Gi

Vyšší odborná škola a Střední průmyslová škola elektrotechnická Olomouc, Božetěchova 3

**PRAKTICKÁ ZKOUŠKA Z ODBORNÝCH PŘEDMĚTŮ**

2D Hra v Pythonu

Autor **Matouš Zatloukal**

Obor **Elektrotechnika**

Vedoucí práce **Mgr. Jaroslav Krbec**

Školní rok **2021/2022**

Na druhé stránce bude vloženo originální zadání.

*(včetně podpisů vedoucího práce a maturujícího žáka)*

Prohlašuji, že jsem praktickou zkoušku z odborných předmětů vypracoval samostatně a všechny prameny jsem uvedl v seznamu použité literatury.

……………………………  
 Matouš Zatloukal

Chtěl bych vyslovit poděkování panu Mgr. Jaroslavu Krbci za odborné konzultace a poskytnuté informace.

……………………………  
 Matouš Zatloukal

Prohlašuji, že nemám námitek proti půjčování nebo zveřejňování mé práce nebo její části se souhlasem školy.

……………………………  
 Matouš Zatloukal

Abstrakt

Tato práce je zaměřena na kompletní vývoj počítačové hry včetně veškeré logiky a animací. Videohry jsou stále populárnější formou zábavy a to nejen u mladších generací. Zároveň roste i velikost herního průmyslu, kde je pozice programátora jednou z těch nejdůležitějších a nejžádanějších.

Cílem práce je vytvořit 2D hru, která bude obsahovat základní herní mechaniky, jako například pohyb, skákání nebo útočení. Vymyslet logiku těchto mechanik a naprogramovat je ve zvoleném jazyce. Dále pak vytvořit animace postav, za které budou uživatelé hrát, a nakonec naprogramovat interaktivní menu a propojit vše dohromady do jednoho programu.

Toho lze docílit použitím některé z grafických knihoven programovacího jazyku Python, kde zejména dobrá je knihovna Pygame, jelikož ta je specificky navržená pro vývoj her. Dále je pak využit program Aseprite, který je ideálním nástrojem na tvoření jednoduché pixelové grafiky a následné animování.

Hra je, stejně jako velké množství her dnešní doby, určena pro zábavu uživatelů. Slouží ale také pro mne, jakožto autora, abych si vyzkoušel programování bez použití jakýchkoliv dalších nástrojů, které jsou v herním průmyslu běžně využívány a které práci výrazně ulehčují. Je to v hlavně z důvodu pochopení základních herních mechanik a nabytí znalostí o jejich funkčnosti.

Obsah

Obsah se vkládá až po napsání celé práce pomocí menu Odkazy / Obsah. Při jakékoliv změně nadpisů v práci je třeba obsah aktualizovat – pravé tlačítko / Aktualizovat pole / celá tabulka.

[Obsah 4](#_Toc525402557)

[Úvod 4](#_Toc525402558)

[1. Kapitola první 4](#_Toc525402559)

[1.1 Ilustrace 4](#_Toc525402560)

[1.2 Tabulky 4](#_Toc525402561)

[1.3 Vzorce 4](#_Toc525402562)

[1.4 Citace 4](#_Toc525402563)

[1.4.1 Příklad citace z knihy 4](#_Toc525402564)

[1.4.2 Příklad citace z www stránky 4](#_Toc525402565)

[2. Pokyny pro psaní textu 4](#_Toc525402566)

[2.1 Nejčastější nedostatky 4](#_Toc525402567)

[2.2 Druhá podkapitola 4](#_Toc525402568)

[2.3 Třetí podkapitola 4](#_Toc525402569)

[Závěr 4](#_Toc525402570)

[Seznam použité literatury 4](#_Toc525402571)

[Seznam obrázků a tabulek 4](#_Toc525402572)

Pozor do obsahu nezahrnujte abstrakt.

Úvod

V této dlouhodobé maturitní práci se budu zaměřovat na vytvoření 2D hry pomocí objektově orientovaného programování v jazyce Python a jeho modulu Pygame bez pomoci jakýchkoliv herních enginů. Budu zároveň tvořit i interaktivní menu, přes které se bude hra spouštět a do kterého se uživatelé budou vždy vracet.

Do hry budu kreslit i své vlastní obrázky stylem „pixel art“, které budu animovat v editoru obrázků Aseprite.

Hra samotná se bude odehrávat v aréně, kde mezi sebou budou bojovat dva hráči hrající za dvě postavy. Ty se od sebe budou lišit jejich dosahem útoků a rychlostí útoků. Cílem každého z hráčů bude dostat svého protivníka na nula životů.

Rozhodl jsem se pro tuto práci hlavně z důvodu mého budoucího studia vývoje počítačových her. Ty si můžeme zahrát, když si zrovna potřebujeme odpočinout, nebo si jen tak užít volný čas s přáteli či rodinou.

V úvodu práce autor seznamuje čtenáře s obsahem práce a její strukturou. Při psaní se vychází ze zadání práce. Úvod by měl vždy obsahovat cíl práce. Zdůrazní se co práce obsahuje, možno uvést i strukturu práce (tj. stavbu kapitol a co každá kapitola obsahuje). Může se uvést důvody, proč zvolil dané téma.

V úvodu by neměly být použity citace, obrázky, grafy apod. Úvod je vždy psán před započetím tvorby práce tj. v budoucím čase.

# Užitý software

Před tím, než se pustím do popisování vytváření animací, herní logiky nebo samotného programování, bude nutné nejprve uvést mnou užité softwarové nástroje pro vývoj hry.

## Python

Python je vysokoúrovňový, interpretovaný programovací jazyk navržený v roce 1991. Je vyvíjen jako open source projekt, který nabízí instalační balíky pro většinu běžných platforem (Unix, MS Windows, macOS, Android).



Obrázek - Logo programovacího jazyku Python

Zdroj: https://www.python.org/community/logos/

Tento programovací jazyk jsem si vybral hlavně z důvodu, že k němu mám nejblíže, a tedy ho i nejvíce ovládám. Vhodnější by ovšem bylo použít například C++ nebo C#, které jsou k vytváření videoher mnohem lépe uzpůsobené.

## Pygame

Pygame je modul programovacího jazyku Python, který slouží k vytváření a designování videoher. Stejně jako Python je i Pygame je podporován na většině hlavních platforem.



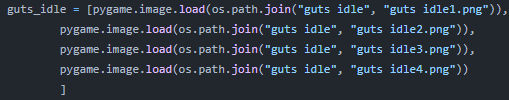
Obrázek - Logo modulu Pygame

Zdroj: https://github.com/pygame/pygame/blob/main/docs/pygame\_logo.png

Programování pomocí modulu Pygame se dělí na dvě části. Deklarování proměnných, příprava funkcí, popř. tříd a poté na herní smyčku, která se neustále opakuje a využívá naše deklarované proměnné, funkce nebo i třídy. V této smyčce také probíhá samotná hra.

## OS

Pomocí modulu OS lze provádět různé interakce s uživatelským operačním systémem. V mé práci jsem tohoto modulu využíval velmi zřídka, a to jen tehdy, když jsem potřeboval načíst obrázky uložené v různých složkách.



Obrázek - Využití modulu OS

Zdroj: Vlastní zpracování

## Random

Modul Random je jeden z nejrozšířenějších Python modulů vůbec. Slouží ke generování pseudonáhodných čísel, což jsou čísla, jejichž posloupnost se zdá být náhodná, ale ve skutečnosti je generována deterministickým algoritmem.

Tento modul jsem podobně jako modul OS využíval jen velmi zřídka, a to pouze, když se náhodně vybírá pozadí arény, ve které se bude hrát.

## Aseprite

Aseprite je program, za jehož pomoci je možné vytvářet obrázky a animace ve stylu pixel art. Stejně jako většina programů na úpravu obrázků podporuje Aseprite vrstvení. Celý program je vyvíjený za pomoci programovacího jazyka C++.



Obrázek - Logo programu Aseprite

Zdroj: <https://github.com/aseprite/aseprite/blob/main/data/icons/ase256.png>

Pro zjednodušení animací pak také umožňuje ukládat jednotlivé obrázky do snímků a zobrazovat jich více zaráz.

Exportování je taktéž velmi snadné, jelikož program umí exportovat do GIF souborů stejně, jako třeba do PNG nebo JPG.

# Logika hry a herní mechaniky

V této kapitole si představíme logiku herních mechanik, jako jsou například pohyby, kolize nebo skákání. Ukážeme si zde také, jak funguje herní menu.

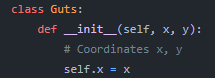
## Herní logika

Podkapitola Herní logika bude zaměřená na hru samotnou, hernímu menu se bude věnovat následující podkapitola.

### Pohyb

U většiny her je pohyb tou nejdůležitější mechanikou a tady tomu není jinak. Jedna z vlastností modulu Pygame je možnost získávat vstupy z kláves na klávesnici. Díky tomu budeme moci monitorovat některé námi specifikované klávesy a poté na jejich zmáčknutí reagovat různými akcemi postav ve hře.

Souřadnice X a Y si budeme ukládat do příslušných proměnných.

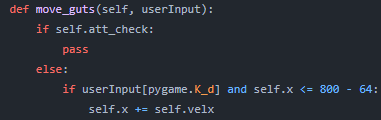


Obrázek , - Ukládání souřadnic

Zdroj: Vlastní zpracování

Prozatím budeme potřebovat pouze souřadnici X, jelikož se budeme pohybovat pouze v ose X na kartézské soustavě souřadnic, se kterou Pygame pracuje. Souřadnici Y budeme potřebovat až v kapitole, která se bude zabývat skákáním.

V herní smyčce pak budeme každou iteraci aktualizovat námi později vytvořenou proměnnou *userInput*, která nám bude zjišťovat vstupy z klávesnice od obou hráčů. Pomocí hodnot této proměnné budeme měnit i pozice hráčů.



Obrázek - Změna souřadnic postavy

Zdroj: Vlastní zpracování

Na tomto snímku je vidět, že pokud je do proměnné *userInput* zaznamenáno zmáčknutí klávesy *d* a postava není pryč z herního okna, které je široké 800 pixelů, tak se k souřadnici X přičte námi definovaná hodnota ve funkci *\_\_init\_\_*.

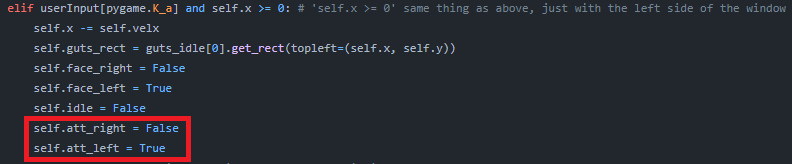
### Útoky

Útoky se budou dělit podle toho, s kterou postavou útočíme. Jedna z nich, Guts, totiž útočí pouze na vzdálenost dosahu svého meče, mezitím co druhá, Mrakoplaš, střílí ohnivé koule po celé šířce arény.

Při útočení s Gutsem budeme potřebovat následující proměnné, které nám pomohou zjistit, na kterou stranu má být při animaci útoku natočen a kdy započít útok.

Bez názvu.png

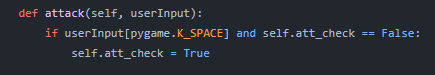
Tyto proměnné se nacházejí ve funkci *\_\_init\_\_* ve třídě *Guts. Att\_right* a *att\_left* se mění při běhání s postavou na pravou či levou stranu.



Obrázek - Příklad změny proměnné při pohybu doleva

Zdroj: Vlastní zpracování

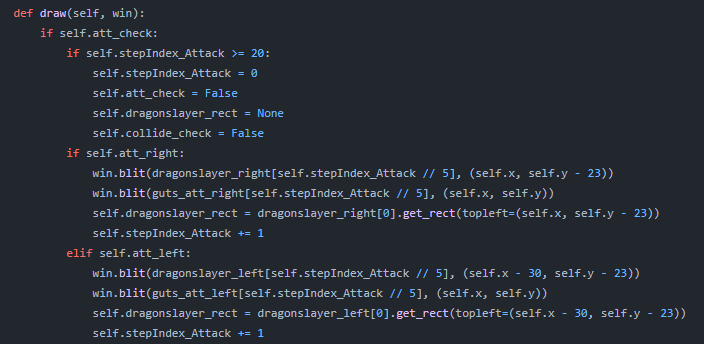
Pomocí proměnné *userInput* opět zjistíme, zda je naše chtěná klávesa stlačena a pokud tomu tak je, nastavíme *att\_check* na True.



Obrázek - Funkce útočení třídy Guts

Zdroj: Vlastní zpracování

Také se nesmí zapomenout ohlídat, aby postava nemohla útočit, pokud už útočí. Pokud tedy víme na kterou stranu se postava dívá a víme, zda hráč zmáčkl příslušnou klávesu, můžeme začít vykreslovat animaci útoku.

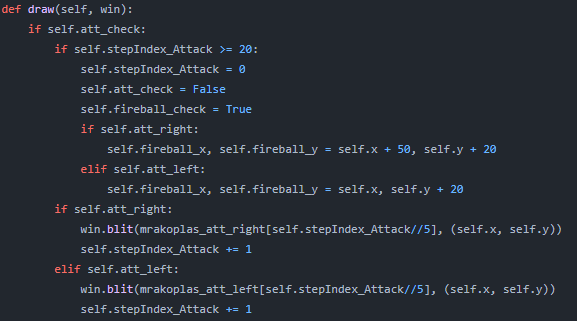


Obrázek - Vykreslení útoku Gutse

Zdroj: Vlastní zpracování

Prvně se provede kontrola, zda se už nedokončila animace, a nemá se tedy útok ukončit, jelikož toto vykreslování probíhá ve smyčce. Následně se zkontroluje, zda se postava dívá na levou či pravou stranu a poté se vykreslí animace těla a meče v daném snímku na dané souřadnice. U této postavy je animace meče a těla samotného rozdělena, jelikož později, až budeme chtít provádět kolize, budeme chtít kolidovat Mrakoplaše pouze s mečem a ne s celým tělem Gutse, což se bude dělat jednodušeji, pokud animace budou rozděleny na dvě.

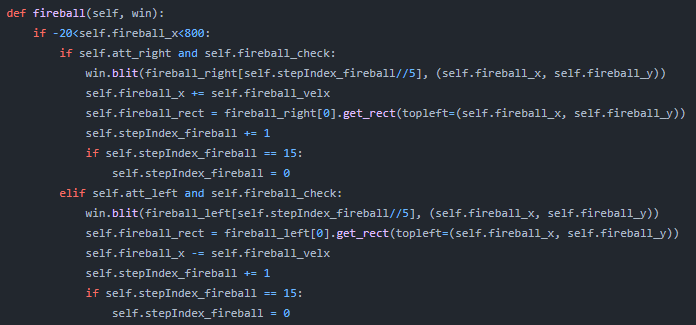
Krom proměnných *att\_right*, *att\_left* a *att\_check* budeme u Mrakoplaše potřebovat i proměnnou *fireball\_check*, která se opět nachází ve funkci *\_\_init\_\_*.



Obrázek - Vykreslení útoku Mrakoplaše

Zdroj: Vlastní Zpracování

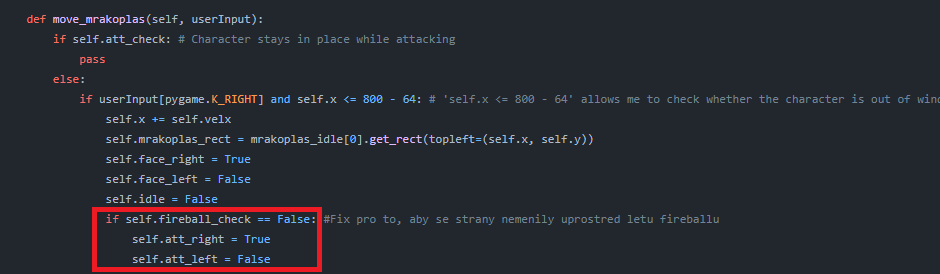
Po ukončení animace se aktivuje proměnná *fireball\_check* a přejde se k vykreslování samotné ohnivé koule, která bude poté kolidovat s tělem Gutse a působit poškození.



Obrázek - Vykreslování projektilu Mrakoplaše

Zdroj: Vlastní zpracování

Při změně proměnných *att\_right* a *att\_left* se ovšem musí ošetřit, aby hráč nemohl měnit směr projektilu po jeho vystřelení.



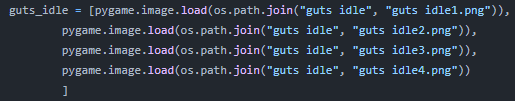
Obrázek - Ošetření změny směru projektilu

Zdroj: Vlastní zpracování

Proměnné se nám tedy mění pouze, pokud projektil není vystřelený. Projektil mizí, pokud koliduje s druhou postavou, nebo se dostane za hranu arény.

### Animace

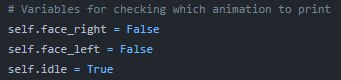
Obrázky jednotlivých animací si prvně musíme nahrát.



Obrázek - Nahrání obrázků pro animace

Zdroj: Vlastní zpracování

Ve funkci *\_\_init\_\_* nalezneme proměnné *face\_right*, *face\_left* a *idle*, které nám pomohou sledovat, které animace pohybů se právě mají přehrávat.



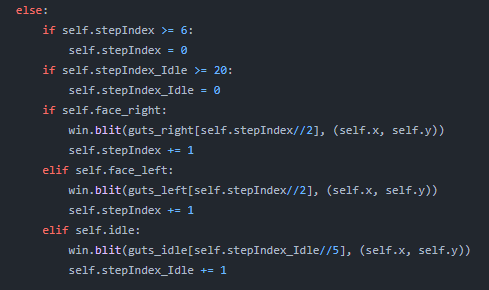
Obrázek - Proměnné pro animace pohybu

Zdroj: Vlastní zpracování

Tyto proměnné se mění na základě toho, zda naší postavou pohybujeme a popřípadě kam.

Další proměnnou, kterou budeme potřebovat je *stepIndex*. Ten nám pomůže zjistit, který snímek animace se má právě vykreslit, jelikož pomocí této proměnné budeme poukazovat do listu obrázků na snímek, který požadujeme.

Ve funkci *draw* pak uplatníme všechny námi připravené proměnné.



Obrázek - Funkce draw

Zdroj: Vlastní zpracování

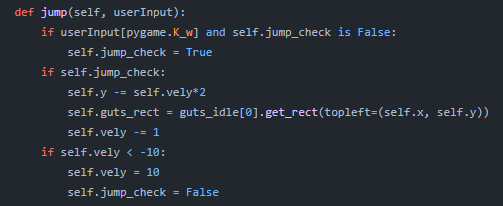
Vykreslování opět probíhá ve smyčce, takže nejprve zkontrolujeme, zda se *stepIndex* nedostal k námi stanovené maximální hodnotě, kde další hodnota už by ukazovala mimo list a nastala by tedy chyba, a pokud ano, tak ho vynulujeme. Poté zjistíme, zda se postava pohybuje doleva či doprava a vykreslíme jeden snímek animace. Nakonec přičteme ke *stepIndexu* jedničku, aby při příští iteraci poukazoval na následující obrázek v listu, a tím vytvoříme animaci.

Může se ovšem stát, že animace budou vypadat příliš rychlé. To je zapříčiněno malým počtem snímků v animaci, rychlým zobrazováním snímků na obrazovku a nebo kombinací těchto dvou věcí. V mém případě zobrazuji na obrazovku 60 snímků za sekundu a moje animace má maximálně 5 snímků.

Musím tedy vykreslovat stejný snímek během více iterací, aby animace nevypadala tak nepřirozeně a rychle. Toho docílíme tak, že vynásobíme počet snímků určitým přirozeným číslem, tím číslem následně budeme dělit *stepIndex*, který poukazuje na snímek při vykreslování, a výsledek tohoto násobení stanovíme jako maximální hodnotu *stepIndexu* při které se nuluje. Jeden obrázek nám tedy zůstane vykreslen déle a animace si takto můžeme libovolně protahovat.

### Skákání

Jednou z dalších důležitých mechanik je skákání. To nám umožní se lépe pohybovat po aréně.



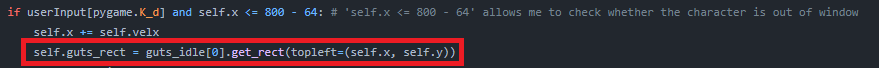
Obrázek - Funkce jump

Zdroj: Vlastní zpracování

Pomocí proměnné *userInput* je opět načítán klávesový vstup. Také se před začátkem skoku musí zkontrolovat, zda už postava neskáče. Pokud je načten správný klávesový vstup a postava neskáče, započne odčítání od proměnné *y*, která zaznamenává pozici na ose Y. Velikost proměnné, která je odčítána od proměnné *y*, se zmenšuje každou iteraci až do doby, kdy se čísla převrátí na záporné, a začne se tedy přičítat. Díky tomu docílíme efektu gravitace, kde čím výše je postava ve vzduchu, tím pomaleji stoupá, až se zastaví úplně, a začne klesat. Při docílení opačné hodnoty *vel.y* se postava zastaví a proměnná se vyresetuje. Tím docílíme toho, že postava skončí na stejné pozici Y, jako když začal skok.

### Kolize

Před vytvářením kolizí mezi postavami je nutné vytvořit rectangle objekty zbraní, ohnivých koulí a těl postav. Rectangle objekty slouží v Pygamu k ukládání a manipulaci s obdélníkovými oblastmi na obrazovce.

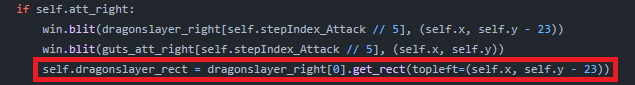


Obrázek - Rectangle objekt postavy Guts

Zdroj: Vlastní zpracování

Zde je vidět aktualizování pozice rectanglu těla Gutse při jeho pohybu doprava. Jako obrázek je možné uvést jakýkoliv snímek jakékoliv animace, jelikož všechny mají velikost 64 pixelů. Souřadnice se ovšem musí uvést aktuální, aby se rectangle pohyboval zároveň s pohybem postavy.

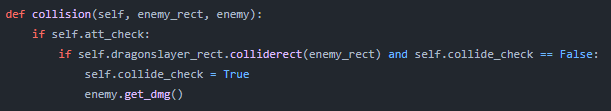
Při útoku Gutse se jako rectangle nastaví prostor celé animace jeho meče.



Obrázek - Rectangle objekt Gutsova meče

Zdroj: Vlastní zpracování

Jako obrázek může být uveden opět kterýkoliv snímek animace, musí se ovšem dát pozor, aby to byla animace správného směru útočení. V tomto případě animace útoku doprava. Jako souřadnice jsou uvedeny upravené souřadnice příslušné postavy tak, aby animace útoku meče byla seřazená vůči animaci útoky postavy samotné.



Obrázek - Funkce collision

Zdroj: Vlastní zpracování

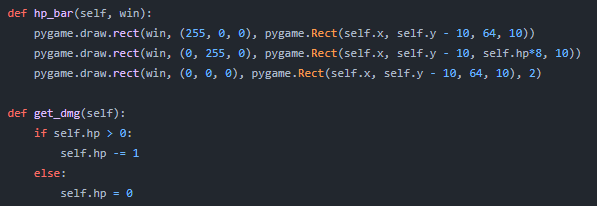
Nyní tyto dva rectangly mohou kolidovat. Do funkce jako *enemy\_rect* předáme rectangle nepřítele (tedy Mrakoplaše). A jako *enemy* předáme funkci objekt Mrakoplaše, aby se mohla vyvolat funkce *get\_dmg*.

Při kolizi Gutse a ohnivé koule od Mrakoplaše je postup stejný s tím rozdílem, že se rectangle koule vytvoří až po jejím seslání a ne už při samotné animaci.

### Health bar

Na kolize bude hra reagovat ubíráním určitého počtu životů z maxima dané postavy. Tuto reakci budeme zobrazovat pomocí ukazatele životů.

Začneme tedy vykreslením barevného ukazatele.



Obrázek - Funkce hp\_bar a get\_dmg

Zdroj: Vlastní zpracování

Při každé iteraci herní smyčky budeme vykreslovat červený pásek, zobrazující chybějící životy postavy. Ten bude překrýván zeleným páskem, zobrazujícím počet životů, který postupným zmenšováním na základě vyhodnocení kolize bude odhalovat červený pásek a tím indikovat ztracené životy. Nakonec vykreslíme černé ohraničení pro lepší viditelnost ukazatelů.

Funkci *get\_dmg* voláme pokud nastane kolize, aby odečetla určitý počet životů. Tyto dvě funkce nám společně zajistí, aby postavy ztrácely životy, když kolidují s nepřátelskou zbraní a aby se tyto životy správně vykreslovaly na obrazovku.

## Menu

V této podkapitole se budu věnovat principům fungování interaktivního menu.

Přes menu mohou uživatelé spustit nebo vypnout hru a taktéž se sem vracejí, pokud byl dohrán již dříve spuštěný zápas.

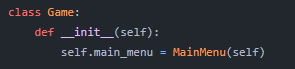
### Parametr *game*

Parametr *game* je jedním ze základních kamenů potřebných pro fungování mnoha funkcí v mém programu. Při definování funkcí třídy *Game*, která slouží, jako herní smyčka, chceme častokrát použít proměnné a funkce třídy *MainMenu*. Toho lze docílit pomocí vytvoření objektu *MainMenu*, na který pak můžeme jednoduše odkazovat v objektu *Game.* Chceme ovšem zároveň využívat proměnné a funkce třídy *Game* při definování funkcí ve třídě *MainMenu*. Pokud tedy přidáme třídě MainMenu parametr *game*, do kterého později v objektu *Game* vložíme sebe samého neboli *self*, můžeme používat všechny funkce a proměnné námi definované ve třídě *Game*.

Bez názvu.png

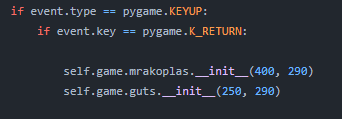
Obrázek - Paramert game u třídy MainMenu

Zdroj: Vlastní zpracování



Obrázek - Vytváření objektu MainMenu

Zdroj: Vlastní zpracování

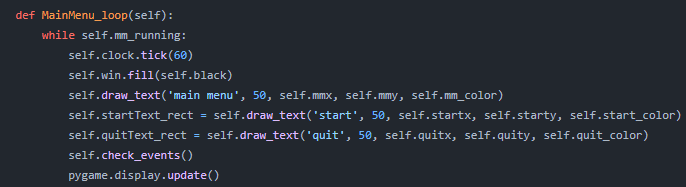


Obrázek – Příklad odkazování pomocí game argumentu ve třídě MainMenu

Zdroj: Vlastní zpracování

### Smyčka *MainMenu*

Když se program spustí, musí se ihned vstoupit do nějaké smyčky, ve které se setrvá, dokud nebude zahájen zápas nebo ukončena hra, a do které se hráči budou vracet po dohrání zápasu. Právě takovou smyčkou se bude zabývat tato podkapitola.



Obrázek - Funkce MainMenu smyčky

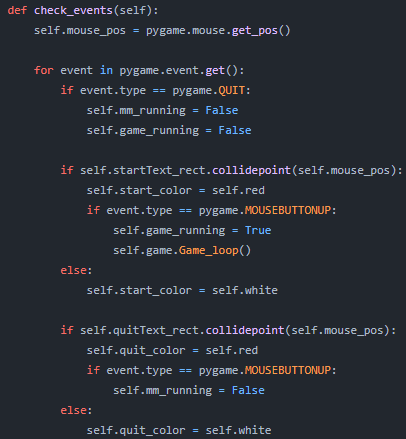
Zdroj: Vlastní zpracování

V této smyčce vykonává program následující akce neustále dokola:

* Uzpůsobí rozestup mezi snímky tak, aby byl počet snímků za sekundu maximálně 60 (zespodu nelze hranici omezit, jelikož tam závisí na výkonnosti počítače)
* Vyplní okno černou barvou
* Nakreslí text „main menu“
* Nakreslí text „start“ a zaznamená ho jako rectangle, se kterým se později bude moct kolidovat
* Nakreslí text „quit“ a taktéž ho zaznamená, jako rectangle
* Provede funkci *check\_events*
* Aktualizuje okno, aby se provedly všechny změny pro daný snímek

### Kontrola událostí v menu

Funkce *check\_events* slouží ke kontrole událostí, neboli „eventů“, jak se jim říká v Pygamu. Za tyto eventy se považuje jakákoliv akce provedená uživatelem, jako např. stisk klávesy nebo pohyb myší.



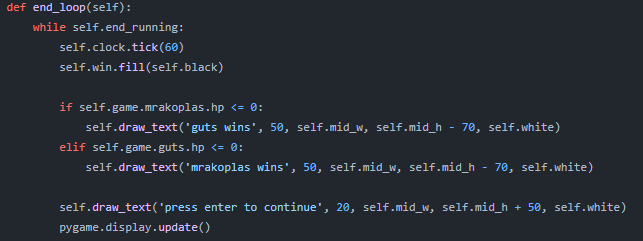
Obrázek - Funkce check\_events

Zdroj: Vlastní zpracování

V naší funkci se nejprve zaznamenají souřadnice myši pro daný snímek, jelikož je budeme později potřebovat při kolidování. Poté funkce zkontroluje, zda uživatel náhodou nevypnul okno a pokud ano, ukončí všechny možné probíhající smyčky. Následně proběhne kontrola kolize s textem „start“. Pokud myš a rectagnle textu kolidují, tak se text vybarví červeně. Pokud uživatel během kolize zmáčkne tlačítko na myši, spustí se herní smyčka. Kolize s textem „quit“ fungují stejně, akorát při zmáčknutí tlačítka na myši se hra vypne.

### Smyčka po konci hry

Poslední věc, která chybí bezproblémovému přechodu z menu do hry a zpět je *end\_loop*. Funkce *end\_loop* se nachází ve třídě *Menu* a slouží, jako smyčka, ve které program setrvá, jakmile jeden z hráčů v zápase klesne pod 0 životů.

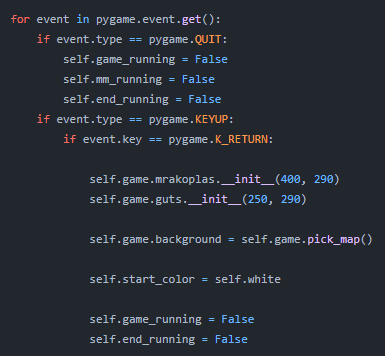


Obrázek - První část funkce end\_loop

Zdroj: Vlastní zpracování

V této smyčce program vykonává následující akce:

* Uzpůsobí rozestup mezi snímky tak, aby byl počet snímků za sekundu maximálně 60
* Vyplní okno černou barvou
* Zkontroluje, která z postav má počet životů 0 a podle toho vykreslí text oznamující výherce zápasu
* Vykreslí text indikující uživateli, jak pokračovat zpět do menu
* Aktualizuje obrazovku



Obrázek - Druhá část funkce end\_loop

Zdroj: Vlastní zpracování

Ve druhé části funkce *end\_loop* probíhá kontrola eventů, které mohou nastat při iterování touto smyčkou. Prvně se kontroluje, zda uživatel nezavřel okno a případně se zastaví všechny spuštěné smyčky. Následně se kontroluje, jestli uživatel nestiskl klávesu *return* neboli *enter* a pokud ano, vytvoří se nové objekty postav, které nahradí ty stávající, to hlavně z důvodu resetování všech proměnných, které se postavě v zápase měnily, poté se vybere jedno ze dvou pozadí arén a vypnou se smyčky *end\_loop* a *Game\_loop*. Tím se program vrátí do smyčky *MainMenu\_loop*, odkud mohou uživatelé znovu zahájit zápas.

# Animace

Plynulejší pocit při hraní dodávají animace. V této kapitole se budeme zabývat animacemi postav při stání na místě, běhu nebo útoku, které využívám v mém programu. Zejména vytvářením jednotlivých snímků, jejich animováním a implementací do hry.

## Inspirace a vytvoření prvního snímku

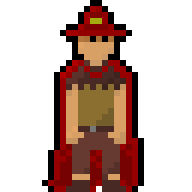
Při tvorbě mých postav jsem se do značné míry inspiroval charaktery z mých oblíbených knih. Jména jsem oběma postavám zanechal stejné, jako v knihách.



Obrázek – Ilustrace Mrakoplaše od Paula Kidbyho

Zdroj: Kniha Výtvarné umění Zeměplochy

První z nich je Mrakoplaš z knižního cyklu Zeměplocha od autora Terryho Pratcheta, kterého jsem překreslil pomocí využití stylu pixel art. Obrázek jsem poté zvětšil do rozlišení 64 pixelů.



Obrázek - Překreslení Mrakoplaše pomocí pixel art

Zdroj: Vlastní zpracování

Druhou z postav je Guts z manga série Berserk, kterou psal a kreslil Kentaró Miura.



Obrázek - Panel z mangy Berserk

Zdroj: Manga Berserk Vol. 29

Toho jsem taktéž stylem pixel art překreslil do obrázku, který jsem později zvětšil na 64 pixelů.



Obrázek - Překreslení Gutse pomocí pixel art

Zdroj: Vlastní zpracování

## Vytváření animací

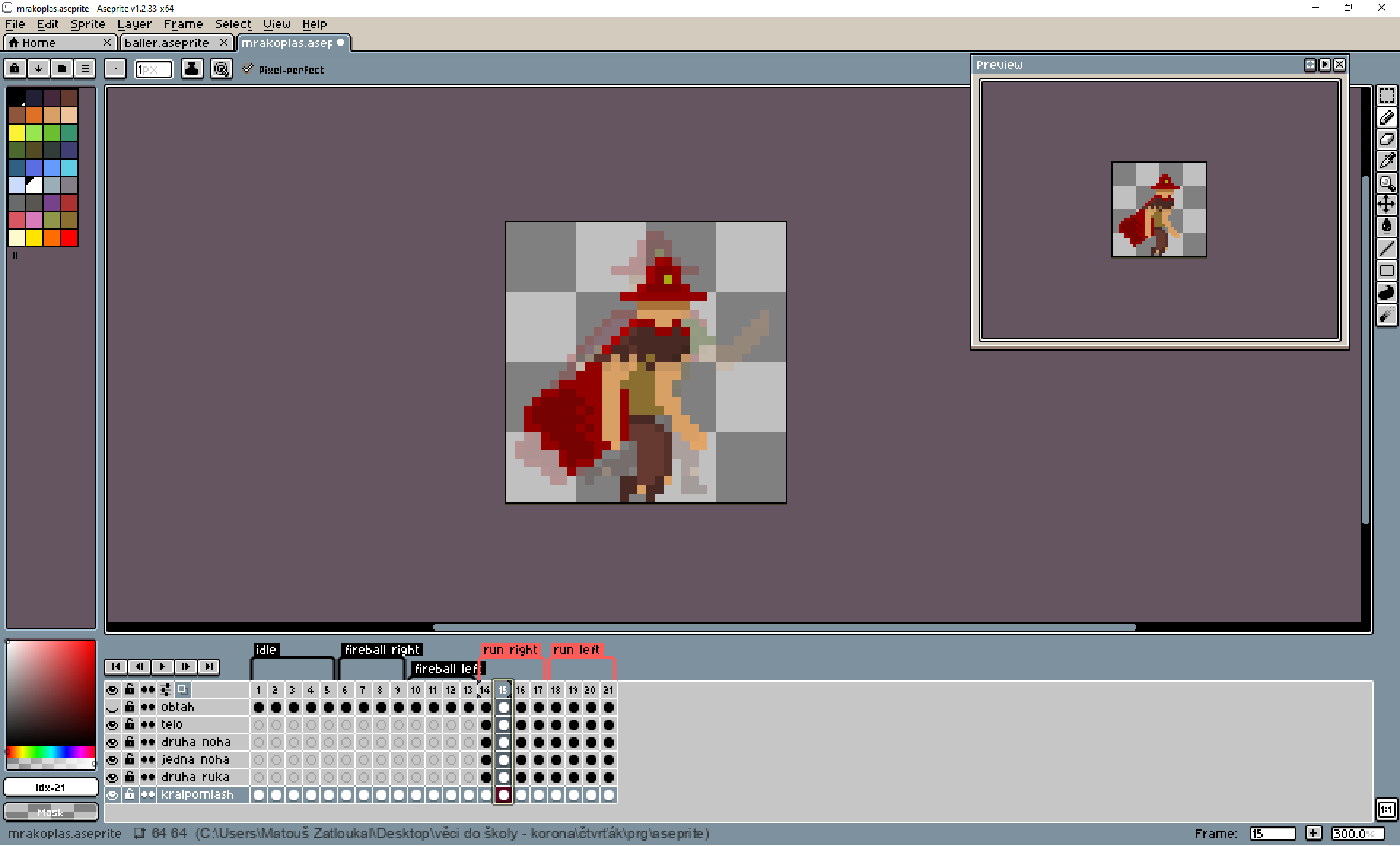
Jelikož je animace jen sled obrázků, vykreslených rychle za sebou, bude potřeba vytvořit další snímky.



Obrázek - Snímky animací Mrakoplaše

Zdroj: Vlastní zpracování

Zde jsou vyobrazeny všechny snímky animací Mrakoplaše krom těch, které jsou pouze vertikálně převráceny (např. běh doleva).



Obrázek - Kreslení animací

Zdroj: Vlastní zpracování

Při tvoření animací budeme využívat hlavně funkci zobrazování předchozích snímků která nám pomůže při vymýšlení pohybů pro následující snímek.



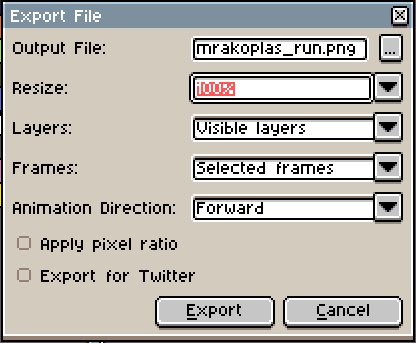
Obrázek - Snímky animací Gutse

Zdroj: Vlastní zpracování

Zde můžeme lépe vidět, jak je rozdělena tělo Gutse při útoku a jeho meč. Kolidování Mrakoplaše s mečem je pak mnohem jednodušší na realizaci, jelikož pouze vezmeme rectangle meče.

## Implementace animací do hry

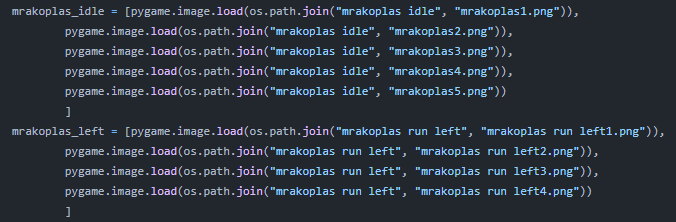
Animace jsou z programu Aseprite exportovány do obrázků ve formátu png, vždy se jeden snímek animace rovná jednomu exportovanému obrázku.



Obrázek - Exportování v programu Aseprite

Zdroj: Vlastní zpracování

Takto exportované obrázky seřadíme do seznamů v našem programu, aby se daly následně používat při animacích a my na ně do seznamu mohli odkazovat pomocí *stepIndexů*.



Obrázek - Načtení obrázků do listů v programu

Zdroj: Vlastní zpracování

## Ilustrace

Je použit styl Nadpis 2.

Všechny ilustrace (obrázky, grafy, fotografie, mapy, diagramy apod.) se jednotně označují jako „Obrázek“. Každý obrázek má své číslo a popisek, který stručně popisuje danou ilustraci. Pokud chcete číslování provádět automaticky, použijte menu **Odkazy / Vložit titulek**. Tím se bude číslo automaticky zvětšovat. Pod každým obrázkem musí být uveden zdroj, ze kterého je obrázek převzat. Příklady jsou uvedeny níže.

Obrázek č. 1 Elektromagnetický ventil

Zdroj: https://www.marsystems.cz

Grafy by měly být srozumitelné, přehledné a musí mít požadovanou vypovídací hodnotu. Můžeme využít barevného tisku a možnosti zobrazit několik průběhů v jednom obrázku. U grafů je nutno uvést název grafu a popis os (veličiny, jednotky). V případě zobrazení více průběhů v jednom grafu je nutno vytvořit odpovídající legendu. Velikost rozměrů grafu volíme tak, aby byl graf čitelný.

Obrázek č. 2 Graf spotřeby plynu v letech 2010, 2011 a 2012

Zdroj: PLÍVA, Z., Metodika zpracování bakalářských a diplomových prací. 2. upravené vydání. Liberec.

Fotografie musíte před vložením do textu vhodně upravit, ořezat rozměr, zpravit soudkovitost, světlost, snížit rozlišení atd. K tomu slouží různé programy, ve škole používáme ZonerPhotoStudio, Gimp, Adobe Photoshop. Abyste nemuseli fotografii složitě upravovat, již před jejím vyfotografováním si rozmyslete, jak by vypadala co nejlépe (pozadí, úhel fotografování, vzdálenost od objektivu, osvětlení, použití blesku atd.).

Obrázek č. 3 Nevhodná fotografie – nevhodné pozadí, nečitelné popisky, malá velikost.

Zdroj: Vlastní zpracování

Obrázek č. 4 Vhodně upravená fotografie

Zdroj: Vlastní zpracování

## Tabulky

Tabulky je potřeba také označit, očíslovat a stručně napsat název tabulky. U veličin je nutné uvést jednotky, záhlaví oddělit rámečkem, barevně.

Tabulka č. 1 Měření zesílení operačního zesilovače

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| R2 [kΩ] | 5 | 10 | 20 | 40 | 60 | 80 | 100 |
| UO [V] | -0,9953 | -1,9887 | -3,974 | -7,946 | -11,918 | -13,170 | -13,196 |
| AU,měřené [-] | -0,250 | -0,499 | -0,997 | -1,993 | -2,989 | -3,303 | -3,310 |
| AU,vypočtené [-] | -0,250 | -0,500 | -1,000 | -2,000 | -3,000 | -4,000 | -5,000 |

Zdroj: Vlastní zpracování

## Vzorce

Použité vzorce píšeme na samostatný řádek, který zarovnáme doprostřed. Na pravé straně píšeme do závorky číslo vzorce, kterým se v dalším textu můžeme na vzorec odvolávat. K psaní vzorců použijeme editor rovnic z menu Vložení / Rovnice.

(1)

(2)

Proud jsme vypočítali podle vzorce (1) a dosadili do vzorce (2). Tím jsme vypočítali výkon na rezistoru R1.

## Citace

Ve svých pracích chcete někdy doslova ocitovat jiného autora. Abyste se vyhnuli nařčení z „vykrádání“ myšlenek a slov někoho druhého a nebyli obviněni z porušování autorských práv, je nutné každou použitou citaci správně označit.

Podobná situace nastává, chcete-li v práci použít obrázek, který jste od někoho převzali, třeba z internetu.

Jak to tedy uděláme?

Pokud použijete v textu převzatý (zkopírovaný) text, musíte na jeho konci dát číslo citace. Na konci stránky se potom uvede poznámka pod čarou, kde konkrétně uvedeme zdroj citace.

### Příklad citace z knihy

Následující odstavec je převzat z knihy Operační zesilovače v elektronice.

Boucherotův člen je běžně zapojován u většiny výkonových integrovaných zesilovačů. Upravuje impedanční poměry výstupu na vysokých frekvencích tak, aby nedocházelo k nežádoucím oscilacím [[1]](#footnote-1).

Na konci slova oscilacím vložíte poznámku pod čarou manu **Odkaz / Vložit pozn. pod čarou**. Tím se vytvoří odkaz **1** a na konci stránky se doplní čára s číslem odkazu. Tam doplníte název knihy.

Jak správně napsat název? Pomůže nám [www.citace.com](http://www.citace.com), kde si knihu podle ISBN nebo názvu najdete a citaci si necháte vygenerovat.

### Příklad citace z www stránky

Následující odstavec je převzat z <https://cs.wikipedia.org/wiki/Opera%C4%8Dn%C3%AD_zesilova%C4%8D>.

Operační zesilovače byly původně vyvinuty pro realizaci [matematických operací](https://cs.wikipedia.org/wiki/Matematick%C3%A1_operace) (odtud pak jejich název) v éře [analogových počítačů](https://cs.wikipedia.org/wiki/Analogov%C3%BD_po%C4%8D%C3%ADta%C4%8D).

První operační zesilovače byly konstruovány z [elektronek](https://cs.wikipedia.org/wiki/Elektronka) a později se přešlo na [diskrétní](https://cs.wikipedia.org/wiki/Diskr%C3%A9tn%C3%AD) [polovodičové](https://cs.wikipedia.org/wiki/Polovodi%C4%8D) součástky. Dnešní operační zesilovače jsou téměř výhradně konstruovány jako [integrované obvody](https://cs.wikipedia.org/wiki/Integrovan%C3%BD_obvod), přičemž často jeden takový obvod sdružuje několik OZ.

První integrované operační zesilovače pocházejí z konce 60. let 20. století. Vůbec první byl obvod [Fairchild](https://cs.wikipedia.org/w/index.php?title=Fairchild_(obvod)&action=edit&redlink=1" \o "Fairchild (obvod) (stránka neexistuje)) μA709, ale ten byl brzy vytlačen obvodem [μA741](https://cs.wikipedia.org/w/index.php?title=%CE%9CA741&action=edit&redlink=1), který je naprostou klasikou ve světě operačních zesilovačů a vyrábí jej mnoho firem v mnoha provedeních dodnes. Oba dva uvedené typy jakož i řada dalších OZ jsou konstruovány pouze z bipolárních [tranzistorů](https://cs.wikipedia.org/wiki/Tranzistor) [[2]](#footnote-2).

# Pokyny pro psaní textu

* Další kapitola začíná vždy na nové stránce.
* Mezera mezi slovy se dělá jen jedna. Chybně je to takto .
* Před tečkou na konci věty se mezera nedělá. Mezi větami se dělá jedna mezera.
* Horní a dolní indexy je nutné psát takto m3, H2O. Používáme k tomu přímo funkci indexy (ikonky X2, X2).
* Psaní číslovek. Napíšete-li 3. (tj. s tečkou) znamená to třetí. Ale napíšete-li 3 (bez tečky) znamená to tři.
* Procento má jiný význam píše-li se s mezerou nebo bez mezery. Napíšeme-li 12 %, znamená to dvanáct procent. Napíšeme-li 12%, potom to je dvanáctiprocentní.
* Pomlčka se odděluje z obou stran mezerami, např. zápas Sparta – Slavia.
* Závorky a uvozovky přiléhají k začátku a konci textu bez mezer, od ostatního textu se oddělí mezerami (Je to „maskot“ družstva.).
* Značky jednoduchých měřících jednotek se píší s mezerou mezi hodnotou a jednotkou (tj. 10 mm, +6 V, 8,77 kg, 7 x 5 m, 10:00 h, 12 min, -15 ºC, 300 kW, 20 VA). Pokud mezeru neuděláte, znamenalo by to číslovku řadovou tedy, 10mm znamená desetimilimetrový.
* Mezní úchylky, 5 ±1, 12 ±0,05, pokud jsou uvedeny měrné jednotky, píšou se s číslem rozměru do závorky: (44 ±0,1) mm, (-13 ±2) ºC.

## Nejčastější nedostatky

* V práci jsou gramatické chyby (interpunkce, skloňování, časování aj.) Je možno použít internetovou jazykovou příručku http://prirucka.ujc.cas.cz
* V práci jsou neopravené chyby (např. překlepy, dělení slov aj.) Využijte kontrolu pravopisu menu Revize / Pravopis a gramatiky.
* Je zřejmé, že autor před odevzdáním práci nekontroloval.
* Nevhodná úprava textů, obrázků a tabulek (např. nezarovnaný text, různé typ písma, nejednotné okraje, nevhodné zvýrazňování aj.)
* Jsou použity různé druhy písma. To svědčí jen o tom, že autor text jen kopíroval.
* Obrázky, tabulky, vzorce aj. nejsou očíslovány a označeny názvem.
* Věty jsou buď příliš dlouhé, nebo se naopak používají jen holé věty.
* Používají se hovorové a slangové výrazy atp.
* Nejsou jasné způsoby výpočtu výsledků a údajů.

## Druhá podkapitola

Takto pokračujeme dále až do konce. Opět styl odstavce Nadpis 2.

## Třetí podkapitola

Text

Závěr

V této kapitole práce se autor věnuje zhodnocení celé práce. Je vhodné hodnotit práci podle bodů zadání. Uveďte, co jste měli udělat, jak jste to udělali a s jakými výsledky, poznatky, úspěchy či neúspěchy, vyjádřit se k splnění cíle práce. Je nutné zaujmout konkrétní stanovisko k jednotlivým výstupům práce.

Závěr je psán v první osobě jednotného čísla, v minulém čase. Rozsah závěru je jedna až dvě strany.

Nestačí napsat, že se „něco vytvořilo“, ale konkrétně s jakými výsledky, nedokonalosti je potřeba zdůvodnit, uvést možné nápravy, náměty na další práci, výhledy do budoucna atd.

Seznam použité literatury

[1] PLÍVA, Z., J. DRÁBKOVÁ, J. KOPRNICKÝ a L. PETRŽÍLKA. Metodika zpracování bakalářských a diplomových prací. 2. upravené vydání. Liberec: Technická univerzita v Liberci, FM, 2014. ISBN 978-80-7494-049-1.

[2] Wikipedie Otevřená encyklopedie, Operační zesilovač [online].

Dostupné z: [https://cs.wikipedia.org/wiki/Opera%C4%8Dn%C3%AD\_zesilova %C4%8D](https://cs.wikipedia.org/wiki/Opera%C4%8Dn%C3%AD_zesilova%20%09%C4%8D)

Pokud nevíte jak správně citovat literaturu, podívejte se na tento web: <http://www.citace.com/>

Seznam obrázků a tabulek

Zde se vkládá jí seznamy obrázků a tabulek. Vložíte to v menu Odkazy / Vložit seznam obrázků, kde vyberete v Obecných / Popisek titulu Obrázek č. a potom Tabulka č.

[Obrázek č. 1 Elektromagnetický ventil 4](#_Toc509907309)

[Obrázek č. 2 Graf spotřeby plynu v letech 2010, 2011 a 2012 4](#_Toc509907310)

[Obrázek č. 3 Nevhodná fotografie – nevhodné pozadí, nečitelné popisky. 4](#_Toc509907311)

[Obrázek č. 4 Vhodně upravená fotografie 4](#_Toc509907312)

[Tabulka č. 1: Měření zesílení operačního zesilovače 4](#_Toc450839160)

Přílohy

V přílohách se uvádějí složitější obrázky, grafy, schémata zapojení, schémata desek plošných spojů, výpisy programů, fotografie a podobně, které není vhodné dávat do základního textu práce.

Každá příloha je číslovaná a pojmenovaná stejně jako obrázky, včetně zdrojů.

Příloha č. 1: Poster k maturitní práci.

Zde bude **povinně vložen obrázek vytvořeného posteru** ve formátu A4, barevný, orientovaný na výšku popř. i šířku.

1. PUNČOCHÁŘ, Josef. *Operační zesilovače v elektronice*. 2. vyd. Praha: BEN - technická literatura, 1996. ISBN 80-901984-3-0. [↑](#footnote-ref-1)
2. *Https://cs.wikipedia.org/: Operační zesilovač* [online]. [cit. 2018-03-27]. [↑](#footnote-ref-2)