώς πιστεύετε ότι πρέπει να επεκταθεί για να γίνει πιο ενδιαφέρον;

*Ο παίκτης έχει απεριόριστο αριθμό προσπαθειών, οπότε κάποια στιγμή θα μαντέψει το μυστικό αριθμό. Θα πρέπει να θέσουμε έναν μέγιστο αριθμό προσπαθειών προκειμένου να τα καταφέρει.*

## Μέτρημα

Ο αριθμός των προσπαθειών που διαθέτει ο παίκτης δεν θα έπρεπε να είναι απεριόριστος. Θα επεκτείνουμε το παιχνίδι ώστε να τερματίζεται όταν εξαντληθούν οι προσπάθειες του παίκτη.

24. *Προσθέστε* την εντολή που ακολουθεί αμέσως μετά τη **while**, δηλαδή στην αρχή της επανάληψης.

```
print("Απομένουν", tries, "προσπάθειες.")
```

Είναι εμφανές ότι η τιμή της μεταβλητής `tries` θα αντιστοιχεί στο πλήθος των προσπαθειών που απομένουν στον παίκτη.

Αν εκτελέσετε το πρόγραμμα όπως έχει θα εμφανιστεί μήνυμα λάθους, αφού επιχειρούμε να εμφανίσουμε την τιμή της `tries`, χωρίς πουθενά προηγουμένως να της έχουμε αποδώσει μια τιμή.

```
NameError: name 'tries' is not defined
```

25. Δώστε στην `tries` μια αρχική τιμή, ίση με το πλήθος των προσπαθειών που διαθέτει ο χρήστης όταν ξεκινά το παιχνίδι. Αποφασίστε εσείς μια τιμή που να σας φαίνεται "δίκαιη".

*Η εντολή που θα προσθέσουμε στο πρόγραμμα είναι:*

```
tries = 4
```

*δεδομένου ότι θα δώσουμε το πολύ τέσσερεις προσπάθειες στον παίκτη.*

Τοποθετήσατε τις νέες εντολές *πριν* την επανάληψη ή *μέσα* σε αυτή; Για ποιο λόγο κάνατε αυτή την επιλογή;

*Η εντολή πρέπει να προστεθεί πριν από τη* `while`*, ώστε η εκτέλεσή της να γίνει μόνο μια φορά. Διαφορετικά, οι προσπάθειες του παίκτη θα παίρνουν σε κάθε επανάληψη και πάλι την τιμή* `4` *αντί να μειώνονται.*

Ποιο είναι το αρχικό πλήθος προσπαθειών που δώσατε στον παίκτη και γιατί θεωρήσατε αυτή την τιμή "δίκαιη".

*Εφόσον ο παίκτης ακολουθήσει τη στρατηγική της διαίρεσης του εκάστοτε διαστήματος στη μέση, επιλέγοντας για παράδειγμα το* `16` *αρχικά και στη συνέχεια το μισό του διαστήματος που απομένει και ούτω καθεξής, θα χρειαστεί το πολύ* `5` *προσπάθειες για να μαντέψει τον μυστικό αριθμό. Δίνοντας του* `4` *προσπάθειες προσθέτουμε στο παιχνίδι το στοιχείο της τυχαιότητας, κάνοντας το πιο δίκαιο.*

Εκτελέστε το πρόγραμμα. Τί παρατηρείτε;

*Ο αριθμός των προσπαθειών δεν μειώνεται σε κάθε επανάληψη.*

26. Η μεταβλητή `tries` θα πρέπει να μειώνεται σε κάθε γύρο του παιχνιδιού, δηλαδή *εντός της επανάληψης*. Μετά την `print` του βήματος 24 που εμφανίζει την τιμή της `tries`, *προσθέστε* τη γραμμή:

```
    tries = tries - 1
```

Εκτελέστε το πρόγραμμα. Μειώνεται το πλήθος των προσπαθειών που απομένουν στον παίκτη;

*Ναι. Στην οθόνη εμφανίζεται σε κάθε γύρο ο νέος, μειωμένος αριθμός προσπαθειών.*

Αν το πλήθος των προσπαθειών δεν μειώνεται, βεβαιωθείτε ότι η εντολή που προσθέσατε στο βήμα 25 και δίνει αρχική τιμή στην `tries` βρίσκεται *πριν* από την επανάληψη και όχι μέσα σε αυτή.

Περιγράψτε πως ακριβώς πιστεύετε ότι λειτουργεί η εντολή που προσθέσατε για να μειώνεται η `tries`.

*Σε κάθε γύρο του παιχνιδιού ο αριθμός των προσπαθειών μειώνεται κατά* `1`*. Αυτό επιτυγχάνεται ως εξής: υπολογίζεται η τιμή της παράστασης* `tries - 1` *και το αποτέλεσμα ονομάζεται πάλι* `tries`*. Η νέα τιμή της μεταβλητής* `tries` *υπολογίζεται με βάση την τρέχουσα τιμή της, την οποία και αντικαθιστά. Επομένως, θα πάρει διαδοχικά τις τιμές* `3`*,* `2`*,* `1` *και* `0`*, δεδομένου ότι ο παίκτης δεν θα μαντέψει το μυστικό αριθμό.*

Υπάρχει κάτι που σας ενοχλεί και φαίνεται να μη δουλεύει σωστά;

*Το παιχνίδι δεν τερματίζει όταν οι προσπάθειες μηδενιστούν. Αντίθετα, συνεχίζει δίνοντας αρνητικές τιμές στη μεταβλητή* `tries`*.*

## Τερματισμός

Το παιχνίδι δεν πρέπει να τερματίζεται μόνο όταν ο παίκτης μαντέψει τον αριθμό, αλλά και όταν τελειώσουν οι προσπάθειές του.

27. Η `while` συνοδεύεται από μια *συνθήκη*. Στην *αρχή* κάθε κύκλου, η συνθήκη αυτή ελέγχεται εκ νέου. Αν η συνθήκη είναι αληθής τότε η επανάληψη συνεχίζεται για άλλον έναν κύκλο.

    Εμείς χρησιμοποιήσαμε μέχρι τώρα την τετριμμένη συνθήκη `True`, η οποία είναι πάντα αληθής, γι' αυτό και η επανάληψη δεν διακόπτονταν λόγω της συνθήκης.

    *Αντικαταστήστε* τη συνθήκη `True` με τη συνθήκη `tries > 0`, που είναι αληθής μόνο όταν απομένουν κι άλλες προσπάθειες στον παίκτη. Σε περίπτωση που αυτό δεν ισχύει, η επανάληψη θα διακοπεί.

    ```
    while tries > 0:
    ```

    Εκτελέστε και πάλι το πρόγραμμα και διερευνήστε τη συμπεριφορά του. Λειτουργεί σωστά ή εντοπίζετε προβλήματα;

    *Ναι, λειτουργεί σωστά. Παρατηρούμε ότι όταν ο παίκτης δεν μαντέψει τον αριθμό το πολύ σε τέσσερεις προσπάθειες, τότε το παιχνίδι σταματά.*

28. Για δοκιμαστικούς λόγους, *προσθέστε* αμέσως κάτω από την εντολή `tries = tries - 1` τη γραμμή:

    ```
    print("Ζντονκ!", tries)
    ```

    Εκτελέστε το πρόγραμμα μέχρι να εξαντληθούν οι προσπάθειες. Εμφανίζεται στο τέλος το μήνυμα `"Ζντονκ! 0"`;

    *Ναι, εμφανίζεται.*

    Ενώ η συνθήκη της `while` είναι `tries > 0`, από το μήνυμα φαίνεται ότι η εκτέλεση των εντολών της επανάληψης δεν διακόπτεται *άμεσα* όταν μηδενιστεί η μεταβλητή `tries` και η συνθήκη πάψει να ισχύει. Άρα η συνθήκη `tries > 0` της `while` δεν ελέγχεται συνεχώς αλλά μόνο στην αρχή κάθε νέου κύκλου της επανάληψης.

    *Αφαιρέστε* τώρα την εντολή που προσθέσατε.

29. Αν ο παίκτης εξαντλήσει τις προσπάθειές του χωρίς να βρει τον αριθμό τότε χάνει και το παιχνίδι σταματά. *Προσθέστε* τις κατάλληλες εντολές στο πρόγραμμα έτσι ώστε, στην περίπτωση αυτή, να εμφανίζει στον παίκτη τον αριθμό που αναζητούσε, για παράδειγμα:

    ```
    Ο μυστικός αριθμός ήταν ο 13.
    ```

    Ουσιαστικά θα επανεισάγετε την εντολή που αφαιρέσατε στο βήμα 17, η οποία εμφανίζει τον μυστικό αριθμό. Θα πρέπει να την τοποθετήσετε στο κατάλληλο σημείο του προγράμματος.

    Φροντίστε να εμφανίζεται το μήνυμα *μόνο όταν είναι απαραίτητο*: αν ο παίκτης βρει τον μυστικό αριθμό, τότε το μήνυμα δε χρειάζεται.

*Στο σημείο αυτό, όταν πλέον έχει τερματιστεί η επανάληψη, μπορούμε να σκεφτούμε δύο πιθανές εκδοχές για να ελέγξουμε αν ο παίκτης έχει χάσει. Η μια είναι να εξετάσουμε αν οι προσπάθειες έχουν μηδενιστεί. Η άλλη είναι να εξετάσουμε αν ο αριθμός που έδωσε ο παίκτης (στην τελευταία προσπάθεια) δεν είναι ίδιος με το μυστικό.*

*Παρόλο που και οι δύο μοιάζουν σωστές, όπως θα διαπιστώσουμε λίγο παρακάτω, μόνο η δεύτερη λειτουργεί με τον τρόπο που θέλουμε. Επομένως, μετά το τέλος των εντολών της επανάληψης, προσθέτουμε:*

```
if number != secret:
    print("Ο μυστικός αριθμός ήταν ο", secret)
```

Εκτελέστε το πρόγραμμα. Λειτουργεί σωστά στην περίπτωση που ο παίκτης μαντέψει τον αριθμό; Μήπως στο τέλος του παιχνιδιού του εμφανίζει τον μυστικό αριθμό, παρόλο που τον έχει βρει;

*Ναι, λειτουργεί σωστά. Ο μυστικός αριθμός εμφανίζεται μόνο στην περίπτωση που ο παίκτης αποτύχει.*

Σε αυτή την περίπτωση, *διορθώστε* το πρόγραμμα.

Το πρόγραμμα λειτουργεί σωστά στην περίπτωση που ο παίκτης μαντεύει τον αριθμό *στην τελευταία του προσπάθεια*; Μήπως του εμφανίζει και πάλι τον μυστικό αριθμό, παρόλο που τον έχει βρει;

*Ναι, λειτουργεί, αφού οι εντολές της* `if` *δεν μπορούν να εκτελεστούν όταν ο παίκτης έχει μαντέψει το μυστικό αριθμό.*

Αν η απάντηση ήταν καταφατική, μάλλον προσπαθείτε να διαπιστώσετε αν ο παίκτης έχασε ελέγχοντας την συνθήκη `tries == 0`. Σκεφτείτε όμως: αν ο παίκτης μαντέψει τον αριθμό στην τελευταία του προσπάθεια τότε θα ισχύει ότι `tries == 0` όμως ο παίκτης δεν θα έχει χάσει. Χρειάζεται να διορθώσετε το πρόγραμμα ελέγχοντας με *διαφορετική συνθήκη* αν ο παίκτης απέτυχε να μαντέψει τον αριθμό.

## Περισσότερη Βοήθεια

**30.** Εκτελέστε το πρόγραμμα. Στην πρώτη σας προσπάθεια δοκιμάστε τον αριθμό 13. Ο μυστικός αριθμός είναι μικρότερος ή μεγαλύτερος;

Αν ο μυστικός αριθμός είναι το 13, απλά εκτελέστε και πάλι το πρόγραμμα.

*Δοκιμάζουμε τον αριθμό* 13 *και το πρόγραμμα εμφανίζει μήνυμα ότι ο μυστικός αριθμός είναι μεγαλύτερος από* 13.

Σε ποιο διάστημα θ' αναζητήσετε τώρα τον μυστικό αριθμό, δηλαδή ποια είναι η ελάχιστη και ποια η μέγιστη δυνατή τιμή που γνωρίζετε τώρα ότι μπορεί να έχει ο μυστικός αριθμός;

*Η μικρότερη δυνατή τιμή του μυστικού αριθμού είναι το* 14*, αφού ο μυστικός αριθμός είναι μεγαλύτερος του* 13*, σύμφωνα με το προηγούμενο, ενώ η μεγαλύτερη δυνατή τιμή παραμένει το* 32.

**31.** Στη δεύτερη προσπάθεια, δοκιμάστε έναν αριθμό που ανήκει στο διάστημα που απαντήσατε προηγουμένως.

Συνεχίστε μέχρι να τελειώσει το παιχνίδι, συμπληρώνοντας τον πίνακα που ακολουθεί. Σημειώστε σε κάθε βήμα τον αριθμό που δοκιμάσατε, την απάντηση του προγράμματος και το διάστημα μέσα στο οποίο "εγκλωβίσατε" κάθε φορά τον μυστικό αριθμό. Το διάστημα αυτό ορίζεται από την ελάχιστη (`low`) και τη μέγιστη (`high`) δυνατή τιμή που έχει νόημα να δοκιμάσετε μετά από κάθε προσπάθεια.

| *αριθμός* `number` | *ο μυστικός είναι* *(μικρότερος / μεγαλύτερος)* | *ελάχιστη* `low` | *μέγιστη* `high` |
|---|---|---|---|
| 13 | *μεγαλύτερος* | 14 | 32 |
| 23 | *μικρότερος* | 14 | 22 |
| 18 | *μικρότερος* | 14 | 17 |
| 15 | *ίσος* | ..... | ..... |

**32.** Τώρα θα επεκτείνουμε το πρόγραμμα έτσι ώστε να βοηθάει το χρήστη ακόμα περισσότερο. Θα χρησιμοποιήσουμε δύο μεταβλητές `low` και `high`, οι οποίες αντιστοιχούν στην ελάχιστη και τη μέγιστη δυνατή τιμή που γνωρίζουμε ότι μπορεί να έχει ο μυστικός αριθμός.

Στην αρχή του προγράμματος, αποδώστε αρχικές τιμές σε αυτές τις μεταβλητές:

```
low = 1
high = 32
```

**33.** Αμέσως πριν από την `input()` με την οποία το πρόγραμμα διαβάζει από το χρήστη έναν αριθμό, *προσθέστε* μια εντολή που εμφανίζει στο χρήστη τα `low` και `high`, για να τον βοηθά στην επιλογή του. Για παράδειγμα, αν τα `low` και `high` είναι αντίστοιχα `14` και `23`, τότε να εμφανίζει:

```
Δοκίμασε ανάμεσα στο 14 και το 23.
```

*Η εντολή που θα προσθέσουμε στο πρόγραμμα είναι:*

```
print("Δοκίμασε ανάμεσα στο", low, "και το", high)
```

Εκτελέστε το πρόγραμμα. Υπάρχει κάτι που σας ενοχλεί και φαίνεται να μη δουλεύει σωστά;

*Ναι, το πρόγραμμα εμφανίζει σε κάθε επανάληψη το μήνυμα:*

```
Δοκίμασε ανάμεσα στο 1 και στο 32 ,
```

*δηλαδή δεν μεταβάλλει κατάλληλα τις τιμές των* `low` *και* `high`*.*

**34.** Στο πρόγραμμα *υπάρχει ήδη* μια **`if`** που ελέγχει αν ο μυστικός αριθμός είναι μικρότερος από τον αριθμό του χρήστη:

```
    if secret < number:
        print("Ο μυστικός αριθμός είναι μικρότερος.")
```

Στην περίπτωση αυτή, όπως φαίνεται κι από τον πίνακα που συμπληρώσατε στο βήμα 31, *μόνο μία* από τις μεταβλητές `low` και `high` χρειάζεται ν' αλλάξει τιμή. Ποιά από τις δύο;

*Στην περίπτωση που ο μυστικός αριθμός είναι μικρότερος πρέπει να αλλάξουμε την τιμή της μεταβλητής* `high`*, όπως παρακάτω:*

```
    if secret < number:
        print("Ο μυστικός αριθμός είναι μικρότερος.")
        high = number - 1
```

*Αφού ο μυστικός αριθμός είναι μικρότερος από τον αριθμό που έδωσε ο παίκτης τότε η μεγαλύτερη δυνατή τιμή που μπορεί να έχει, δηλαδή η τιμή της* `high`*, είναι τουλάχιστον μικρότερη κατά 1 από τον αριθμό αυτό.*

Να *προσθέσετε* σε αυτή την περίπτωση της **`if`** μια εντολή που μεταβάλλει κατάλληλα την τιμή της `low` ή της `high`. Αν δυσκολευτείτε, ανατρέξτε στον πίνακα που συμπληρώσατε στο βήμα 31.

Στο πρόγραμμα υπάρχει επίσης μια **`if`** που ελέγχει αν ο μυστικός αριθμός είναι μεγαλύτερος από τον αριθμό του χρήστη:

```
    elif secret > number:
        print("Ο μυστικός αριθμός είναι μεγαλύτερος.")
```

Να *προσθέσετε* και σε αυτή την περίπτωση την εντολή που μεταβάλλει κατάλληλα την τιμή της `low` ή της `high`.

*Στην περίπτωση που ο μυστικός αριθμός είναι μεγαλύτερος πρέπει να αλλάξουμε την τιμή της μεταβλητής* `low`*, όπως παρακάτω:*

```
    elif secret > number:
        print("Ο μυστικός αριθμός είναι μεγαλύτερος.")
        low = number + 1
```

*Αφού ο μυστικός αριθμός είναι μεγαλύτερος από τον αριθμό που έδωσε ο παίκτης τότε η μικρότερη δυνατή τιμή που μπορεί να έχει, δηλαδή η τιμή της* `low`*, είναι τουλάχιστον μεγαλύτερη κατά 1 από τον αριθμό αυτό.*

35. Εκτελέστε το πρόγραμμά σας αρκετές φορές και ελέγξτε το διεξοδικά, για να διαπιστώσετε αν οι τιμές των μεταβλητών `low` και `high` μεταβάλλονται σωστά.

    Λειτουργεί σωστά το πρόγραμμα; Υπάρχει κάποια περίπτωση στην οποία η συμπεριφορά του είναι προβληματική;

*Στην περίπτωση που ο παίκτης δώσει τιμές εκτός των ορίων* `low` *και* `high`*, το πρόγραμμα δε λειτουργεί σωστά.*

36. **Επέκταση**: Το βήμα αυτό είναι προαιρετικό.

    Εκτελέστε ακόμα μερικές φορές το πρόγραμμα. Δοκιμάστε, ως παίκτης, να δίνετε τιμές που είναι *εκτός των ορίων* που προτείνει το πρόγραμμα. Τί παρατηρείτε; Δημιουργείται πρόβλημα;

    

    *Ναι, δημιουργείται. Το πρόγραμμα αναπροσαρμόζει τις τιμές των ορίων με λάθος τρόπο. Για παράδειγμα, αν στην πρώτη προσπάθεια ο παίκτης δώσει τον αριθμό* `40` *τότε η τιμή της* `high` *θα γίνει* `39`.

    Αν υπάρχει πρόβλημα όταν ο χρήστης δίνει τιμές εκτός του διαστήματος που ορίζουν οι `low` και `high`, προσθέστε τους κατάλληλους ελέγχους στο πρόγραμμα ώστε να διορθωθούν τυχόν σφάλματα.

    

    *Τα όρια θα πρέπει να προσαρμόζονται μόνο όταν ο αριθμός* `number` *που δίνει ο χρήστης είναι εντός αυτών των ορίων. Επομένως, οι έλεγχοι που θα προσθέσουμε στο πρόγραμμα είναι:*

```
if secret < number:
    print("Ο μυστικός αριθμός είναι μικρότερος.")
    if number <= high:
        high = number - 1
elif secret > number:
    print("Ο μυστικός αριθμός είναι μεγαλύτερος.")
    if number >= low:
        low = number + 1
```