**ЛАБОРАТОРИСКА ВЕЖБА 3**

30.12.2020

**ПРОГРАМИРАЊЕ НА**

**ВИДЕО ИГРИ И СПЕЦИЈАЛНИ ЕФЕКТИ**

*Simulate puzzle game*

Професор*:* д-р Катарина Тројачанец Динева

Студент*:* Кирил Зеленковски

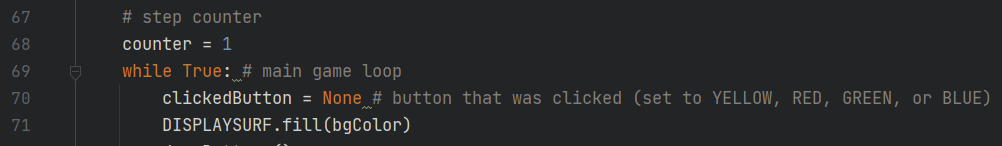
1. ***Барање 1***

*Секвенцата што играчот треба да ја повтори потребно е да биде комплетно различна од таа во претходната итерација на играта така што секој елемент од секвенцата ќе биде додаден со случаен избор од можните полиња.*

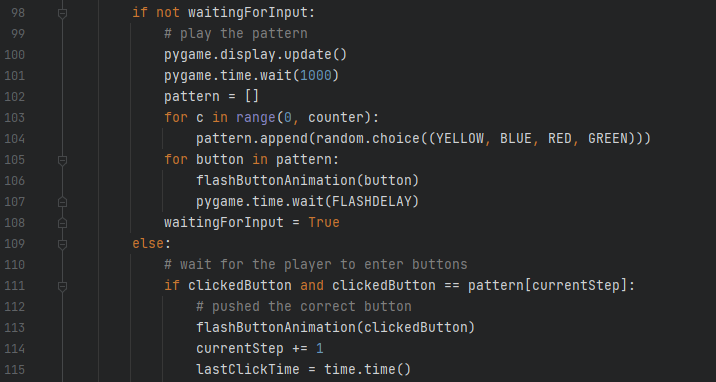
Ова барање се врти околу манипулација на главната листа од чекори. По default таа секој пат кога играчот ќе погоди чекор ја зголемува листата со тоа што рандом прикачува елемент од можните бои (коцки со боја кои се црвено, зелено, жолто, плаво). И потоа од 0 го прикажува тој pattern, елемент по елемент.

Јас прво дефинирав бројач, counter кој го поставувам иницијално на 1 и потоа секој пат кога играчот ќе погоди боја, следната итерација ја сетирам листата pattern = [] да биде празна и во јамка од ново додавам рандом елементи од 0 до counter (бројачот на погодени доаѓа). И секој пат кога ќе погодам го зголемувам бројачот и автоматски ја полнам листата со нови елементи.

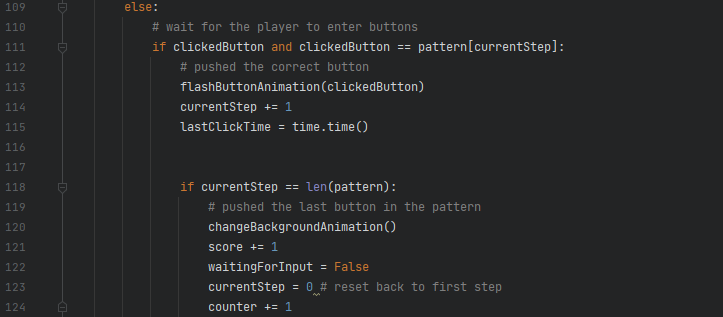
Додавање на променливата counter = 1, пред јамката за главната игра.



Потоа кога сме погодиле чекаме да ни се зголеми резултатот и да се промени секвенцата. Тука ја поставуваме листата да биде празна, потоа ја двжиме јамката од 0 до бројачот и се доделуваме вредности рандом за таа секвенца (ред 102-104).



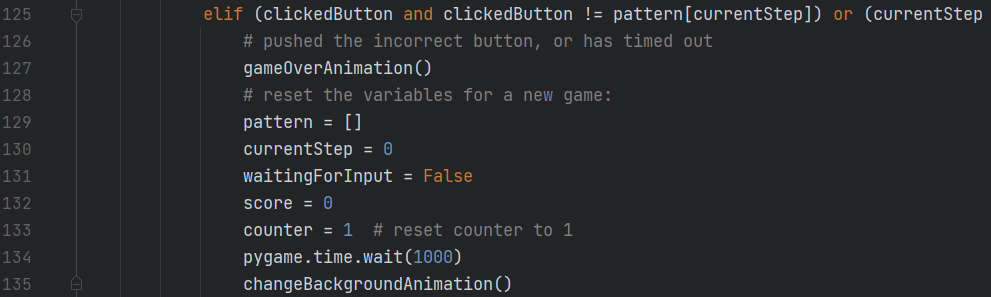
Потоа кога ќе се стигне до истиот број на чекори при повторување од играчот се зголемува бројачот за листата (ред 124).

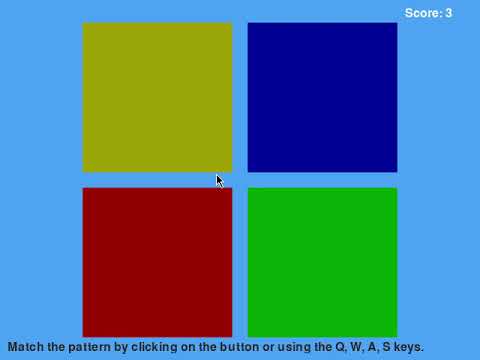


Резултатот од една итерација (видео линк подолу) на мојот компјутер од ваквиот код беше следна:

**1 итерација:** црвена   
**2 итерација:** зелена, црвена  
**3 итерација:** зелена, жолта, жолта   
...

Исто така доста важно, откако ќе згрешиме и ќе имаме **gameOver** треба исто така да се ресетира бројачот на 1 повторно за да не продолжи нова игра од пример нивото или итерацијата до која сме стигнале.



[](https://www.youtube.com/embed/T98T4Va1STw?feature=oembed)Бидејќи сметав дека подобро ќе биде визуелно да се види промената место screenshots снимив видео и е достапно тука:

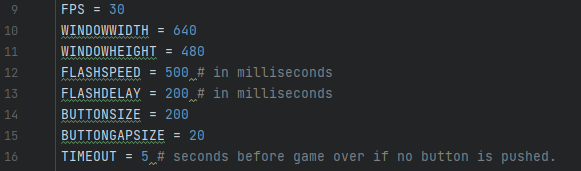
<https://youtu.be/T98T4Va1STw>

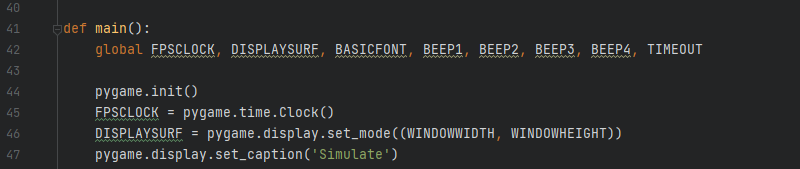
1. ***Барање 2***

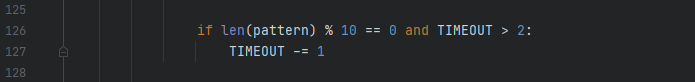
*Намалете ја вредноста на timeout (почнувајќи од 5 дo 3) на секои 10 промени на секвенцата. По достигнување на вредност 3, повеќе не се прави намалување.*

Најпрво ја поставуваме променливата TIMEOUT на 5 во самиот почеток од кодот и ја додаваме како глобална променлива за да избегнеме конфилкт за доделување на вредност пред да се дефинира бидејќи ако сакаме да и смениме вредност а не е глобална ќе имаме грешка (линија 16 и линија 42).

Главна јамка:



Дефинирање како глобална:

Потоа правиме промена во главната јамка каде идејата е следна ако должината на чекорите (секвенцата) која играчот треба да ја повтори е делива со 10 (тоа е случај на 10, 20, 30 .. должина) и TIMEOUT е поголем од 2 (тоа е сучај за 5, 4, 3) тогаш, намали го TIMEOUT. Со тоа намалуваме на 10 чекори од 4 на 5, на 20 од 4 на 3 и потоа секогаш е 3.

Ова се случува во самата јамка и после промената на 3 нема да влезе во овој услов и вредноста за TIMEOUT останува 3.

1. ***Барање 3***

*Да се зголеми матрицата од полиња за едно по двете димензии (ширина и висина) секогаш кога ќе се достигне резултат со вредност што цел број пати содржи 10 (10, 20, 30, ...).*

За ова барање проверката е слична како претходното барање, кога резултатот % 10 ќе биде 0 тогаш ќе додаваме едно поле во ширина и едно во висина.

Graphical user interface, application

Description automatically generatedБидејќи повеќе бои ќе беа неопходни барањето го правев за 0-10 резултат, 11-20, 21-30 и потем 30 останува истата матрица. Визуелно ги прикажувам додавањата на полињата со следниот цртеж (правен во draw.io)

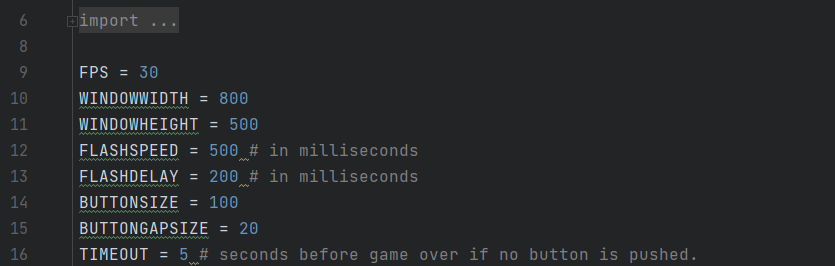
Прво рештката ја намалувам на 2 полиња и потоа зголемувам по 2 полиња (едно висина едно ширина). Причина зошто намалувам рештката е бидејќи функционалноста е иста независно дали започнувам од 1х2 или од 2х2, само повеќе бои ќе треба доколку полињата се од 2х2 (3то ниво вака ќе биде 2х4).

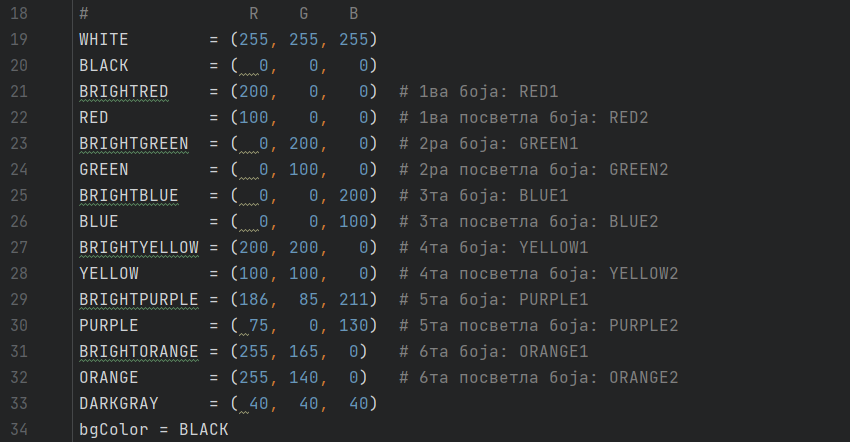
Чекорите се слични како во работата на час, променливи кој ги менувам се следни:

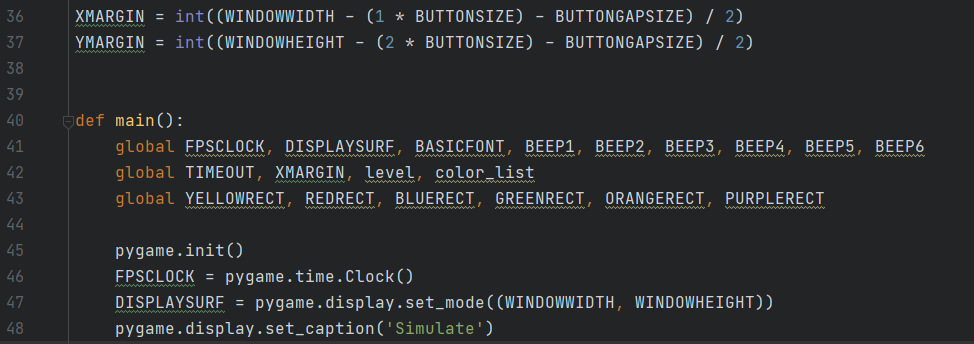
* **XMARGIN:** оваа променлива е од големо значење треба да ја менуваме за да може да собира 1 колона, 2 колони, 3 колони, ..
* **color\_list:** ова е листа од торки, каде секоја торка е боја.
  + 1 ниво: Иницијално е поставена на 2 бои – YELLOW, RED
  + 2 ниво: Потоа се проширува оваа листа за уште 2 бои – YELLOW, RED, BLUE, GREEN
  + 3 ниво: И на крај се сите 6 – YELLOW, RED, BLUE, GREEN, ORANGE, PURPLE
* **level:** ова е обичен цел број кој го чувам за подоцнежни проверки во самите функции го подавам како аргумент и исто така го користам за испиушување на нивото на самиот екран слично како поените собрани

NOTE: За ова барање при тестирање морав да ја сменам промената од барање 1 за рандом иницирање на секвенцата бидејќи тешко станување за играње. И исто така при тестирање користам помали резултати за нивоата (1 ниво: 0-3, 2 ниво: 4-6, 3 ниво: 7-..). Во финалниот код прикачен ќе ги модифицирам на 10, 20, 30.

Иницијални чекори:

Менување на иницијалните големи на прозорецот (800 на 500):

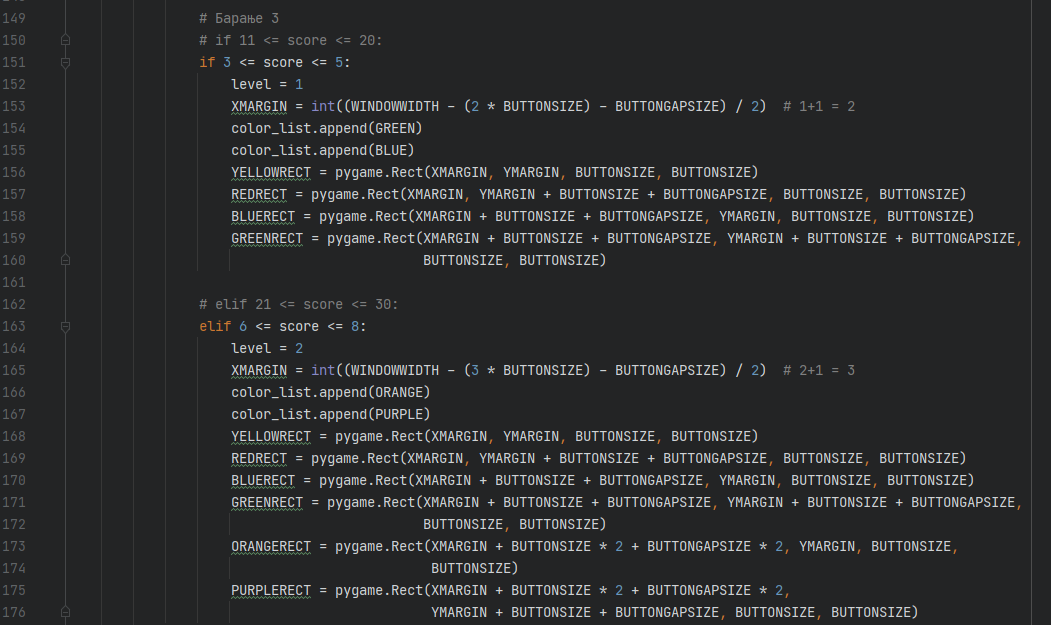
Додавање бои (обични и светли за трепкање) за сите копчиња:

Дефинирање на почетна маргина (1 на 2 матрица почетна) како и дефинирање на повеќето од променливите како глобални за да овозможиме подоцнежен пристап:

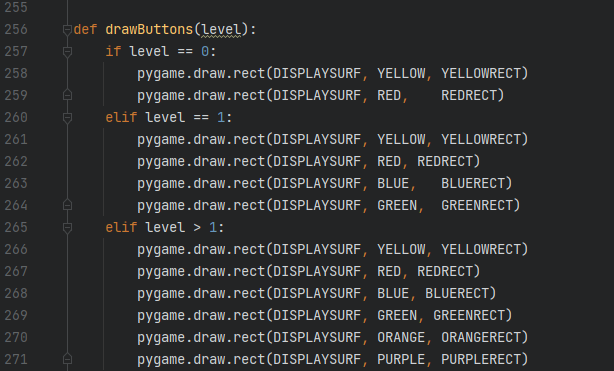
Дефинирање на правоаголник со текст за score, и за level, додавање на сите звуци:

До сега сите дефиниции се фокусирани на имплементација на самите копчиња. Сега веќе започнуваме со самите модификации на нивоа. Прво проверуваме дали сме на ниво потоа ако идеме во ново ниво:

* Дефинираме ново ниво
* Пресметуваме маргина нова
* Додавање на новите бои во листата (color\_list)
* Редефинирање на копчињата (ова го правам за поубаво да ги поместува при промена на маргина во самиот почетен код тие се дефинираат надвор но вака ќе бидат статични јас сакав да ги направам подинамични со тоа што ќе се менуваат на секое ниво)

Еве ги промените во самиот код:

Потоа останува да се додат на одредени места исцртувањата на копчињата а тоа е ф-јата draw\_buttons(), која ја модифицирав така што прима аргумент level.



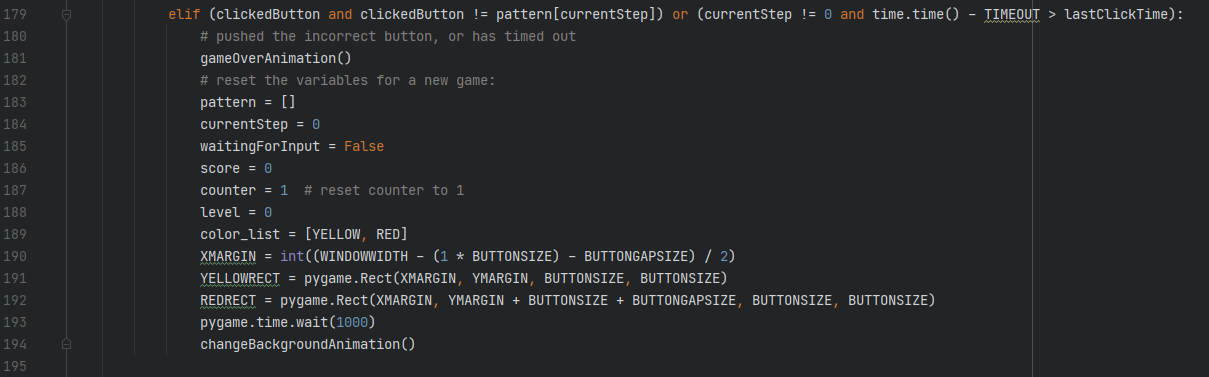
**1 ниво:** жолта, црвена

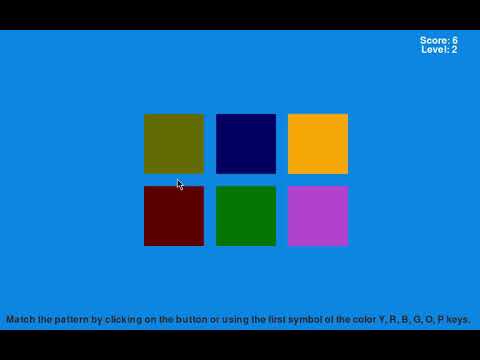
**2 ниво:** жолта, црвена, плава, зелена

**3 ниво:** сите 6

Потоа на одредени места само остана да се додадат опциите за притиснување на буква од тастатура за копче и некаде ако требаше да се додадат копчињата.

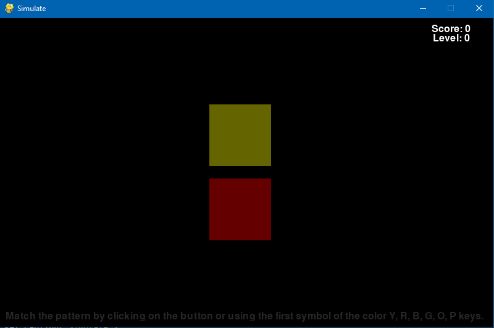
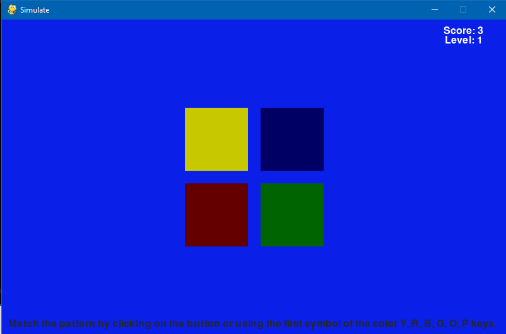
Притоа играта кога завршува треба да се ресетираат сотојбите на почетни:

* Маргината на првобитна состојба
* Нивото (level) на 0
* Листата од бои само на првите две бои
* Секвенцата на празна

[](https://www.youtube.com/embed/TGrtMdM-csA?feature=oembed)За поубаво да се види имплементацијата на ова барање снимив видео кое е достапно на следниот линк:

<https://youtu.be/TGrtMdM-csA>

Притоа ова се неколку screenshots од играта низ првите 3 нивоа (со 6 место со 10 бидејќи полесно беше за тестирање). Промените за %10 ги имам наведено во кодот кој го прикачувам може и со %10 и со фиксни вредности и двете работеа.



3 Level

2 Level

1 Level