

Ατομική Εργασία

(αντικειμενοστραφής προγραμματισμός)

Ονοματεπώνυμο: Κρότορ Καταρτζίου Ιωάν

Α.Μ.: Π21077 (εξάμηνο 2)

Η εργασία που υλοποίησα αποτελεί ένα δείγμα της γνώσης πάνω στη γλώσσα προγραμματισμού java. Αρχικά, καλύπτεται το θέμα που ζητείται το οποίο είναι εφαρμογή διαχείρισης ζωολογικού κήπου στην οποία καταγράφονται τα ζώα. Το πρόγραμμα περιλαμβάνει μία βασική κλάση `Animal`, η οποία είναι `abstract` εφόσον δεν δημιουργεί στιγμιότυπα και στην οποία ορίζονται τα ζητούμενα πεδία (`id`, `name`, `homotaxy`, `weight`, `age`) τα οποία αρχικοποιούνται μέσω ενός `constructor`, εκτός από το `id`, το οποίο δηλώνεται αυτόματα με τη προσθήκη νέου ζώου. Επιπλέον, υπάρχουν 13 κλάσεις διαφορετικών ζώων οι οποίες κληρονομούν την `Animal` και επιπλέον η κάθε κλάση (ζώο) έχει ένα μοναδικό, ξεχωριστό χαρακτηριστικό.

Όσον αφορά τις διάφορες λειτουργίες που ζητούνται, δημιούργησα μία ξεχωριστή κλάση `Zoo`, στην οποία υλοποιείται το `menu` που εμφανίζεται στην κονσόλα, η οποία διαχειρίζεται μέσω `switch` και παρέχει 9 + 1 λειτουργίες

1. Εμφάνιση των ζώων που διατίθενται
2. Εισαγωγή νέου ζώου
3. Αναζήτηση ζώου βάσει ονόματος
4. Αναζήτηση ζώου βάσει κωδικού
5. Επεξεργασία ζώου βάσει κωδικού
6. Διαγραφή ζώου βάσει κωδικού
7. Παίξτε με τα ζώα

8. Πληροφορίες για τα ζώα (Wikipedia) και εμφάνιση ascii απεικόνισή τους
9. Έξοδος
10. (extra) εμφάνιση ενός ascii τοπίου εάν εισαχθεί ο αριθμός 141 (τα πρώτα 3 ψηφία του αριθμού π)

Η διάφορες λειτουργίες επιτυγχάνονται μέσω διάφορων βοηθητικών συναρτήσεων, η πλειοψηφία των λειτουργιών τους βασίζεται σε διάσχιση λίστας μέσω `foreach loop`, αφού όλα τα ζώα αποθηκεύονται σε ένα `static` (αφού είναι κοινό για όλα τα ζώα) `Array List` τύπου `Animal` εφόσον είναι υπερκλάση για κάθε μία από τις κλάσεις ζώων. Οι επιμέρους λειτουργίες αναλύονται περαιτέρω μέσα στα σχόλια του κώδικα.

Επιπλέον, υπάρχει και μία διεπαφή (`interface`), την οποία υλοποιεί η `Animal` και επομένως και η υπόλοιπες κλάσεις αφού επεκτείνουν (`extends`) την `Animal`. Η διεπαφή περιέχει δύο μεθόδους: μία για τον ήχο του κάθε ζώου και μία η οποία εμφανίζει τη σπανιότητα του ζώου στον συγκεκριμένο ζωολογικό κήπο με βάση το πόσα στιγμιότυπα της κλάσης του εκάστοτε ζώου έχουμε δημιουργήσει.

Επιπροσθέτως, οι περισσότερες μέθοδοι είναι `static` εφόσον καλούνται στη `main` η οποία είναι `static` και για αυτό κάθε φορά που καλείται μια `static` μέθοδος δημιουργώ ένα ανώνυμο αντικείμενο το οποίο δεν χρησιμοποιείται παρά μόνο για τη μέθοδο και για αυτό δεν αποθηκεύω κάποια αναφορά του

Συνεχίζοντας για τη προβολή στην κονσόλα χρησιμοποίησα το παρακάτω try-catch block:

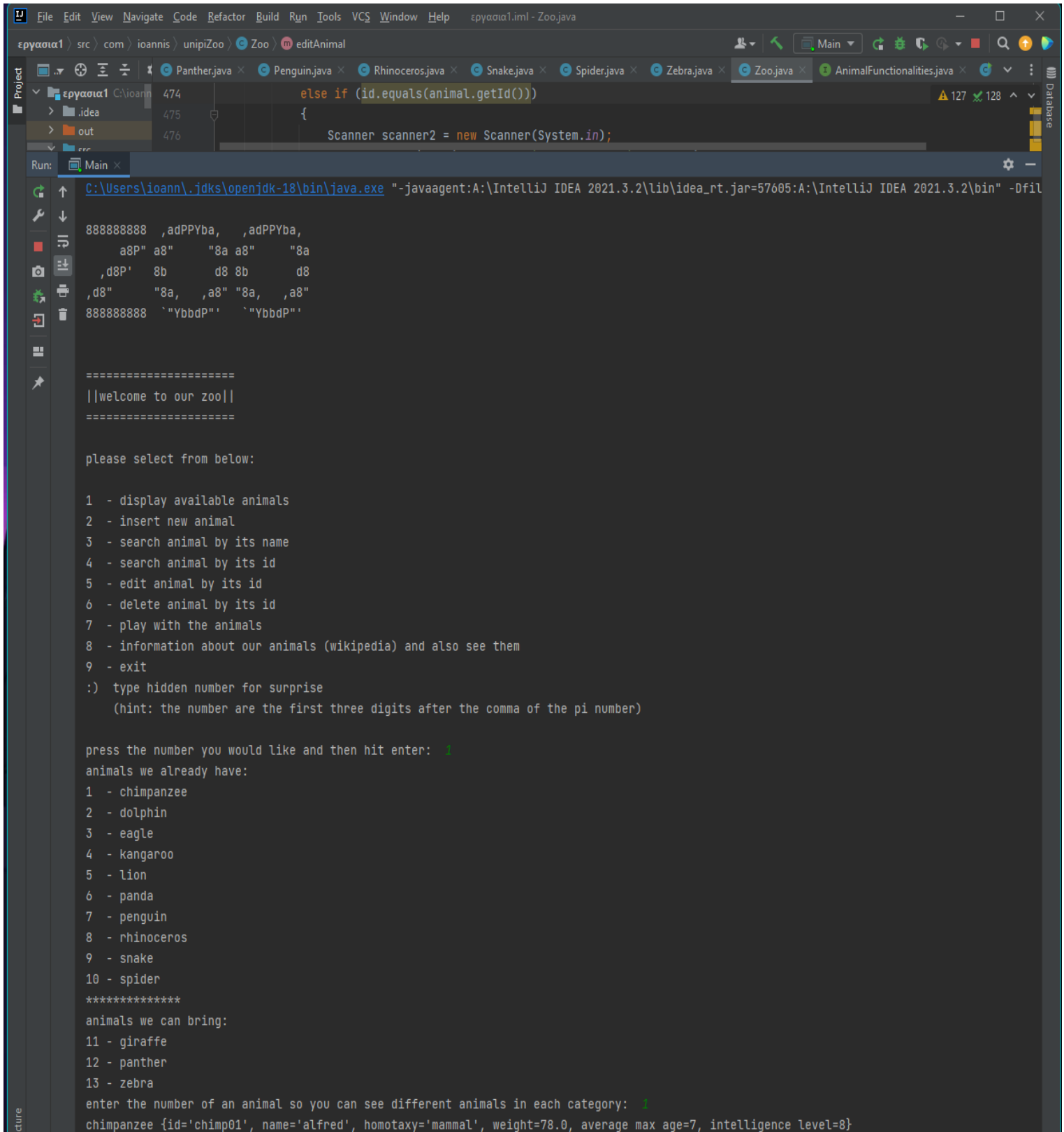
```
try {  
    Thread.sleep(5000);  
} catch (InterruptedException e) {  
    e.printStackTrace();  
}
```

το οποίο σταματάει την εκτέλεση του τρέχοντος πυρήνα για τόσα milliseconds όσα του ορίσουμε. Με αυτόν τον τρόπο κατάφερα να δείχνω τις πληροφορίες για συγκεκριμένο χρόνο στη κονσόλα, πριν συνεχίσει η εκτέλεση του προγράμματος το οποίο βρίσκεται σε ένα while(true) βρόχο που τερματίζεται μόνο εάν ο χρήστης εισάγει λάθος πληροφορία σε πεδίο η πατήσει το 9 στο μενού το οποίο τερματίζει το πρόγραμμα. Κάθε φορά που τρέχει το πρόγραμμα δημιουργούνται 10 ενδεικτικά στιγμιότυπα ζώων, κάθε ένα από διαφορετική κλάση.

Τέλος, ο κώδικας είναι αρκετά μεγάλος (~3000 γραμμές) λόγω των πολλών κλάσεων με διαφορετικά χαρακτηριστικά η κάθε μία οπότε η κάθε μία χρειάστηκε διαφορετική διαχείριση.

Δεν χρησιμοποίησα GUI καθώς αφιέρωσα πολύ χρόνο στο να φτιάξω μία ολοκληρωμένη εμπειρία κονσόλας, ενώ παράλληλα ήθελα να αναδείξω την δυνατότητα της ascii τέχνης όπως και της αισθητικής μου, για αυτό όλες οι ροές εξόδου (οι εμφανίσεις στη κονσόλα) είναι γραμμένες με μικρά γράμματα και καθόλου κεφαλαία (προσωπική προτίμηση).

screenshots



```
File Edit View Navigate Code Refactor Build Run Tools VCS Window Help εργασία1.iml - Zoo.java
εργασία1 src > com > ioannis > unipiZoo > Zoo > editAnimal
Project εργασία1 C:\ioannis 474 else if (id.equals(animal.getId()))
> .idea 475 {
> out 476 Scanner scanner2 = new Scanner(System.in);
Run: Main
C:\Users\ioannis\jdk\openjdk-18\bin\java.exe "-javaagent:A:\IntelliJ IDEA 2021.3.2\lib\idea_rt.jar=57605:A:\IntelliJ IDEA 2021.3.2\bin" -Dfile.encoding=UTF-8
8888888888 ,adPPYba, ,adPPYba,
a8P" a8" "8a a8" "8a
,d8P' 8b d8 8b d8
,d8" "8a, ,a8" "8a, ,a8"
8888888888 `Yb bdP" `Yb bdP"

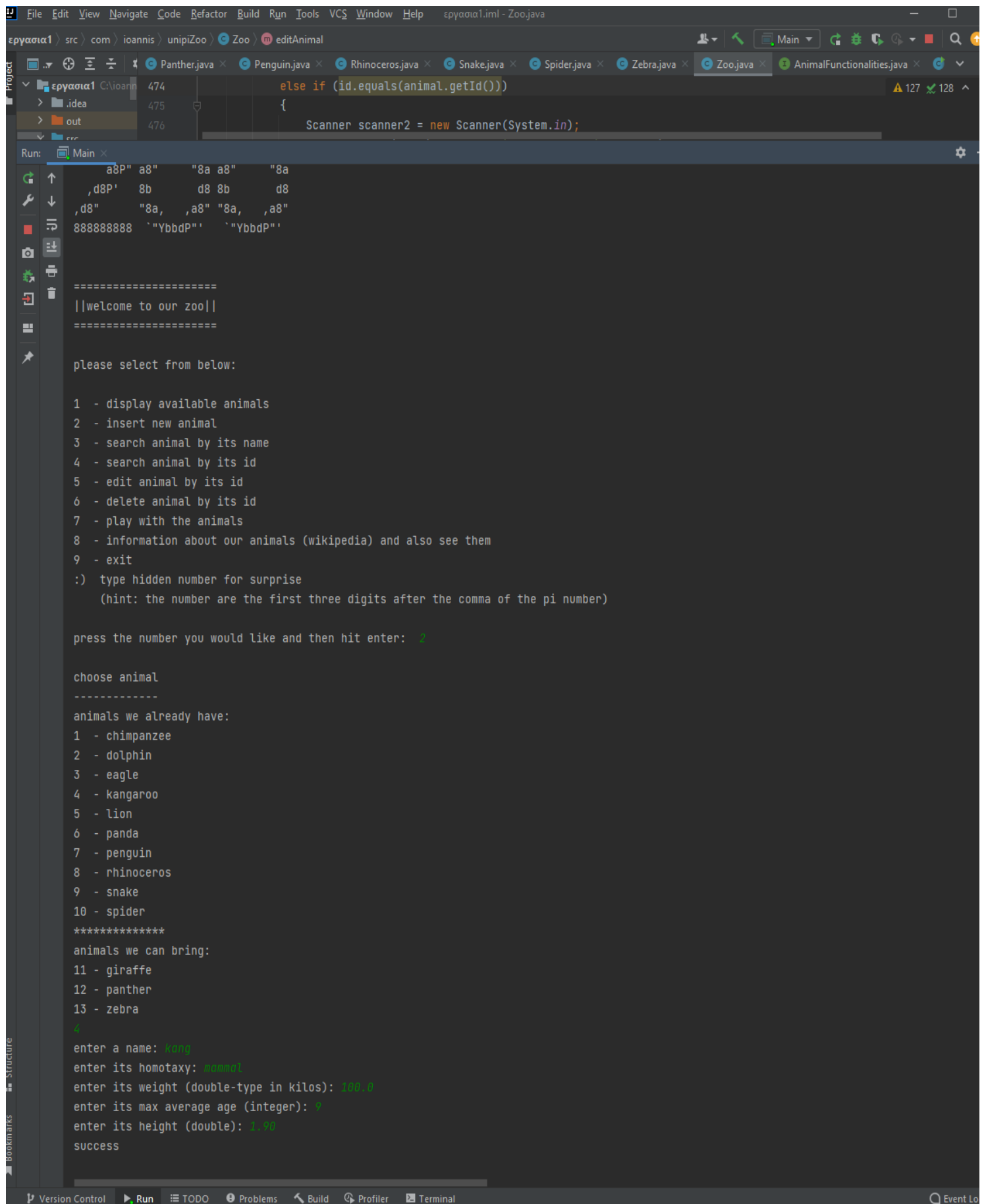
=====
||welcome to our zoo||
=====

please select from below:

1 - display available animals
2 - insert new animal
3 - search animal by its name
4 - search animal by its id
5 - edit animal by its id
6 - delete animal by its id
7 - play with the animals
8 - information about our animals (wikipedia) and also see them
9 - exit
:) type hidden number for surprise
(hint: the number are the first three digits after the comma of the pi number)

press the number you would like and then hit enter: 1
animals we already have:
1 - chimpanzee
2 - dolphin
3 - eagle
4 - kangaroo
5 - lion
6 - panda
7 - penguin
8 - rhinoceros
9 - snake
10 - spider
*****
animals we can bring:
11 - giraffe
12 - panther
13 - zebra

enter the number of an animal so you can see different animals in each category: 1
chimpanzee {id='chimp01', name='alfred', homotaxy='mammal', weight=78.0, average max age=7, intelligence level=8}
```



please select from below:

- 1 - display available animals
- 2 - insert new animal
- 3 - search animal by its name
- 4 - search animal by its id
- 5 - edit animal by its id
- 6 - delete animal by its id
- 7 - play with the animals
- 8 - information about our animals (wikipedia) and also see them
- 9 - exit

:) type hidden number for surprise

(hint: the number are the first three digits after the comma of the pi number)

press the number you would like and then hit enter: 3

enter the name of the animal you want to search: *alfred*

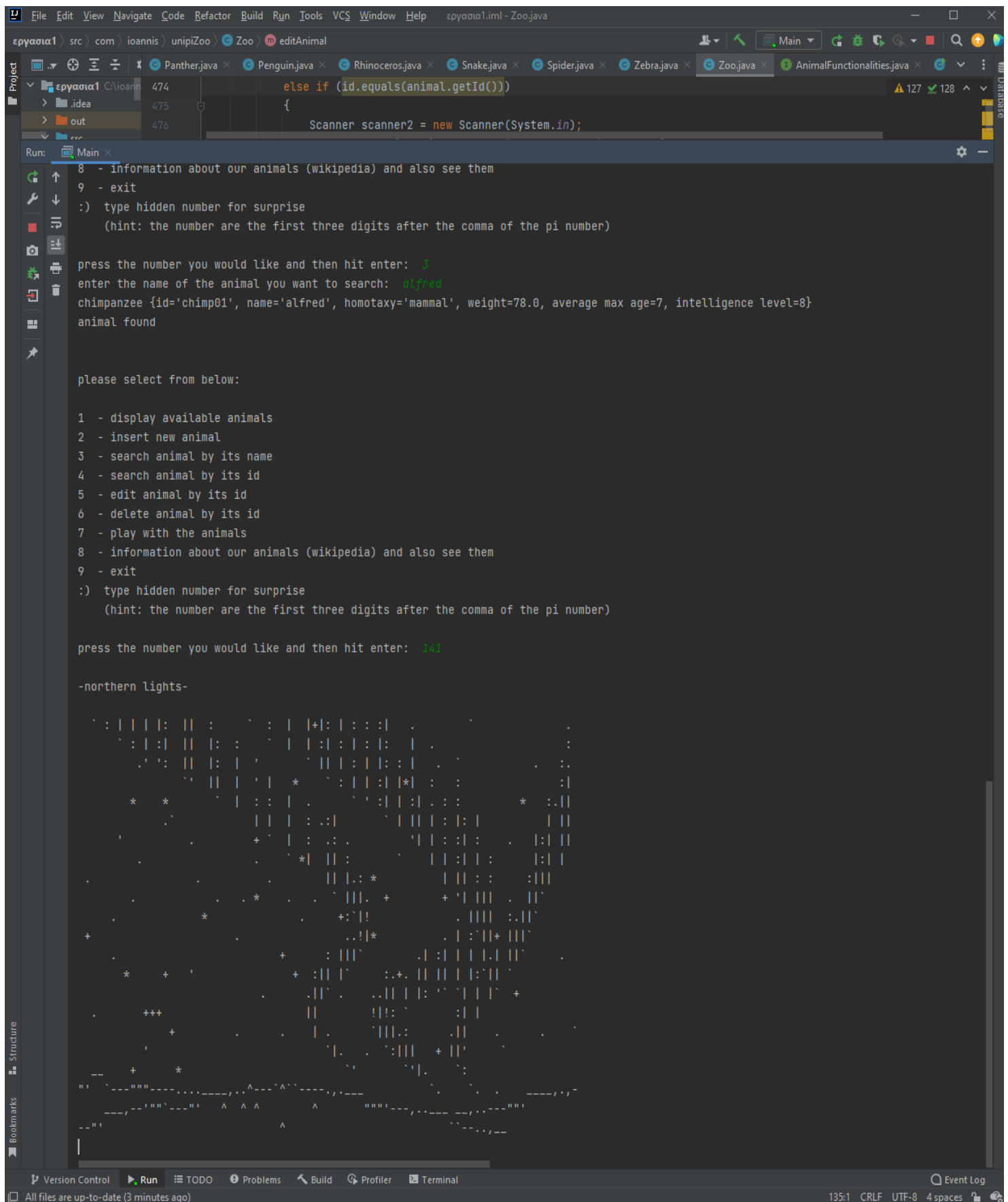
chimpanzee {id='chimp01', name='alfred', homotaxy='mammal', weight=78.0, average max age=7, intelligence level=8}

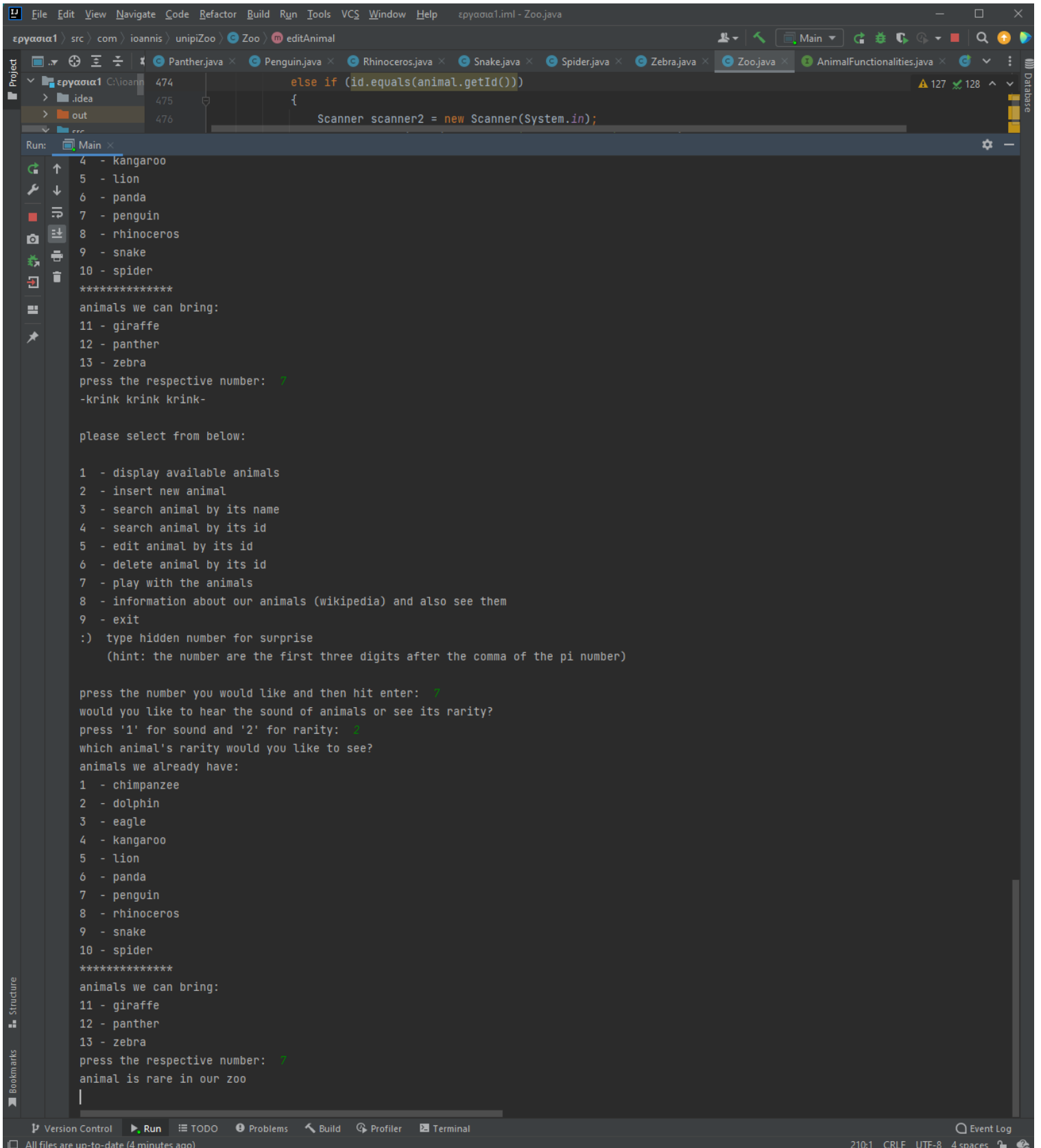
Run | TODO | Problems | Build | Profiler | Terminal

Event Log

up-to-date (2 minutes ago)

92:1 CRLF UTF-8 4 spaces





[illegible]