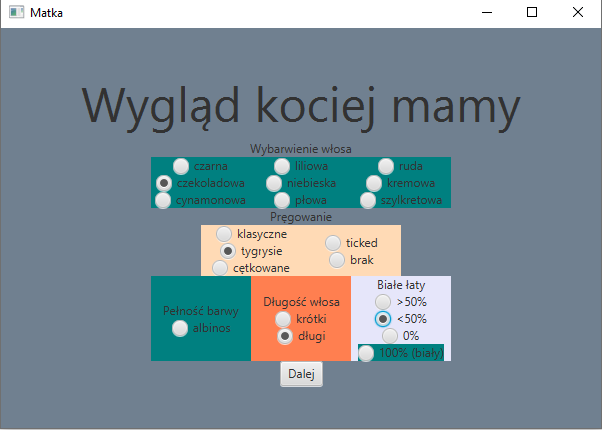
**Dla nru albumu 66459**

*Projekt dotyczy określenia wyglądu kociąt po podaniu cech matki i io jca.*

**Ekrany:**

1) wygląd matki

W tym ekranie użytkownik zaznacza pola określające wygląd matki kotów.

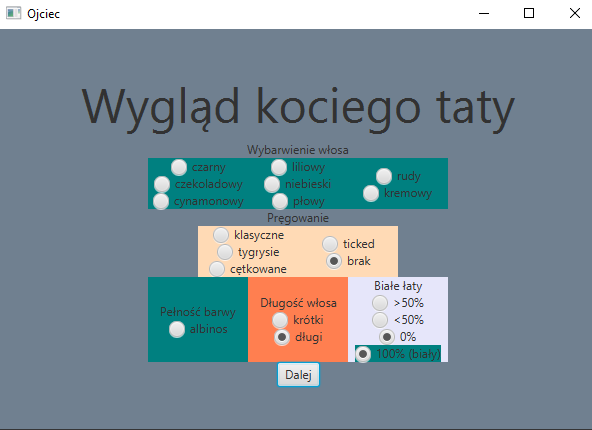


Rys. 1

Cechy są ułożone według locus, czyli B - czarny, D - ciemny, O - rudy, Mc - tackled, Ta - pręgowanie, A - widoczność pręgowania, C - pełne wybarwienie, L - długość sierści, S - łaty, W - białość, zaś pokolorowane według grupy, czyli wybarwienie - B, D, O, C, W; pręgowanie - Mc, Ta, A; długośc sierści - L; łaty - S.

2) wygląd ojca

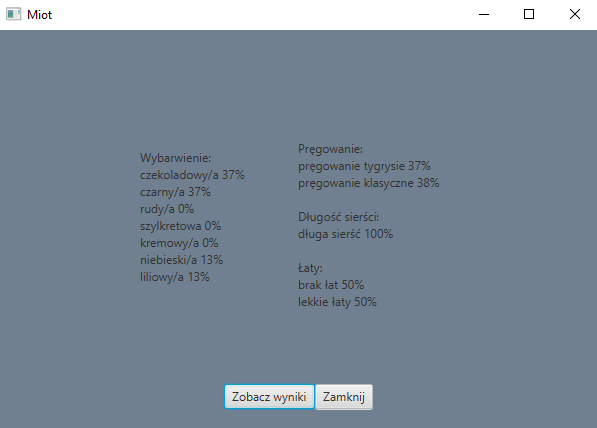
W tym ekranie użytkownik zaznacza pola określające wygląd ojca kotów.



Rys. 2

3) podsumowanie miotu

W tym ekranie użytkownik przelicza zaznaczone cechy na prawdopodobieństwo wystąpienia którejś z nich w miocie. Po wciśnięciu przycisku "Zobacz wyniki" przeliczane są obecne stany ankiet, czyli jeśli nie zamknięto poprzednich okien, można wprowadzać zmiany w cechach rodziców i liczyć inne pary. Przycisk "Zamknij" zamyka wszystkie okna.



Rys. 3

**Opis klas, metod i zmiennych**

1. Gene  
   Pola: String name, String allel1, String allel2  
   Metody:

* String getAllel1() - getter jednego z alleli
* String getAllel2() - getter jednego z alleli
* boolean isItSo(String a1, String a2) - metoda porównująca podaną parę alleli z allelami bieżącego obiektu

1. Cat

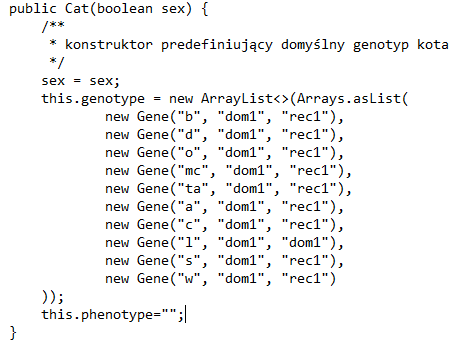
Pola: String phenotype - opis, List<Gene> genotype – zbiór genów, boolean sex – płeć



Rys. 4

Cat mother i Cat father – obiekty na których prowadzone są wszystkie obliczenia genetyczne w programie

Metody:

* Cat(boolean sex) – konstruktor predefiniujący domyślny genotyp kota
* 

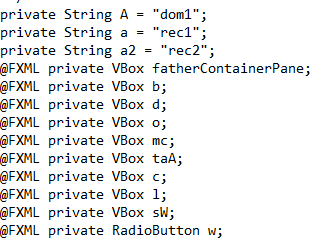
Rys. 5

* Gene getGene(int ind) - getter konkretnego genu z genotypu wskazany indeksem
* String getPhenotype() - ta metoda zwraca tylko wygenerowany wcześniej fenotyp
* void setGene(int ind, Gene gene) - metoda ustawiająca dany gen na danym miejscu
* void setPhenotype() - setter tłumaczący genotyp na fenotyp
* void setSex(boolean sex) - metoda zmieniająca płeć
* boolean wA(int ind, int alNum, String allel, Cat cat) - metoda sprawdzająca czy w genie genotype(ind) występuje podany allel w danym porządku
* boolean zigChk(int ind, String a, String b, String c, String d) - metoda sprawdzająca allele podanego genu matki i ojca

List<Map<String, Integer>> intercourse() – metoda zwracająca cechy w miocie i ich prawdopodobieństwo

1. MotherPollView i FatherPollView

Pola:

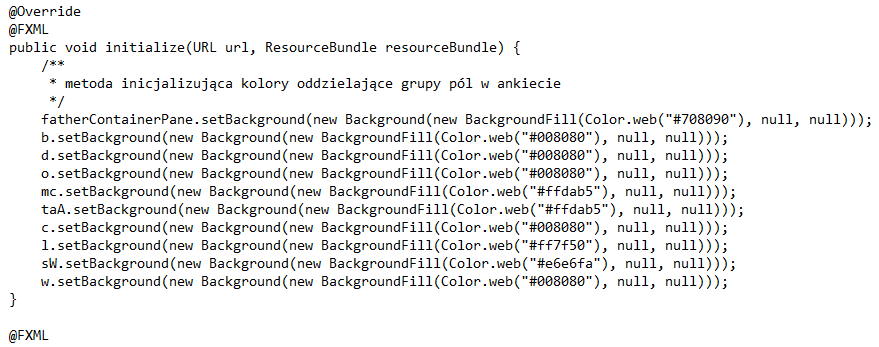


Rys. 6

Trzy stałe alleli i fx:id do zmiany koloru

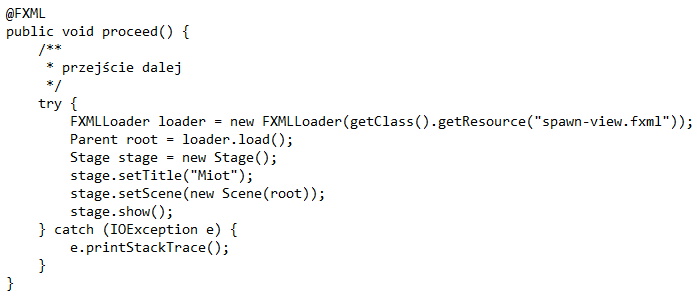
Metody:

* initialize() – metoda ustawiająca kolory widoku

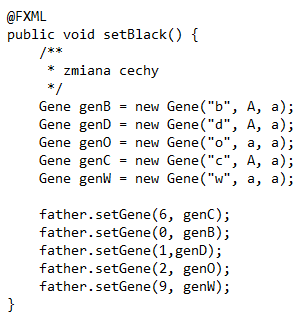


Rys. 7

* proceed() – metoda zmieniająca ekran (u matki na ojca a u ojca na miot)



Rys. 8

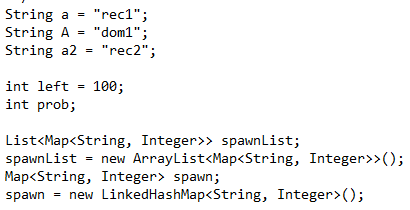
* metody ustawiające cechy
* 

Rys. 9

**Opis działania**

W kolejnych dwóch ankietach użytkownik opisuje dwa osobniki kota – ojca i matkę obliczanych kociąt – za pomocą radiobuttonów oznaczonych odpowiednimi cechami. Cechy są zapisywane automatycznie po kliknięciu radiobuttona i są całkowicie zmienialne w czasie działania programu. Cechy rodziców są zczytywane w momencie kliknięcia przycisku „Zobacz wyniki”.

Działanie metody Cat.intercourse()



Rys. 10

String a – zmienna oznaczająca allel recesywny, A – dominujący, a1 – recesywny wobec recesywnego

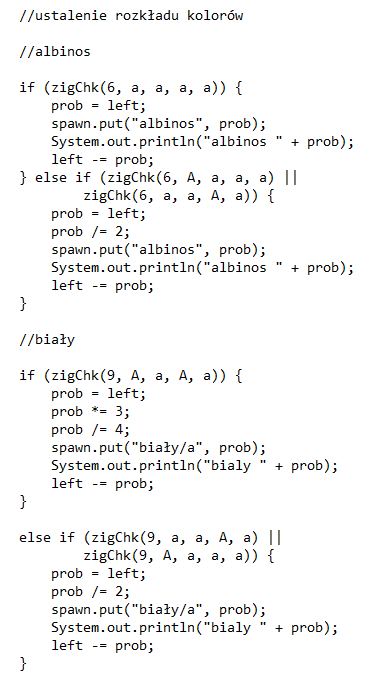
int left – zmienna wskazująca ile procent miotu zostało jeszcze do przydziału

int prob – zmienna wskazująca prawdopodobieństwo wpisywanej cechy

List<Map<String, Integer>> spawnList – większa lista, z czterema mapami, po wypełnieniu zwracana przez funkcje

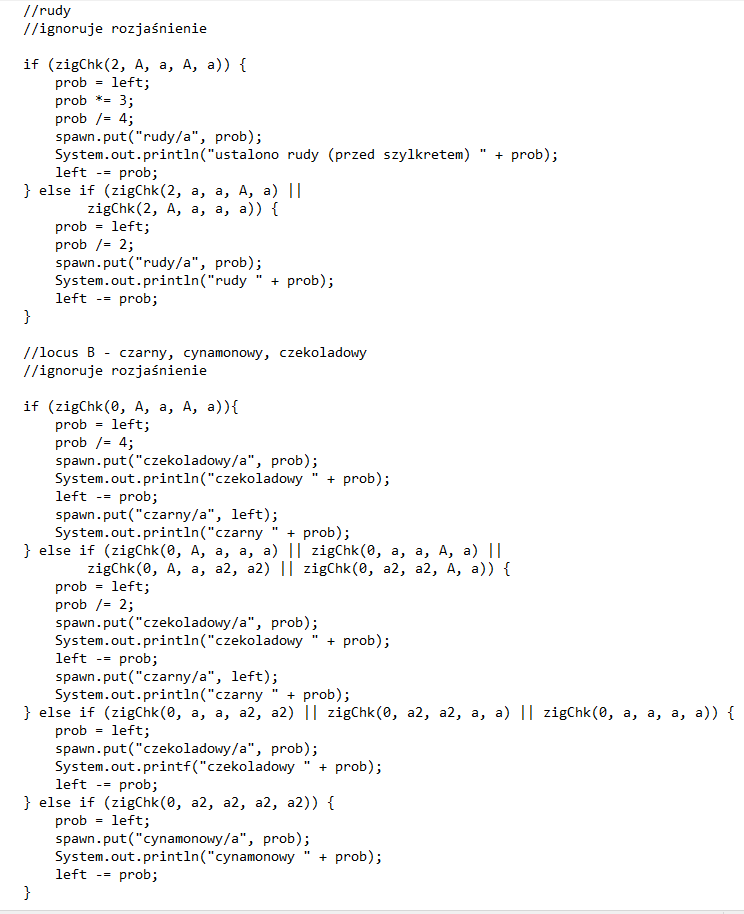
Niektóre geny posiadają więcej niż dwa allele, przykładem jest gen w locus B, który posiada trzy allele: B (dominujący nad b i b2) odpowiadający za kolor czarny, b (dominujący nad b2) kodujący kolor czekoladowy oraz b2 – cynamonowy. Locus C posiada aż 5 alleli, ale z powodu ich rzadkości, w tej aplikacji potraktowałem je wszystkie jeko jeden allel recesywny.

Map<String, Integer> spawn – mniejsza mapa, po wypelnieniu dodawana do większej listy



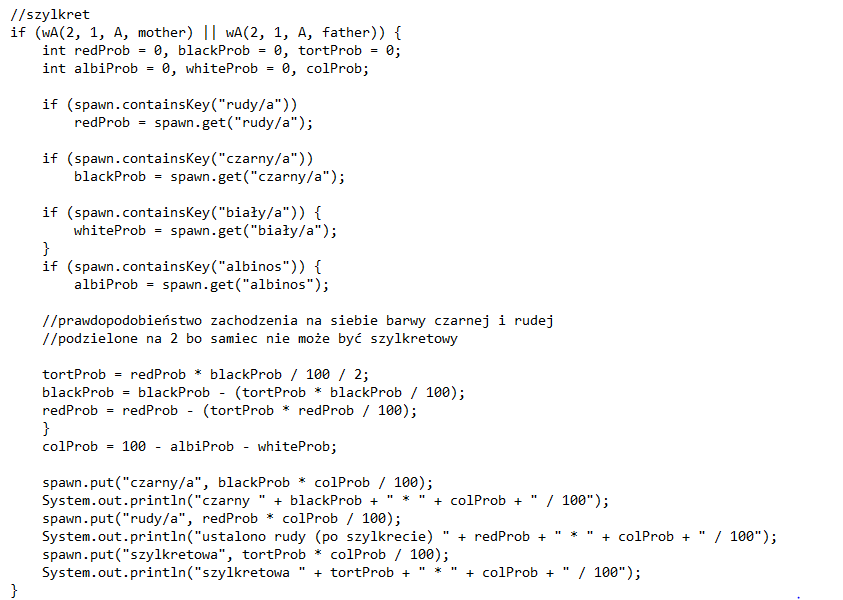
Rys. 11

Metoda po kolei sprawdza geny zaczynając od najbardziej wykluczających miejsc jakimi są biały i albinos, te cztery ify reagują jeśli w miocie jest biały kot lub albinos.



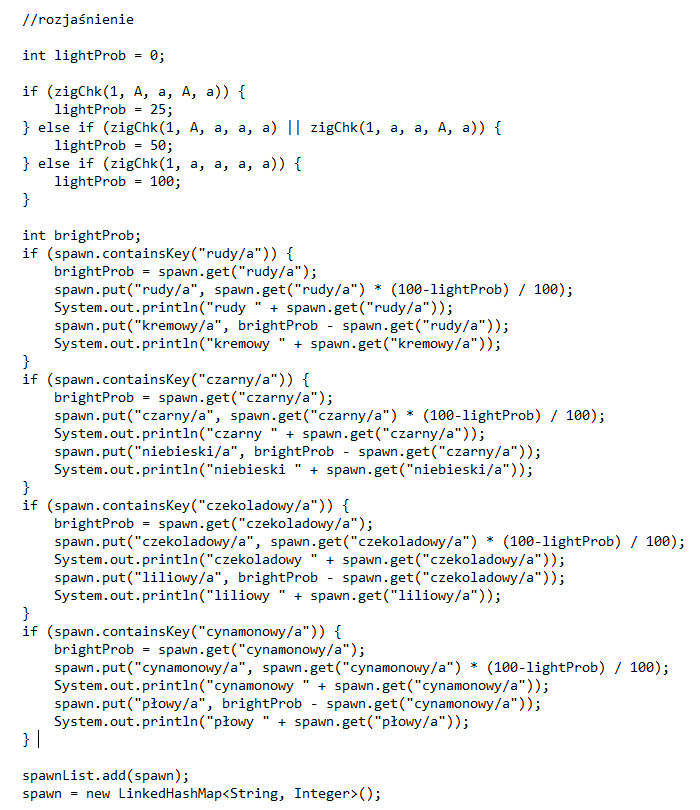
Rys. 12

Metoda sprawdza wszystkie kombinacje alleli w locusach O i B i ustawia stosowne prawdopodobieństwa. Jak widać po kodzie, gen w locus B ma trzy allele, jeden dominujący nad drugim, gdzie b jest dominujące względem b2.



Rys. 13

W tym miejscu metoda liczy prawdopodobieństwo szylkretki które wynosi redProb \* blackProb / 100 / 2



Rys. 14

Pod koniec ustalania wybarwienia włosa, odświażane są wartości barw, które można rozjaśnić, tj. (jak na obrazku) rudy do kremowego, czarny do niebieskiego itd.



Pod koniec każdej sekcji (jakimi są: wybarwienie, pręgowanie, długość włosa, łaty) mapa dodawana jest do listy, a następnie czyszczona.

Sekcje są od siebie oddzielone, bo poszczególne cechy z nich nie mają wpływu na te z innych, poza wyjątkami jakimi są biały kot i albinos, takie nie mają łat ani pręgowania.

**Bibliografia**

https://poznajkota.pl/podstawy-genetyki-kotow.html