KATOLÍCKE GYMNÁZIUM ŠTEFANA MOYSESA BANSKÁ BYSTRICA

Simulátor kociek pre hru Dungeons and Dragons

Ročníková práca

Patrik Valentíny

2021

KATOLÍCKE GYMNÁZIUM ŠTEFANA MOYSESA BANSKÁ BYSTRICA

Simulátor kociek pre hru Dungeons and Dragons

Ročníková práca

Patrik Valentíny

Banská Bystrica

2021

Konzultant: Jozef Sklenka

Úvod

V predkladanej ročníkovej práci demonštrujem tvorbu a využitie užívateľsky prívetivého programu, ktorý bude simulovať hod kockou formou náhodného generovania výslednej hodnoty.

V tejto práci sa zameriam na stolnú RPG hru Dungeons and Dragons, pretože pri jej hraní je potrebných až 6 typov kociek, z ktorých mnohé sú potrebné viackrát. Takýto set teda môže obsahovať až 11 kociek, čo môže byť pri začiatkoch pre niekoho finančne náročné. Takisto je potrebné tieto kocky niekde odkladať, môžu sa stratiť alebo poprípade poškodiť. Preto je jednoduchšie mať tieto kocky v jednom elektronickom simulátore, ktorý je nenáročný na priestor, financie, nedokážeme ho poškodiť alebo stratiť a máme ho stále so sebou.

Výsledný program obsahuje samotný algoritmus na hod kockou, spôsob výberu rôznych kociek, ktoré sa v Dungeons and Dragons nachádzajú a možnosť hodu viacerými kockami zároveň. Doplnkovou funkciou programu je zobrazenie histórie našich hodov. Všetky tieto časti programu sú zakomponované do spustiteľného užívateľského prostredia, ktoré je pre užívateľa intuitívne a zároveň má príjemný a prehľadný vzhľad.

Pre spustenie programu potrebujeme mať vo svojom počítači nainštalovaný Python verziu 3 a k nemu stiahnutelný modul kivy vo verzii 2.0.0.

Moja práca je rozdelená do dvoch kapitol. V prvej kapitole je popísaná stolová hra Dungeons and Dragons, jej pravidlá, priebeh hry a základný popis kociek, ktoré sa v nej používajú. V druhej kapitole sa nachádza popis postupu vytvorenia simulácie na hod kockou v jazyku Python. V jej podkapitolách sa nachádza krátka história jazyka Python, popis vytvárania algoritmov potrebných pre túto simuláciu a vytvorenia užívateľského prostredia.

1 Čo je Dungeons and Dragons

Dungeons and Dragons alebo skrátene len D&D je fantasy stolová hra, v ktorej sa hráči vžívajú do roly jednej z jej hrdinov. Jej návrhárom je Gary Gygax a Dave Arneson.[1] Jej prvá verzia bola publikovaná v roku 1974 spoločnosťou Tactical Studies Rules, Inc. Dnes existuje už jej piata edícia, ktorú, ako aj všetky ostatné, od roku 1997 publikovalo vydavateľstvo Wizards of the Coast, ktoré je súčasťou Hasbro.[2]

1.1 Priebeh hry

1.1.1 Dungeon Master

Na začiatku hry sa určí Dungeon Master - Pán Jaskyne (ďalej len DM). Ten určuje príbeh hry, je jej rozhodcom a v neposlednom rade pripravuje prekážky pre hrdinov. Je to najdôležitejšia úloha v celej hre. On musí určiť aké možnosti výberu hráči dostanú, musí vymyslieť zaujímavé lokácie, prekážky v nich, nepriateľov a ich schopnosti.

1.1.2 Hrdinovia

Hrdinov si vytvárajú hráči samostatne pričom si vyberajú z niekoľkých rás (človek, trpaslík, elf, hobit), a povolaní. Následne svojim hrdinom na základe hodu kockou určia hodnotu vlastností, ktoré sú sila, obratnosť, odolnosť, inteligencia, múdrosť a charizma. Jednotlivé povolania potrebujú rozdielne vlastnosti, napríklad pre bojovníka je dôležitá sila, no pre kúzelníka je dôležitá inteligencia. Takisto môžu mať špeciálne vlastnosti, ako napríklad schopnosť vidieť v tme alebo rozumieť cudziemu jazyku.

---

[1] Darren Waters. 2004. What happened to Dungeons and Dragons?. [online] 26 Apríl 2004. [cit.2020-12-10]. Dostupné na internete: <http://news.bbc.co.uk/2/hi/uk_news/magazine/3655627.stm>

[2] Cecilia D'Anastasio. 2019. Dungeons & Deceptions: The First D&D Players Push Back On The Legend Of Gary Gygax. [online]. 26 August 2019. [cit.2020-12-10]. Dostupné na internete: <https://kotaku.com/dungeons-deceptions-the-first-d-d-players-push-back-1837516834>

1.1.2 Dobrodružsvá

Dobrodružstvá sa v D&D môžu byť veľmi rôznorodé. Jedno môže prebiehať v podobe preskúmavania jaskyne a ďalšie musí vyriešiť vzťahy v kráľovstve. Napriek tomu majú všetky podobný priebeh. Na začiatku popíše DM prostredie, v ktorom sa hrdinovia nachádzajú. Následne hráči opíšu, ako chcú v lokácií postupovať. Tu sa využíva pohyb, interakcia s prostredím a dialóg. Nakoniec DM vylíči následky ich konania. Tie ovplyvňujú postup hrdinu a takisto môžu viesť k stretu s inými postavami, ktoré môžu byť priateľské alebo nepriateľské, tiež označované ako príšery. Častým výsledkom postupu je aj boj s danými príšerami. Pri ňom sa využíva z veľkej časti kocka. Podľa týchto troch bodov postupne pokračuje celá hra.[3]

1.2 Element náhody - Kocka

Využíva sa pri náročnejších úkonoch alebo pri boji. Existuje 7 rôznych tvarov kocky. Zároveň sa niektoré kocky používajú viackrát, takže na jednu hru môže byť potrebných až 11 kociek.

1.2.1 Kocka D20

20 hranná kocka (viď obr.1), ktorá je jedným zo znakov D&D. Využíva sa dvadsaťsten, ktorého každá strana má tvar rovnostranného trojuholníka (viď obr.1) s číslom od 1 do 20. Každá hodnota má šancu 5%.

Táto kocka sa používa najmä pri rozhodovaní, či bola činnosť hrdinu úspešná alebo nie. Pri útočení kocka D20 rozhodne o tom, či bol útok úspešný. Následne hodnotu poškodenia určujú menšie kocky.

Obrázok 1 Kocka d20

[3] Wizards RPG Team. 2014. Player's Handbook. Piata edícia. Renton, WA : Wizards of the Coast LLC. 9780786965601.

1.2.2 Kocka D10

10 hranná kocka (viď obr.2) určuje percentuálny podiel. Používajú sa dve kocky zároveň, pričom na jednej sa nachádzajú čísla od 0 do 9 a na druhej čísla od 0 do 90 v násobkoch po 10. Po hode sa obidve čísla sčítajú s výnimkou dvoch núl – vtedy je hod rovný 100.

Obrázok 2 Kocka d10

1.2.3 Kocka D6

Kocka D6 (viď obr. 3) je všeobecne používaná šesťhranná kocka a používa sa na začiatku hry pri vytváraní hrdinov. V jednom sete kociek sú často až štyri takéto kocky pre urýchlenie kôl.

Obrázok 3 Kocka d6

1.2.4 Kocky D12 a D8

Kocky, ktoré sú používané na vyhodnotenie poškodenia vykonaného väčšími zbraňami.
D12 (viď obr. 4) je tvorená dvanástimi päťuholníkmi s hodnotami od 1 do 12.

Obrázok 4 Kocka d12

D8 (viď obr. 5) vyzerá ako dve spojené pyramídy ktoré majú na každej strane 4 rovnostranné trojuholníky s číslami od 1 do 8.

Obrázok 5 Kocka d8

1.2.5 Kocka D4

D4 (viď obr. 6) je najmenšia zo všetkých. Má tvar trojstrannej pyramídy / tetraédra a na rozdiel od tradičnej kocky, kde sa na každej strane nachádza jedno číslo, tu sa nachádzajú až tri – jedno pri každom vrchole steny. Hodnotu hodu určuje číslo pri hornom vrchole. Táto kocka sa používa pri vyhodnocovaní poškodenia malými zbraňami.[4]

Obrázok 6 Kocka d4

[4] Die Hard Dice Store. 2020. What are DnD Dice?. [online]. [cit.2020-12-18]. Dostupné na internete: <https://www.dieharddice.com/pages/d-d-dice-explained>

2 Programovanie kocky

Keďže na jednu hru je potrebných až 11 kociek, je jednoduchšie využívať elektronický generátor, ktorý hru zrýchľuje a zároveň odstraňuje časť vecí, ktoré sú potrebné na začatie s DnD.

2.1 Jazyk Python

Python je programovací jazyk, ktorý môže byť použitý vo veľkom množstve prostredí. Jeho vývoj bol orientovaný v takom smere, aby bol jednoducho čitateľný vo všetkých využitiach, či už ide o malé, alebo veľké projekty.[5]
Táto filozofia je sumarizovaná v dokumente The Zen of Python v niekoľkých aforizmoch ako napríklad “Krásne je lepšie ako škaredé” alebo “Jednoduché je lepšie ako zložité”.[6]
Jeho prvá verzia bola vydaná v roku 1991 programátorom Guidom van Rossumom. Vývoj Pythonu 2.0, ktorý bol vydaný v roku 2000, bol v roku 2020 zrušený. Dnes sa už používa Python 3.0 a jeho najnovšia stabilná verzia je Python 3.9.1.
V Decembri 2020 bol tretím najpopulárnejším programovacím jazykom hneď po C a Jave.
Je využívaný prakticky v každom odvetví a takisto aj vo vedeckých výskumoch. Časté využitia sú automatizácia funkcií v iných softvérových aplikáciách. Využívajú ho aj veľké organizácie ako Google, NASA, IBM, CIA.[7]

---

5 Python Software Foundation. 2020. Organizations Using Python. [online]. [cit.2020-12-18] Dostupné na internete: <https://wiki.python.org/moin/OrganizationsUsingPython>

6 PETERS, Tim (19 August 2004). PEP 20 – The Zen of Python. [online]. 19 August 2004 [cit.2020-12-18]. Dostupné na internete: <https://www.python.org/dev/peps/pep-0020/>

7 Python Software Foundation. 2020. Quotes about Python. [online]. [cit.2020-12-18] Dostupné na internete: <https://www.python.org/about/quotes/>

2.2 Základný algoritmus kocky

2.2.1 Čo je algoritmus, funkcia, premenná a moduly v Pythone

Algoritmus je “presný a logicky jednoznačný predpis na vykonanie určitej sústavy operácií, pričom je určené aj poradie na riešenie úloh daného typu, t.j. úloh, ktoré majú matematický model”[8]

Funkcia v Pythone je skupina údajov, ktoré vykonajú určitú úlohu po jej zavolaní. Funkcie nám pomáhajú rozdeliť náš program do menších častí, ktoré umožňujú modulárnosť a znovu-použiteľnosť. Takisto obmedzujú zbytočné opakovanie rovnakých častí kódu. V Pythone je definovaná pomocou “def názov funkcie (premenné)”.

Premenná je symbol, ktorý reprezentuje konkrétnu hodnotu v jednom čase. Základné typy premenných sú “string”, ktorá sa používa pre variabilné zapisované ako slová. Tieto premenné sa často používajú napríklad pri zadávaní mien, ktoré budú použité v programe neskôr. Následne “integer” a “float” označujú čísla, ktoré môžeme neskôr použiť v matematických operáciách. Integer sa používa pre celé čísla a float pre racionálne čísla. Poslednou z najčastejšie používaných je “boolean”. Boolean určuje, či je premenná pravdivá, alebo nepravdivá”

Modul v Pythone je súbor, ktorý obsahuje funkcie, triedy a variabilné, ktoré môžeme po importovaní používať ako časť nášho programu. Na importovanie sa využíva funkcia import. Môže byť napísaná buď ako import “názov modulu”, toto umožní používanie celého modulu. Alternatívne môžeme použiť from “názov modulu” import “názov funkcie/triedy/variabilnej” – táto alternatíva vloží do nášho programu len časti, ktoré potrebujeme, čo ušetrí využitie zdrojov, ktoré nám počítač ponúka.

8 IVANOVÁ-ŠALINGOVÁ, Mária - MANÍKOVÁ, Zuzana. 1979. Slovník cudzích slov A/Z. Prvé vydanie. Bratislava: Slovenské pedagogické nakladateľstvo. strana 53. ISBN 67-567-79.

2.2.2 Problémy kocky v našom prípade

Náš základný algoritmus bude mať za úlohu vybrať jedno číslo v rozmedzí od čísla 1 až po maximálnu hodnotu kocky. V tomto prípade nastáva problém, keďže vytvárame program pre viaceré maximálne hodnoty. Máme na výber dve možnosti. Jednou z nich je vytvorenie priamo danej funkcie pre každú hodnotu kocky. Pri tejto možnosti nastáva veľká repetitívnosť. Každá funkcia bude mať väčšinu údajov rovnakú a bude sa meniť len jedno číslo, ktoré udáva veľkosť kocky. Preto je výhodnejšie použiť v našej funkcii premennú, ktorá udáva veľkosť kocky a následne ju definovať pri volaní funkcie. Týmto sa v našom prípade stane zo šiestich funkcií jedna, v ktorej sa nachádza premenná, čo zlepší prehľadnosť kódu a jeho modularitu.

2.2.3 Funkcia kocky v Pythone

Základnú funkciu môžeme vidieť v prílohe A.1. Ako každá funkcia, začína sa slovom “def”, následne pokračuje názov funkcie, v našom prípade je to “kocka” a na konci sa v zátvorkách nachádza premenná, ktorú môžeme udávať pri volaní funkcie. V našom prípade je jej názov “numMax” a je typ integer, čiže celé číslo. Vo vnútri funkcie sa na začiatku nachádza ďalšia funkcia print(). Táto funkcia má za úlohu vpísať do konzoly to, čo jej určíme. V prvom prípade vypíše písmeno “D” a za ním, po prekonvertovaní variabilnej z integeru na string, k nemu pridá hodnotu “numMax”. Ďalší bod funkcie je určenie hodnoty hodu kockou. Túto hodnotu zapíšeme do premennej throw a jej hodnotu určíme pomocou funkcie randint() z modulu random, ktorý sme importovali na začiatku nášho kódu. Randint() určí náhodné číslo v rozmedzí, ktoré do funkcie zadáme. V našom prípade to bude číslo jedna a premenná numMax. Posledná časť z tejto funkcie bude zobrazenie hodnoty hodu. Na tento úkon znovu použijeme funkciu print a zadáme do nej premennú throw. V tomto prípade už nemusíme integer premieňať na string, pretože ju nespájame s iným typom vypisovanej hodnoty.

2.3 Grafické používateľské rozhranie - GUI

V tomto momente je náš program veľmi jednoduchý a všetko čo dokáže je vypísať 6 hodnôt v nami zadanom rozmedzí. Toto program urobí do takzvaného terminálu, čo nie je pre bežného užívateľa praktické a ani jednoducho zrozumiteľné. Kvôli tomuto musíme v programe vytvoriť rozhranie, v ktorom sa bude vedieť používateľ jednoducho vyznať a porozumie mu. Takisto sem implementujeme možnosť výberu hodnoty kocky a opakovania hodov kockou bez potreby znovu spustiť program. Tento program bude prvá, užívateľsky použiteľná forma nášho programu.

V pythone je možné si vybrať z viacerých prostredí v ktorých je možné používateľské rozhranie naprogramovať. Keďže v tomto bode je pre nás dôležitejšia funkčnosť programu ako jeho výzor, budeme používať najjednoduchší a de facto základný modul - tkinter.
Aby sme s ním mohli pracovať, musíme ho do programu importovať. V tomto prípade jeho názov pre zrýchlenie práce premenujeme z “tkinter” na “tk”.

Na začiatku vytvoríme okno, v ktorom bude náš program pracovať. Poslúžia nám na to funkcie tkinter.Canvas, v ktorej uvedieme šírku, výšku a farbu pozadia, a druhá funkcia pack(), ktorá umožní vložiť “Canvas” do okna.

Ďalšia časť je vytvorenie tlačidla pre hod kockou. Na to nepoužijeme skutočné tlačidlo, ale len nakreslený štvorec v ktorom sa bude nachádzať text “Roll D” a na konci bude číslo kocky, ktoré sa bude podľa výberu meniť. Pre spustenie funkcie hodu bude potom potrebná ďalšia funkcia, ktorá je opísaná neskôr.
Štvorec aj text budú “kreslené” funkciami create_rectangle a create_text. V oboch bude definovaná ich pozícia a farba. Pri štvorci bude pridaná šírka okraja a farba na akú sa má meniť pri prechode myšou naň. Pri texte to bude samotný text, jeho font a veľkosť.

Pri tlačidlách výberu kocky je princíp rovnaký. Vytvoríme štvorec a doň vložíme text. Text v ňom bude pre jednoduchosť iba “D” a číslo kocky. Keďže je ale

týchto tlačidiel šesť bolo by neefektívne opakovať rovnaké príkazy viackrát, preto ich vložíme do jednej triedy. Trieda je podobná funkcii, ale premenné do nej zadané môžu byť použité aj mimo ňu. Toto bude potrebné pre určenie polohy tlačidla a priradenie funkcie k nemu. Náš program v tomto momente môžete vidieť na obrázku 7.

Obrázok 7 Prvotné GUI

2.3.1 Funkcia pre výber kocky

Jej úlohou bude zmeniť maximálnu hodnotu, ktorú môže náš hod dosiahnuť. Bude vyzerať nasledovne: názov funkcie bude len jednoduché “d” pretože kocky v D&D sú označované podľa ich veľkosti napr. d20, d12 atď. a premenná, ktorú bude možné zadávať pri jej volaní bude veľkosť kocky. V jej vnútri bude zapísanie premennej veľkosti kocky. Vymazanie textu, ktorý sa je vpísaný do tlačidla na hod kockou (pri spustení programu je to Roll D20) a znovu napísanie tohto textu s hodnotou, ktorá závisí od daného tlačidla.

2.3.2 Funkcia pre hod kockou

Funkcia pre opakovanie hodov bude podobná ako v prvej verzií nášho programu, ale maximálnu hodnotu hodu bude určovať premenná, ktorá bude daná tlačidlom pre výber kocky. Preto sa pri randint zmení premenná “numMax” na nami novo zadanú “dice_option” určovanú tlačidlom na výber kocky. V závere funkcie bude, podobne ako v predošlej, vymazanie textu, ktorý je v tomto prípade vypísaná hodnota hodu kocky, alebo pri spustení je to len prázdna premenná a následné zobrazenie novej hodnoty hodu.

2.3.3 Prvé spustenie programu

Nakoniec, keďže sme nevytvárali skutočné tlačidlá, ale len geometrické útvary a text, musíme tieto útvary premeniť na fungujúce tlačidlá. Najjednoduchší spôsob bude vytvorenie funkcie, ktorú priradíme k stlačeniu ľavého tlačidla myši. Po jeho stlačení bude zavolaná funkcia a do jej premennej budú zapísané súradnice, v ktorých bolo tlačidlo stlačené. My túto skutočnosť využijeme

pomocou logickej operácie if, ktorá určuje, či je daná podmienka pravdivá alebo nepravdivá a následne vykoná nami zadaný príkaz. V jej prvom riadku definujeme, že ak sú súradnice stlačenia v rozmedzí veľkosti štvorca, ktorý využívame ako tlačidlo pre hod kockou, aby zavolala funkciu pre hod kockou. Následne zopakujeme rovnaký proces pre všetky tlačidlá, ktoré menia hodnotu kocky. Pri nich nebudeme zadávať súradnice ako číslo, ale ako premennú, ktorú sme zadali pri definovaní štvorca v každej triede. Tento postup nám umožní meniť veľkosti štvorcov bez toho, aby sme museli meniť čísla na dvoch miestach. Tak isto to predíde chybám v programe, v prípade, že by sme zmenili len jednu veľkosť a na druhú by sme zabudli.

V tomto momente je náš program funkčný z technického hladiska, no má mnohé chyby vo vizuálnych stránkach. Tlačidlá sa prefarbia na inú farbu, keď sa nachádzame nad nimi, ale zmenia farbu na pôvodnú, akonáhle sme nad ich textom. Taktiež nie je užívateľsky jednoducho zrozumiteľný a nie každý by ho dokázal používať.
Ďalej sa v ňom dajú doplniť mnohé funkcie ako je napríklad hodenie viacerých kociek naraz, a k nemu výpis čiastkových výsledkov a zároveň aj celkového.

Tieto problémy a funkcie by sa síce v module Tkinter dali vyriešiť, no efektívnejším spôsobom bude použitie modulu, ktorý je priamo určený na vytváranie užívateľských prostredí. Sú na výber mnohé možnosti, každá z nich má isté výhody a nevýhody. My budeme používať modul “Kivy”, pretože umožňuje použitie rovnakého kódu na viacerých typoch zariadení, a tým umožní relatívne jednoduchý prechod v prípade, že by sme chceli tento program používať na prenosných dotykových zariadeniach, ktoré používajú iný operačný systém.

2.4 Finálna verzia programu

V tejto časti, tak ako už bolo spomenuté, budeme na vytvorenie užívateľského prostredia používať modul Kivy a s ním spojené. Asi najväčší rozdiel, medzi Kivy a Tkinter je ten, že pri Kivy vytvárame používateľské prostredie vo

vlastnom jazyku Kv, ktorý sa píše v samostatnom súbore a v pythone sa na tento súbor len odkazuje. Toto nám v značnej miere sprehľadní obe časti kódu. V pythone sa budú nachádzať logické operácie a v Kv bude celé užívateľské prostredie. Kvôli tomuto sa bude prakticky celá naša časť python programu nachádzať v jednej triede.

Na začiatku budeme musieť spustiť náš program, zároveň odkázať na súbor s GUI a určiť vlastnosti okna v ktorom bude spustený. Toto bude vykonané pomocou dvoch príkazov z modulu Kivy. Prvý z nich bude “Builder.load_file()”, ktorý načíta súbor s GUI a definujeme pri ňom vlastnosti nášho okna. Druhým bude príkaz run() z modulu Kivy a triedy App, ktorý už dané okno s GUI vytvorí.

2.4.1 Hlavné okno

Základom GUI bude, tak ako v predchádzajúcom prípade, veľké tlačidlo, v ktorom sa bude nachádzať text Hoď, a za ním počet a druh kociek. Vykonávať bude rovnakú funkciu, kde sa podľa danej kocky vyberie náhodné číslo, pomocou funkcie randint(). Toto sa vykoná podľa zadania užívateľa určitý počet krát a jednotlivé výsledné hodnoty sa zapíšu spolu s ich súčtom do textového poľa pod daným tlačidlom. Toto môžeme vidieť v obrázku 8.

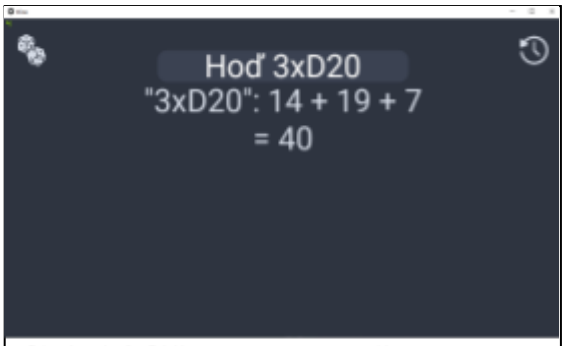
2.4.2 Menu pre výber kociek

Tlačidlá pre výber a počet kociek sa v tomto prípade nebudú nachádzať v rovnakom okne. Pre prehľadnosť ich vložíme do menšieho okna, ktoré bude vidieť len po stlačení tlačidla s ikonou kocky na ľavej strane obrazovky (viď obr.9). Aj keď to vyzerá ako nové okno, v skutočnosti sú to len dva obdĺžniky, jeden s rovnakou farbou, ako je pozadie hlavného okna a druhý s čiernou farbou no len osemdesiat percentnou viditeľnosťou, ktorá vytvorí efekt stmievania pozadia. Na menšom obdĺžniku bude 6 tlačidiel na výber druhu kocky a pod nimi priestor na určenie počtu kociek.

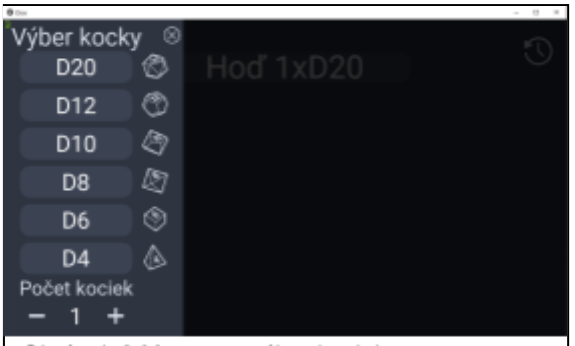
Funkcia pomocou ktorej vyberáme počet kociek nebude taká jednoduchá, ako sa môže zdať. Používateľ s ňou bude pracovať pomocou tlačidiel plus a mínus.

Medzi nimi bude textové pole, ktoré zobrazí vybraný počet kociek. Obe tlačidlá budú posúvať toto číslo o jedno, v im prislúchajúcom smere. Aby obe tlačidlá fungovali je potrebné vytvoriť funkciu, ktorá najprv prečíta aká hodnota je práve zapísaná a zmenší alebo zväčší danú hodnotu. Následne zmení zobrazované číslo pri tlačidlách a zároveň aj na tlačidle hodu. Ďalej je potrebné definovať v akom rozmedzí má táto funkcia fungovať. Nemôže sa tu nachádzať číslo menšie ako nula a zároveň pre veľkosť vypisovaného textu ani väčšie ako pätnásť. Na definovanie tohto rozmedzia budeme potrebovať funkciu if, ktorá bude určovať či terajší počet kociek je v prípade plus menší alebo rovný pätnásť a v prípade mínus, či je väčší ako nula.

Poslednou ich funkciou bude menenie farby po stlačení tlačidla a následné jej vrátenie po pustení. Tlačidlá v Kivy majú dve vlastnosti, ktorými je možné volať funkcie. Jedna sa aktivuje pri stlačení a druhá pri pustení. Pri týchto dvoch tlačidlách je pri stlačení spustená len zmena farby a zvyšné časti sú vykonané až po pustení.



Obrázok 8 GUI v novom prostredí



Obrázok 9 Menu pre výber kociek

Teraz už dokážeme určiť počet kociek, akým chceme hádzať. Zároveň sa tento počet zobrazuje na tlačidle, ktorým hádžeme. Zatiaľ ale stále nevypisujeme výsledky jednotlivých hodov, ale len súčet všetkých hodov. Ak chceme výsledok vypisovať vo forme “3xD20: 14 + 9 + 7 = 40”, budeme musieť vytvoriť funkciu, ktorá v prípade, že hádžeme viac ako jednou kockou, zapíše každý výsledok do premennej, ktorá bude mať formu “list”. Takáto premenná dokáže v sebe uchovávať viaceré výsledky nezávisle od toho, či sú to slová alebo čísla. Následne podľa dĺžky tohoto “listu” zapíšeme jednotlivé čísla do ďalšej premennej, ktorá bude vo formáte “string”, pričom za každým číslom až na posledné sa bude nachádzať plus. Toto dokážeme vďaka tomu, že každá

hodnota v “liste” vie byť zavolaná podľa jej pozície a zároveň vieme určiť počet hodnôt v danom “liste”. Preto použijeme ďalšiu funkciu “for”, zabudovanú v pythone. Táto funkcia vykoná niekoľkokrát, v tomto prípade podľa množstva hodnôt v danom liste, nami zadané funkcie. Pre zapísanie hodnôt to bude teda o jeden krát menej ako je dĺžka listu a funkcia v nej bude zapísanie každého až na posledný výsledok a plus vedľa neho do premennej, ktorá sa bude vypisovať pod hlavným tlačidlom. Pod ňou, už mimo funkcie for, bude zapísanie výsledku posledného hodu bez plus. Nakoniec túto premennú, so všetkými výsledkami pri stlačení tlačidla spolu s počtom kociek, jej typom a výsledkom zapíšeme do textového poľa pod tlačidlom. Kód potrebný pre vykonanie tohto úkonu môžete vidieť v obrázku 10.

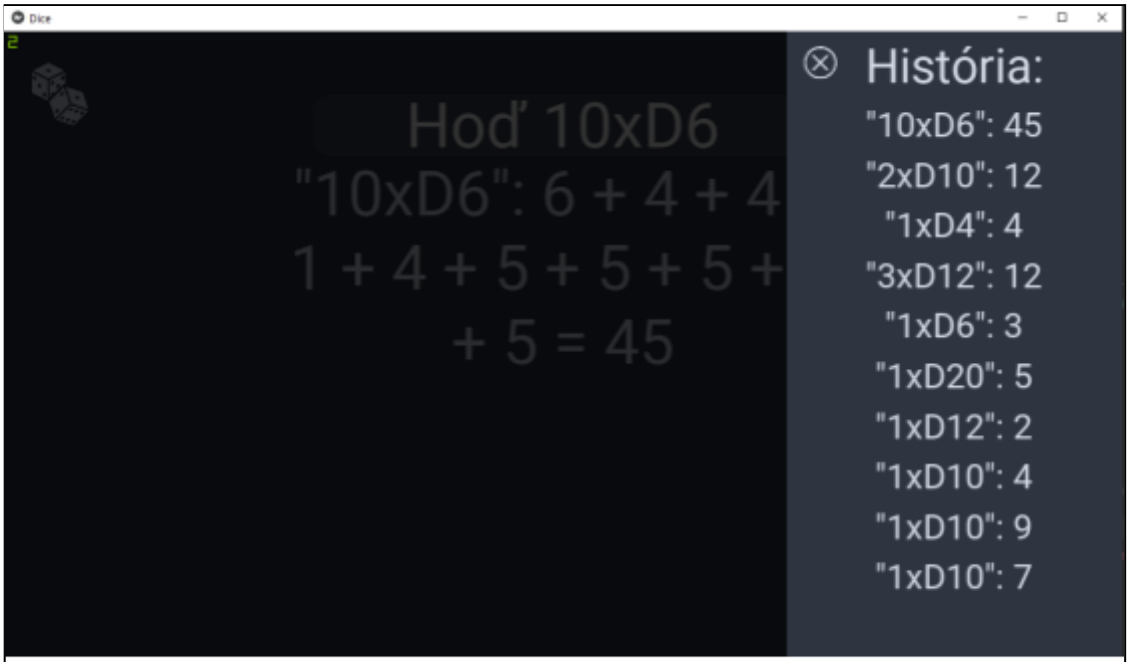
```
def rollValue(self):
    valuelist = []
    value = 0
    for i in range(0, self.diceCount()):
        valuelist.insert(0, randint(1, size))

    for j in range(0, len(valuelist)):
        value += valuelist[j]

    return valuelist, value

def rollButton_dwn(self):
    if self.diceCount() == None:
        self.mainText.text = 'Nezadaný počet kociek'

    elif 0 < float(self.diceCount()) <= 15:
        valuelist, value = self.rollValue()
        answer = ''
        if len(valuelist) == 1:
            answer += str(valuelist[0]) + ' '
        else:
            for i in range(0, len(valuelist) - 1):
                answer += str(valuelist[i]) + ' + '
            answer += str(valuelist[-1])

        self.mainText.text = '"' + str(self.diceCount()) + 'xD' + str(size) + '": ' + answer + ' = ' + str(value)
```

Obrázok 10 Kód pre zapisovanie výsledkov

2.4.3 História

Posledným prídavkom do nášho programu bude možnosť zobraziť históriu našich hodov. Toto sa bude diať v podobnom okne, v akom je aj výber kociek, no namiesto tlačidiel sa tu bude nachádzať desať textových polí a nápis história (viď obr.11). Vypisovanie do textových polí bude iniciované stlačením hlavného tlačidla a pozícia, v ktorej sa má daný text zobraziť, bude zasa využívať premennú list. Každým stlačením tlačidla sa do “listu” na prvú pozíciu pridá hodnota hodu, v tomto prípade kvôli veľkosti textu bez medzivýsledkov. Podľa

pozície v liste a očíslovania textových polí sa budú vďaka tomu, že výsledky vkladáme na začiatok “listu”, najnovšie výsledky zobrazovať na vrchu zoznamu a s každým hodom posúvať nižšie. Kód aký sme pre toto použili môžete vidieť na obrázku 12.



Obrázok 11 Okno s históriou

```
self.historyList.insert(0, '"' + str(self.diceCount()) + 'xD' + str(size) + '": ' + str(value))

self.historyLabel1.text = self.historyList[0]
self.historyLabel2.text = self.historyList[1]
self.historyLabel3.text = self.historyList[2]
self.historyLabel4.text = self.historyList[3]
self.historyLabel5.text = self.historyList[4]
self.historyLabel6.text = self.historyList[5]
self.historyLabel7.text = self.historyList[6]
self.historyLabel8.text = self.historyList[7]
self.historyLabel9.text = self.historyList[8]
self.historyLabel10.text = self.historyList[9]
```

Obrázok 12 Kód pre zobrazenie histórie

2.4.4 Tlačidlo na zatváranie okna výberu kociek a histórie

V oboch oknách sa bude musieť nachádzať spôsob, akým dané okno zatvoríme. Na toto využijeme nové tlačidlo s ikonou „x“ v ľavej, respektíve pravej časti okna. Jeho úlohou bude pri stlačení zmeniť jeho farbu a pri pustení vrátiť obe okná do pozície, v ktorej ich nevidíme a znovu umožniť stlačiť tlačidlá na zobrazenie oboch rozbaľovacích okien. Toto stlačenie sme deaktivovali pre druhé tlačidlo v momente, ako sme stlačili jedno z tlačidiel na výber našich rozbaľovacích okien.

Záver

Cieľom tejto práce bolo vytvorenie programu pre hod kockou s primárnym využitím v stolovej hre Dungeons and Dragons. Všetky funkcie, ktoré bolo potrebné zakomponovať do tohoto programu (hod kockou, výber typu a počtu kociek, zobrazenie histórie) sme dokázali naprogramovať a fungujú správne. Hlavnou časťou celého programu je generovanie náhodných hodnôt na základe funkcie randint() z modulu random, ktorý je v pythone priamo integrovaný.

Posledným krokom bolo vytvorenie užívateľského prostredia, ktoré je jednoducho použiteľné. V prvej verzií bolo síce toto prostredie funkčné, no nie veľmi intuitívne. Preto bolo potrebné začať celý program od začiatku so zameraním na intuitívnosť a vizuálnu jednoduchosť. Vo finálnej verzií sa to výrazne odzrkadlilo a používateľ by nemal mať problém náš program používať.

V prípade, že by sme chceli tento program ďalej rozvíjať, máme na výber mnohé smery, ktorými postupovať. Pri terajších trendoch by bol najlogickejším smerom práve vývoj tohoto programu pre dotykové zariadenia. Takýto vývoj by nevyžadoval veľké technické zmeny, no bolo by potrebné zmeniť dizajn užívateľského prostredia, aby sa dal používať na menších obrazovkách a dokázal sa prispôsobiť rôznym tvarom a veľkostiam zariadení. Druhou možnosťou vývoja by mohlo byť pridanie 3D animácie hodenia kocky. Program by simuloval hod, kotúľanie a zastavenie kociek, pričom by ukončil hod na zobrazení čísel, ktoré sme hodili. Takéto 3D generovanie by sa následne dalo zakomponovať do ešte väčšieho rozšírenia programu o rozšírenú realitu, kde by sme na obrazovkách telefónu videli simulovaný hod priamo na hracej doske.

Zoznam literatúry

Darren Waters. 2004. What happened to Dungeons and Dragons?. [online] 26 Apríl 2004. [cit.2020-12-10]. Dostupné na internete: <http://news.bbc.co.uk/2/hi/uk_news/magazine/3655627.stm>

Cecilia D'Anastasio. 2019. Dungeons & Deceptions: The First D&D Players Push Back On The Legend Of Gary Gygax. [online]. 26 August 2019. [cit.2020-12-10]. Dostupné na internete: <https://kotaku.com/dungeons-deceptions-the-first-d-d-players-push-back-1837516834>

Wizards RPG Team. 2014. Player's Handbook. Piata edícia. Renton, WA : Wizards of the Coast LLC. 9780786965601.

WILLIAMS, J. Patrick - HENDRICKS, Sean Q. - WINKLER W. Keith. 2006. Gaming as Culture, Essays on Reality, Identity and Experience in Fantasy Games. Jefferson, N.C.: McFarland & Company. Strany 1–14, 27. ISBN 0-7864-2436-2.

Die Hard Dice Store. 2020. What are DnD Dice?. [online]. [cit.2020-12-18]. Dostupné na internete: <https://www.dieharddice.com/pages/d-d-dice-explained>

Python Software Foundation. 2020. Organizations Using Python. [online]. [cit.2020-12-18] Dostupné na internete: <https://wiki.python.org/moin/OrganizationsUsingPython>

PETERS, Tim (19 August 2004). PEP 20 – The Zen of Python. [online]. 19 August 2004 [cit.2020-12-18]. Dostupné na internete: <https://www.python.org/dev/peps/pep-0020/>

Python Software Foundation. 2020. Quotes about Python. [online]. [cit.2020-12-18] Dostupné na internete: <https://www.python.org/about/quotes/>

IVANOVÁ-ŠALINGOVÁ, Mária - MANÍKOVÁ, Zuzana. 1979. Slovník cudzích slov A/Z. Prvé vydanie. Bratislava: Slovenské pedagogické nakladateľstvo. strana 53. ISBN 67-567-79.

Obrázky 1 - 6 d20, d10, d6, d12, d8, d4 od Lonnie Tapscott. Prevzaté z thenounproject.com

Príloha A.1 - Verzia Alfa

```
from random import randint

def kocka(numMax):
    print("D"+ str(numMax))
    throw = randint(1,numMax)
    print(throw)



kocka(20)
kocka(12)
kocka(10)
kocka(8)
kocka(6)
kocka(4)
```

Príloha A.2 - Verzia Beta

```
from random import *
from tkinter import *
import tkinter as tk

# Variables
global roll_value
dice_option = 20
roll_value_text = ''

# Style
canvas_background = "#202529"
roll_button_fill = '#32383D'
button_activefill = "#4C555C"

# Buttoncolors
color1 = '#009BFF'
color2 = '#F3F5F8'
button4_fill = color1
button6_fill = color2
button8_fill = color1
button10_fill = color2
button12_fill = color1
button20_fill = color2



# Roll and show number
def roll():
    global roll_value
    global dice_option
    roll_value = randint(1, dice_option)
    print("D" + str(dice_option))
```

```
    print("Hodnota = " + str(roll_value))

    global roll_value_text
    c.delete(roll_value_text)
    roll_value_text = c.create_text(canvas_width / 2, (canvas_height + 170) / 2,
font=("Arial", 300), text=str(roll_value), fill='#F3F5F8')



# On click
def onclick(position):
    x = position.x
    y = position.y
    if x < canvas_width and y < 70:
        roll()
    # Changing text and roll_value
    elif x < d4.x1 and 70 < y < 170:
        d(4)
    elif d4.x1 < x < d6.x1 and 70 < y < 170:
        d(6)
    elif d6.x1 < x < d8.x1 and 70 < y < 170:
        d(8)
    elif d8.x1 < x < d10.x1 and 70 < y < 170:
        d(10)
    elif d10.x1 < x < d12.x1 and 70 < y < 170:
        d(12)
    elif d12.x1 < x < d20.x1 and 70 < y < 170:
        d(20)



# Number Buttons Action
def d(size):
    global dicenumber
    global dice_option
```

```
    dice_option = size
    c.delete(dicenumber)
    dicenumber = c.create_text(canvas_width / 2, 70 / 2, font=("Arial", 30),
text='Roll D' + str(size), fill='#F3F5F8')



# GUI

window = tk.Tk()
window.title('Dice')

canvas_width = 600
canvas_height = 600

c = tk.Canvas(window, width=canvas_width, height=canvas_height,
bg=canvas_background)
c.pack()

# Roll Button
c.create_rectangle(0, 0, canvas_width, 70, width=0, activefill=button_activefill,
fill=roll_button_fill)
# Roll Button Text
dicenumber = c.create_text(canvas_width / 2, 70 / 2, font=("Arial", 30),
text='Roll D20', fill='#F3F5F8')



# Number Buttons
class Ddicebutton:
    def __init__(self, x, x1, color, text):
        self.x = x
        self.x1 = x1
        self.color = color
        self.text = text
```

```
        c.create_rectangle(self.x, 70, self.x1, 170, width=0,
activefill=button_activefill, fill=self.color)
        c.create_text(self.x1 - 50, 120, font=("Arial", 20), text=self.text)



d4 = Ddicebutton(0, 100, button4_fill, '4')
d6 = Ddicebutton(100, 200, button6_fill, '6')
d8 = Ddicebutton(200, 300, button8_fill, '8')
d10 = Ddicebutton(300, 400, button10_fill, '10')
d12 = Ddicebutton(400, 500, button12_fill, '12')
d20 = Ddicebutton(500, 600, button20_fill, '20')

c.bind('<Button-1>', onclick)

tk.mainloop()
```

Príloha A.3 - Verzia 1.0

Oba súbory a priložené obrázky sa musia pre správne fungovanie programu nachádzať v rovnakom priečinku a mať rovnaké názvy.

Python súbor - Názov “Rocnikova_praca_V2.py”:

```
# qpy:kivy
import kivy

kivy.require('2.0.0')

from kivy.app import App
from kivy.core.window import Window
from kivy.lang import Builder
from kivy.properties import ObjectProperty
from kivy.uix.screenmanager import Screen
from random import randint

size = 20


class mainWindow(Screen):
    rollButton = ObjectProperty(None)
    mainText = ObjectProperty(None)
    diceCountVar = ObjectProperty(None)
    historyLabel1 = ObjectProperty(None)
    historyLabel2 = ObjectProperty(None)
    historyLabel3 = ObjectProperty(None)
    historyLabel4 = ObjectProperty(None)
    historyLabel5 = ObjectProperty(None)
    historyLabel6 = ObjectProperty(None)
```

```
historyLabel7 = ObjectProperty(None)
historyLabel8 = ObjectProperty(None)
historyLabel9 = ObjectProperty(None)
historyLabel10 = ObjectProperty(None)
menuButton = ObjectProperty(None)
menuImage = ObjectProperty(None)
diceMenu = ObjectProperty(None)
historyButton = ObjectProperty(None)
menuImage1 = ObjectProperty(None)
xImage = ObjectProperty(None)
xImage1 = ObjectProperty(None)
closeImage = ObjectProperty(None)
minusImage = ObjectProperty(None)
plusImage = ObjectProperty(None)

diceCount = 1
historyList = ['', '', '', '', '', '', '', '', '', '']
nord0 = (46 / 255, 52 / 255, 64 / 255, 1)
nord1 = (59 / 255, 66 / 255, 82 / 255, 1)
nord2 = (67 / 255, 76 / 255, 94 / 255, 1)
nord3 = (76 / 255, 86 / 255, 106 / 255, 1)
nord4 = (216 / 255, 222 / 255, 233 / 255, 1)
nord5 = (229 / 255, 233 / 255, 240 / 255, 1)
nord6 = (236 / 255, 239 / 255, 244 / 255, 1)

def rollValue(self):
    valuelist = []
    value = 0
    for i in range(0, self.diceCount()):
        valuelist.insert(0, randint(1, size))

    for j in range(0, len(valuelist)):
        value += valuelist[j]
```

```
        return valuelist, value

    def rollButton_dwn(self):
        if self.diceCount() == None:
            self.mainText.text = 'Nezadaný počet kociek'

        elif 0 < float(self.diceCount()) <= 15:
            valuelist, value = self.rollValue()
            answer = ''
            if len(valuelist) == 1:
                answer += str(valuelist[0]) + ' '
            else:
                for i in range(0, len(valuelist) - 1):
                    answer += str(valuelist[i]) + ' + '
                answer += str(valuelist[-1])

            self.mainText.text = '"' + str(self.diceCount()) + 'xD' + str(size) + '": ' +
answer + ' = ' + str(value)
            self.historyList.insert(0, '"' + str(self.diceCount()) + 'xD' + str(size) + '": ' +
str(value))

            self.historyLabel1.text = self.historyList[0]
            self.historyLabel2.text = self.historyList[1]
            self.historyLabel3.text = self.historyList[2]
            self.historyLabel4.text = self.historyList[3]
            self.historyLabel5.text = self.historyList[4]
            self.historyLabel6.text = self.historyList[5]
            self.historyLabel7.text = self.historyList[6]
            self.historyLabel8.text = self.historyList[7]
            self.historyLabel9.text = self.historyList[8]
            self.historyLabel10.text = self.historyList[9]
```

```
    elif float(self.diceCount()) == 0:
        self.mainText.text = 'Počet kociek je 0'

    elif float(self.diceCount()) < 0:
        self.mainText.text = 'Počet kociek je záporný'
    elif float(self.diceCount()) > 15:
        self.mainText.text = 'Počet kociek je príliš velký'

def diceSize(self, dice):
    global size
    size = dice
    self.diceCountVar.text = '1'
    self.rollButton.text = 'Hoď ' + str(self.diceCount()) + 'xD' + str(size)

def diceCount(self):
    if self.diceCountVar.text == '':
        return None
    else:
        count = int(self.diceCountVar.text)
        return count

def plusButton_down(self):
    self.plusImage.source = 'plusButton_down.png'

def plusButton_up(self):
    if self.diceCountVar.text == '':
        self.diceCountVar.text = '1'
    elif float(self.diceCountVar.text) < 15:
        self.diceCountVar.text = str(int(self.diceCountVar.text) + 1)
        self.rollButton.text = 'Hoď ' + str(self.diceCount()) + 'xD' + str(size)
    self.plusImage.source = 'plusButton.png'
```

```
def minusButton_down(self):
    self.minusImage.source = 'minusButton_down.png'

def minusButton_up(self):
    if self.diceCountVar.text == '':
        self.diceCountVar.text = '1'
    elif 1 < float(self.diceCountVar.text) <= 20:
        self.diceCountVar.text = str(int(self.diceCountVar.text) - 1)
        self.rollButton.text = 'Hod ' + str(self.diceCount()) + 'xD' + str(size)
    self.minusImage.source = 'minusButton.png'

def historyButton_down(self):
    self.menuImage1.source = 'historyButton.png'

def historyButton_up(self):
    self.menuImage1.source = 'historyButton_white.png'
    self.historyMenu.pos_hint = {'x': 0.7}
    self.menuButton.disabled = True

def menuButton_up(self):
    self.menuImage.source = 'diceButton_white.png'
    self.diceMenu.pos_hint = {'x': 0}
    self.historyButton.disabled = True

def menuButton_down(self):
    self.menuImage.source = 'diceButton.png'

def diceCloseButton_up(self):
    self.xImage.source = 'xButton_white.png'
    self.xImage1.source = 'xButton_white.png'
    self.diceMenu.pos_hint = {'x': 1}
    self.historyMenu.pos_hint = {'x': 1}
    self.menuButton.disabled = False
```

```
        self.historyButton.disabled = False
        if self.diceCountVar.text == '':
            self.diceCountVar.text = '1'
        else:
            self.rollButton.text = 'Hod ' + str(self.diceCount()) + 'xD' + str(size)

    def diceCloseButton_down(self):
        self.xImage.source = 'xButton.png'
        self.xImage1.source = 'xButton.png'

    def closeButton(self):
        Window.close()

    def closeButton_down(self):
        self.closeImage.source = 'closeButton_down.png'


kv = Builder.load_file('my.kv')


class DiceApp(App):
    def build(self):
        Window.clearcolor = mainWindow.nord0
        Window.size = (1280, 720)
        Window.left = 320
        Window.top = 180
        return kv


if __name__ == '__main__':
    DiceApp().run()
```

Kv súbor - Názov “my.kv”:

```
#:kivy 2.0.0
mainWindow:

<RoundedButton@Button>
    background_color: (0,0,0,0)
    background_normal: ''
    canvas.before:
        Color:
            rgb: (59 / 255, 66 / 255, 82 / 255, 1) if self.state == 'normal' else (76 /
255, 86 / 255, 106 / 255, 1)
        RoundedRectangle:
            size: self.size
            pos: self.pos
            radius: [22]

<historyLabel@Label>
    text:''
    size_hint:(0.9, 0.1)
    color: (216 / 255, 222 / 255, 233 / 255, 1)
    text_size: self.size
    font_size: self.height - 32
    halign: 'center'

<sizeButton@RoundedButton>
    size_hint: (0.2, 0.1)
    font_size: self.height - 20
    text_size: self.size
    halign: 'center'
    valign: 'center'

<mainWindow>
    name: 'main'
```

```
size: 1920, 1080
rollButton: rollButton
mainText: mainText
diceCountVar: diceCountVar
menuButton: menuButton
menuImage:menuImage
diceMenu:diceMenu
menuImage1:menuImage1
historyMenu:historyMenu
xImage:xImage
xImage1:xImage1
#closeImage:closeImage
historyButton: historyButton
historyLabel1: historyLabel1
historyLabel2: historyLabel2
historyLabel3: historyLabel3
historyLabel4: historyLabel4
historyLabel5: historyLabel5
historyLabel6: historyLabel6
historyLabel7: historyLabel7
historyLabel8: historyLabel8
historyLabel9: historyLabel9
historyLabel10: historyLabel10
minusImage:minusImage
plusImage:plusImage
FloatLayout:
    size: root.size

    RoundedButton:
        id: rollButton
        text:'Hod' 1xD20'
        text_size: (self.width, None)
        halign:'center'
```

```
    valign:'center'
    multiline: False
    size_hint: (0.45, 0.1)
    font_size: self.height #i f self.height <
    pos_hint: {'center_x':0.5, 'center_y':0.85}
    on_press: root.rollButton_dwn()

Label:
    color: root.nord4
    id: mainText
    text:''
    font_size: rollButton.font_size
    text_size: (self.width, self.height)
    halign:'center'
    valign:'top'
    size_hint: (0.5, 0.8)
    pos_hint: {'center_x':0.5, 'top': 0.8}

Button:
    id: menuButton
    text: ''
    size_hint: (0.1, 0.1)
    pos_hint: {'center_x':0.05, 'center_y':0.9}
    background_color:46 / 255, 52 / 255, 64 / 255, 0
    background_normal:''
    disabled: False
    on_press:root.menuButton_down()
    on_release: root.menuButton_up()
    Image:
        id: menuImage
        source: 'diceButton_white.png'
        center: self.parent.center
        allow_stretch: True
```

```
            keep_ratio: False
            size_hint_x: None
            size_hint_y: None
            height: self.parent.height
            width: self.parent.height

    Button:
        id: historyButton
        text: ''
        size_hint: (0.1, 0.1)
        pos_hint: {'center_x':0.95, 'center_y':0.9}
        background_color:46 / 255, 52 / 255, 64 / 255, 0
        background_normal:''
        disabled: False
        on_press:root.historyButton_down()
        on_release: root.historyButton_up()
        Image:
            id: menuImage1
            source: 'historyButton_white.png'
            center: self.parent.center
            allow_stretch: True
            keep_ratio: False
            size_hint_x: None
            size_hint_y: None
            height: self.parent.height
            width: self.parent.height

FloatLayout:
    id: diceMenu
    size: (root.size)
    pos_hint: {'x': 1}
    canvas.before:
        Color:
```

```
    rgba:(46 / 255, 52 / 255, 64 / 255, 1)
    #rgba:(1,0,0,1)
Rectangle:
    size: self.width - root.width * 0.68, self.parent.height
    pos: self.pos
Color:
    rgba:(0,0,0,0.8)
Rectangle:
    size: root.width - root.width * 0, self.parent.height
    pos: (self.x + root.width * 0.32 , self.y)
Color:
    #rgba:(216 / 255, 222 / 255, 233 / 255, 1)
    rgba:(1,1,1,1)
Rectangle:
    source:'d20Icon.png'
    size:(self.width*0.06,self.width*0.06)
    pos:(self.x+self.width*0.24,self.height*0.8)
Rectangle:
    source:'d12Icon.png'
    size:(self.width*0.06,self.width*0.06)
    pos:(self.x+self.width*0.24,self.height*0.68)
Rectangle:
    source:'d10Icon.png'
    size:(self.width*0.06,self.width*0.06)
    pos:(self.x+self.width*0.24,self.height*0.56)
Rectangle:
    source:'d8Icon.png'
    size:(self.width*0.06,self.width*0.06)
    pos:(self.x+self.width*0.24,self.height*0.44)
Rectangle:
    source:'d6Icon.png'
    size:(self.width*0.06,self.width*0.06)
    pos:(self.x+self.width*0.24,self.height*0.32)
```

```
    Rectangle:
        source:'d4Icon.png'
        size:(self.width*0.06,self.width*0.06)
        pos:(self.x+self.width*0.24,self.height*0.2)

Button:
    id: xButton
    text: ''
    size_hint: (0.1, 0.1)
    pos_hint: {'x':0.25, 'y':0.9}
    background_color:46 / 255, 52 / 255, 64 / 255, 0
    background_normal:''
    on_press:root.diceCloseButton_down()
    on_release: root.diceCloseButton_up()
    Image:
        id: xImage
        source: 'xButton_white.png'
        center: self.parent.center
        allow_stretch: True
        keep_ratio: False
        size_hint_x: None
        size_hint_y: None
        height: self.parent.height
        width: self.parent.height
Label:
    id:vyberkocky
    text:'Výber kocky'
    color: root.nord4
    text_size: (d20Button.width*2, d20Button.height)
    font_size: historyLabel1.font_size + 15
    halign: 'center'
    pos_hint: {'center_x':0.13, 'center_y':0.95}
```

```
sizeButton:
    id: d20Button
    text: 'D20'
    pos_hint: {'center_x':0.13, 'y':0.8}
    on_press: root.diceSize(20)

sizeButton:
    id: d12Button
    text: 'D12'
    pos_hint: {'center_x':0.13, 'y':0.68}
    on_press: root.diceSize(12)

sizeButton:
    id: d10Button
    text: 'D10'
    pos_hint: {'center_x':0.13, 'y':0.56}
    on_press: root.diceSize(10)

sizeButton:
    id: d8Button
    text: 'D8'
    pos_hint: {'center_x':0.13, 'y':0.44}
    on_press: root.diceSize(8)

sizeButton:
    id: d6Button
    text: 'D6'
    pos_hint: {'center_x':0.13, 'y':0.32}
    on_press: root.diceSize(6)

sizeButton:
    id: d4Button
    text: 'D4'
```

```
    pos_hint: {'center_x':0.13, 'y':0.20}
    on_press: root.diceSize(4)

Label:
    text: 'Počet kociek'
    color: root.nord4
    text_size: (d20Button.width*2, d20Button.height)
    font_size: historyLabel1.font_size + 5
    halign: 'center'
    pos_hint: {'center_x':0.13, 'center_y':0.17}

Button:
    size_hint: (0.05, 0.05)
    pos_hint: {'center_x':0.2, 'center_y': 0.07}
    background_color:(0,0,0,0)
    on_press:root.plusButton_down()
    on_release:root.plusButton_up()
    Image:
        id: plusImage
        source: 'plusButton.png'
        size:self.parent.size
        center_x: self.parent.center_x
        center_y: self.parent.center_y
Button:
    size_hint: (0.05, 0.05)
    pos_hint: {'center_x':0.06, 'center_y': 0.07}
    background_color:(0,0,0,0)
    on_press:root.minusButton_down()
    on_release:root.minusButton_up()
    Image:
        id: minusImage
        source: 'minusButton.png'
        size:self.parent.size
```

```
            center_x: self.parent.center_x
            center_y: self.parent.center_y


    Label:
        id: diceCountVar
        text: '1'
        size_hint: (0.1, 0.1)
        pos_hint: {'center_x': 0.13, 'center_y': 0.07}
        background_color: root.nord4
        multiline: False
        border:(0,0,0,0)
        halign: 'center'
        valign: 'center'
        font_size: self.height - 20
FloatLayout:
    id: historyMenu
    size_hint: (0.3, 1)
    #pos_hint: {'x': 0.7}
    pos_hint: {'x': 1}
    canvas.before:
        Color:
            rgba:(46 / 255, 52 / 255, 64 / 255, 1)
            #rgba:(1,0,0,1)
        Rectangle:
            size: self.size
            pos: self.pos
        Color:
            rgba:(0,0,0,0) if self.pos_hint == {'x':1} else (0,0,0,0.8)
        Rectangle:
            size: root.width - root.width * 0.3, self.parent.height
            pos: (0,0)
    Button:
```

```
    id: xButton
    text: ''
    size_hint: (0.1, 0.1)
    pos_hint: {'x':0.05, 'y':0.9}
    background_color:46 / 255, 52 / 255, 64 / 255, 0
    background_normal:''
    on_press:root.diceCloseButton_down()
    on_release: root.diceCloseButton_up()
    Image:
        id: xImage1
        source: 'xButton_white.png'
        center: self.parent.center
        allow_stretch: True
        keep_ratio: False
        size_hint_x: None
        size_hint_y: None
        height: self.parent.height
        width: self.parent.height
Label:
    text:'História:'
    color: root.nord4
    text_size: d20Button.size
    font_size: historyLabel1.font_size + 15
    halign: 'center'
    pos_hint: {'center_x':0.5, 'center_y':0.95}

historyLabel:
    id:historyLabel1
    pos_hint: {'center_x':0.5, 'center_y':0.87 }
historyLabel:
    id:historyLabel2
    pos_hint: {'center_x':0.5, 'center_y':0.79 }
historyLabel:
```

```
    id:historyLabel3
    pos_hint: {'center_x':0.5, 'center_y':0.71 }
historyLabel:
    id:historyLabel4
    pos_hint: {'center_x':0.5, 'center_y':0.63 }
historyLabel:
    id:historyLabel5
    pos_hint: {'center_x':0.5, 'center_y':0.55 }
historyLabel:
    id:historyLabel6
    pos_hint: {'center_x':0.5, 'center_y':0.47 }
historyLabel:
    id:historyLabel7
    pos_hint: {'center_x':0.5, 'center_y':0.39 }
historyLabel:
    id:historyLabel8
    pos_hint: {'center_x':0.5, 'center_y':0.31 }
historyLabel:
    id:historyLabel9
    pos_hint: {'center_x':0.5, 'center_y':0.23}
historyLabel:
    id:historyLabel10
    pos_hint: {'center_x':0.5, 'center_y':0.15}
```

Ikony použité v programe:

Názvy obrázkov podľa poradia:d20Icon, d12Icon, d10icon, d8Icon, d6Icon, d4Icon, xButton_white, xButton, diceButton_white, diceButton, historyButton_white, historyButton, minusButton, minusButton_down, plusButton, plusButton_down