**Vyšší odborná škola a Střední průmyslová škola elektrotechnická Olomouc,**

**Božetěchova3**

**PRAKTICKÁ ZKOUŠKA Z ODBORNÝCH PŘEDMĚTŮ**

Tutoriál na WxPython

2020 Filip Plachý

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

……………………………  
 jméno a příjmení žáka

Chtěl bych vyslovit poděkování panu Ing. Markovi Nožkovi za odborné konzultace a poskytnuté informace.

……………………………  
 jméno a příjmení žáka

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

……………………………  
 jméno a příjmení žáka

# Abstrakt

Cílem diplomové práce je vytvořit sbírku tutoriálů popisující základy knihovny wxWidgets v programovacím jazyce Python. Samotná knihovna pythonu podporující wxWidgets se nazývá wxPython. Jedná se o alternativu k vytváření aplikací s uživatelským grafickým rozhraním.

Sbírka je určena pro studenty programování začínající s GUI, kteří již mají za sebou základní zkušenosti s pythonem. Součástí tutoriálů je úvod do objektově orientovaného programování, které se při tvorbě grafického rozhraní využívá. Po dokončení základních tutoriálů je součástí sbírky i sada názorných příkladů, řešící základní jednoduché aplikace.

# 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 5](#_Toc63866775)

[Úvod 6](#_Toc63866776)

[1. Teoretická část 7](#_Toc63866777)

[1.1 Python 7](#_Toc63866778)

[1.1.1 Úvod do jazyka python 7](#_Toc63866779)

[1.1.2 Historie 7](#_Toc63866780)

[1.2 wxPython 9](#_Toc63866781)

[1.2.1 Historie wxPythonu 10](#_Toc63866782)

[1.3 wxWidgets 10](#_Toc63866783)

[1.3.1 Historie wxWidgets 10](#_Toc63866784)

[1.4 Jupyter Notebook 11](#_Toc63866785)

[1.4.1 Historie Jupyter Notebooku 11](#_Toc63866786)

[1.5 Project Jupyter 12](#_Toc63866787)

[1.6 IPython 12](#_Toc63866788)

[1.7 NbViewer 13](#_Toc63866789)

[1.8 Objektově orientované programování 13](#_Toc63866790)

[1.8.1 Programovací paradigmata 13](#_Toc63866791)

[1.8.2 OOP 15](#_Toc63866792)

[1.8.3 Funkcionalita OOP v Pythonu 16](#_Toc63866793)

[2. Praktická část 17](#_Toc63866794)

[Závěr 18](#_Toc63866795)

[Seznam použité literatury 19](#_Toc63866796)

[Seznam obrázků a tabulek 20](#_Toc63866797)

[Přílohy 21](#_Toc63866798)

# Úvod

Práce se rozděluje na 2 části. Teoretická a praktická. Teoretická obsahuje dopodrobna popsané nástroje a funkce, které byly při tvorbě tutoriálu využity. V praktické části si rozebereme samotný tutoriál knihovny wxPython. Tutoriál je určen pro studenty s již základními zkušenosti v programovacím jazyce python. Při práci s wxPython je potřeba mít i znalost Objektově orientovaného programování (OOP), tudíž součástí sady tutoriálů je i lekce do OOP. Cílem práce je vytvořit představu a znalostní základ, se kterým student zvládne vytvářet aplikace s grafickým uživatelským rozhraním.

# Teoretická část

Teoretická část obsahuje základní informace všech programů a nástrojů, které byly při práci využity.

## Python

Cílem této podkapitoly je obecný úvod do programovacího jazyka Python.

### Úvod do jazyka python

Python je univerzální programovací jazyk[[1]](#footnote-2).

Jazyk paří mezi tzv. "interpretované jazyky". To znamená, že napsaný zdrojový kód v jazyce Python je převeden (interpretován) pomocí programu (interpreter/tlumočník) do jazyka, se kterým pracuje počítač. Samotný tlumočník pro Python je k dispozici na všech operačních systémech přímo na stránkách [www.python.org](https://www.python.org).

Pro práci s pythonem slouží tzv. „IDE“ (Integrated Development Environment) , česky: „editor“, který pomáhá uživateli s formátováním kódu tak, aby byl nejen přehledný pro uživatele, ale také aby se dal převést do formátu proveditelného počítačem. Nejznámějšími editory jsou: Pycharm, Pydev, Visual Studio Code, VIM, Atom/Atom-IDE, IDLE, Spyder...

Python byl navržen tak, aby se jeho veškeré funkce nacházely přímo v jádru programovacího jazyka.

Má jednodušší a méně přeplněnou syntaxi a gramatiku, např. díky využití mezer. Další jeho výhodou je rozšiřitelnost o další knihovny/moduly.

### Historie

Programovací jazyk Python byl navrhnut mezi roky 1990-1991 holaňdanem Guidem van Rossumem[[2]](#footnote-3) v národním výzkumném institutu pro matiku a informatiku (CWI) v Amsterdamu[[3]](#footnote-4). Samotné pojmenování pythonu nemá nic společného s druhem hada, neboť Van Rossum pojmenoval python po televizním pořadu anglické BBC Monty Pythonův létající kruh[[4]](#footnote-5).

První verzi kódu zveřejnil Van Rossum v únoru roku 1991 (verze 0.9.0). Již v této fázi vývoje bylo možné v pythonu pracovat s třídy a dědičnostmi. K dispozici již také byli i základní datové typy jako string, list či dict. V počátečním vydání Python již obsahoval modulový systém[[5]](#footnote-6).

Python dosáhl verze 1 v lednu 1994. Novými funkcemi byla lambda[[6]](#footnote-7) a práce s mapováním[[7]](#footnote-8), filtrováním[[8]](#footnote-9) a redukování[[9]](#footnote-10) vyšších funkcí[[10]](#footnote-11) (Funkce vyšších řádů berou jednu nebo více funkcí nebo atributů a jako výsledek vrací jednu funkci). Později Van Rossum opouští CWI a pokračuje na pythonu ve CNRI[[11]](#footnote-12) (Korporace pro národní výzkumné iniciativy) ve Virginii. Další důležitou verzí je 1.4, kde se objevili pojmenované parametry[[12]](#footnote-13) nebo podpora komplexních čísel.

Během Van Rossumova pobytu ve CNRI se institut snažil o zpřístupnění možnosti programování pro veřejnost se základním vzdělání. Tohle mělo později za následek menší "šachování" mezi licencemi pythonu 1.6, které měli za cíl získání licence svobodného softwaru[[13]](#footnote-14) (Free-software licence). Další verze 1.6.1 sice neobsahovala žádné důležité funkce, ale měla změněnou licenci na GPL/GNU[[14]](#footnote-15) (General Public License neboli bezplatná softwarová licence).

Verze 2.0 byla vydána v říjnu 2000, která představila seznamy[[15]](#footnote-16).

2.2 představila sjednocení typů Pythonu (typů napsaných v C) a tříd (typů napsaných v Pythonu) do jedné hierarchie.

2.5 představil prohlášení with[[16]](#footnote-17) umožňující otevření a zavření souboru a další funkce.

Python 2.6 byl vydán, aby se shodoval se souběžným vývojem Pythonu 3.0 a varoval hlavně o funkcích, které jsou ve verzi 3.0 odstraněny.

Podobně byl vydán i 2.7, který se shodoval s 3.1. Python 2.7 byl posledním vydáním ve druhé sérii. V listopadu 2014 byl oznámen konec podpory 2.7 do roku 2020. Uživatelé byli vyzváni, aby postupně přešli na Python 3.0.

1. ledna 2020 byl "zmražen"[[17]](#footnote-18) kód Pythonu 2.7. Konečné vydání, 2.7.18, došlo 20. dubna 2020 a zahrnovalo opravy kritických chyb a blokace vydání.

Verze 3.0 přišla 3. prosince 2008. 3.0 byla navržena, aby napravila základní konstrukční chyby jazyka. Tyhle změny avšak znemožnili zpětnou kompabilitu se staršími verzemi (tudíž došlo k samotnému oddělení verzí z 2.x na 3.0). Hlavním motem změn bylo odstranění nadbytečných, duplicitních konstrukcí a modulů. Vznikl nástroj tzv. 2to3, který dokázal přepsat automaticky Python 2 do nové verze, avšak nástroj nefunguje na 100% a některé aspekty nedokáže převést.

Hlavními úpravami byly:

* Změna print, aby se jednalo o vestavěnou funkci. V Pythonu 2.6 a 2.7 print() je k dispozici jako vestavěná funkce, avšak je maskovaná syntaxí příkazu, který lze deaktivovat zadáním „from \_\_future import print\_functionv“ v hlavičce souboru
* Odebrání inputu ve verzi 2. Z verze poté byl přebrat raw\_input, který byl přejmenován na klasický input.
* Přidání podpory pro anotace jednotlivých funkcí (Když v kódu byla nepoužívaná funkce se špatnou syntaxí, tak se program ve 2.x nespustil)
* Sjednocení str / unicode typů
* Odebrání funkcí zpětné kompatibility, včetně tříd starého stylu, výjimek řetězců a implicitních relativních importů
* Změna funkce celočíselného dělení. (ve 2.0 5 / 2 = 2, nyní 5 / 2 = 2.5. ve 3.0 vznikla náhradní syntaxe 5 // 2 = 2)

## wxPython

WxPython[[18]](#footnote-19) je obal pro multiplatformní GUI[[19]](#footnote-20) (grafické uživatelské rozhraní) aplikační programovací rozhraní wxWidgets (napsán v C++) pro programovací jazyk Python. Jedná se o otevřený (veřejný kód) rozšiřující modul Pythonu. Oficiální stránky wxPythonu: https://www.wxpython.org/

### Historie wxPythonu

WxPython byl vytvořen Robinem Dunnem[[20]](#footnote-21), když potřeboval GUI k operačnímu systému HP-UX a Windows verze 3.1 (1992-1995). Při hodnocení komečních řešení narazil na vazby Pythonu se sadou nástrojů wxWidgets.

První verze verze byly vytvořeny ručně. Avšak brzy se kódová základna velmi obtížně udržovala synchronizovaná s novými verzemi wxWidgets. Pozdější verze byly vytvořeny pomocí SWIG [[21]](#footnote-22)(nástroj pro propojení knihoven v C se skriptovacími jazyky), který výrazně snížil množství práce na aktualizaci.

První "moderní" verze 0.3 byla oznámena v roce 1998.

Aktuální verze s podtitulem „Excaping the Quarantine“ je 4.1.0.

Celý vývoj wxPython nalezneme na oficiálních stránkách:

<https://wxpython.org/pages/changes/index.html>

## wxWidgets

Samotná sada nástrojů wxWidgets je knihovna nástrojů pro vytváření graficky uživatelských rozhraní napříč všemi platformami. WxWidgets umožňuje kódu GUI kompilovat a spouštět na několika počítačových platformách s žádnými nebo minimálními změnami kódu.

Jedná se o bezplatný a otevřený software distribuovaný za podmínek licence WxWidgets[[22]](#footnote-23), která je obdobná GPL/GNU u Pythonu. Stránky wxWidgets: <https://www.wxwidgets.org/>

### Historie wxWidgets

WxWidgets (původně wxWindows) zahájil v roce 1992 Julian Smart[[23]](#footnote-24) z Edinburské Univerzity[[24]](#footnote-25). V roce 2004 došlo k přejmenování wxWindows v důsledku požadavků společnosti Microsoft pro distribuci v UK.

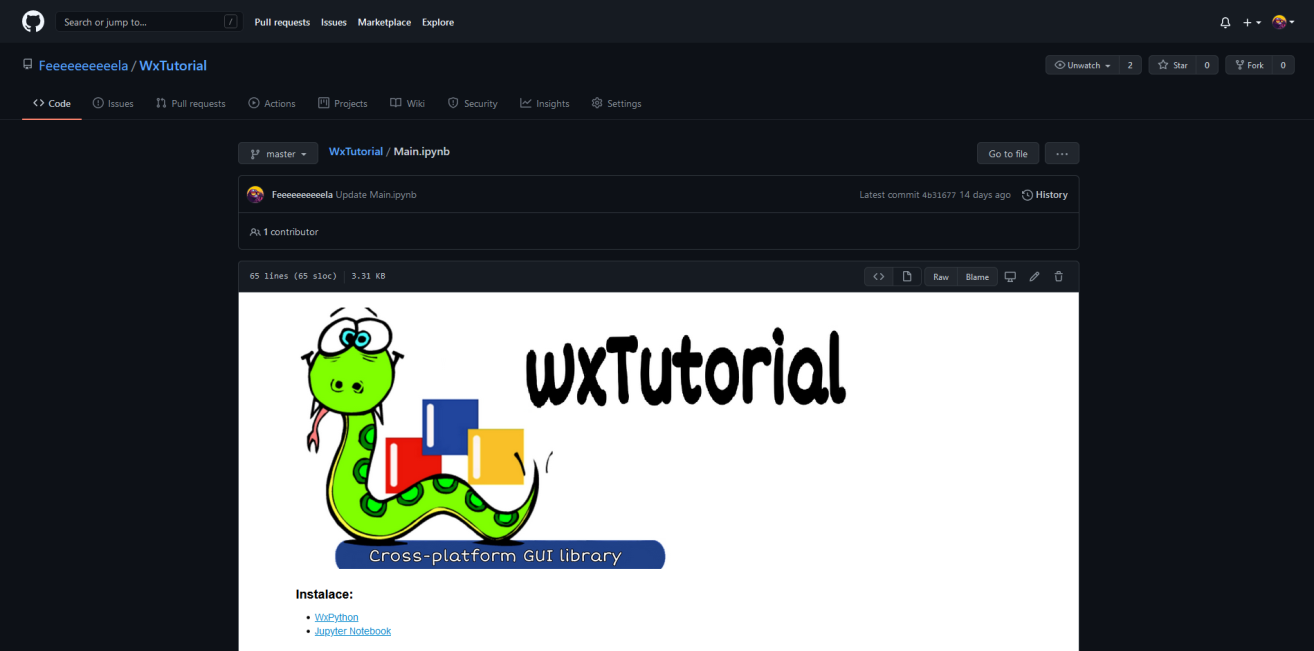
Hlavní verze byla vydána 6. ledna 2004. Samotná verze 3.0 byla vydána 11. listopadu 2013.

## Jupyter Notebook

Jedná se o webové interaktnivní prostředí výpočetní prostředí pro vytváření stejnojmeného dokumentu, který je součástí Projektu Jupyter. Samotný notebook je kombinace Markdown[[25]](#footnote-26) dokumentu formátovaným do JSON[[26]](#footnote-27) (formát slovníku). Zápis do notebooku je prováděn pomocí tzv "buněk", které mohou obsahovat funkční kód mnoha jazyků[[27]](#footnote-28), text, matematiku, grafy, obrázky. Dokumenty mají koncovku "ipynb".

### Historie Jupyter Notebooku

V roce 2014 vznikl Projekt Jupyter oddělením od IPythonu. IPython nadále existuje jako prostředí Pythonu, zatímco Notebook a další jazykově nezávislé části IPythonu se přesunuly pod názvem Jupyter.

V roce 2015 GitHub[[28]](#footnote-29) a projekt Jupyter oznámili nativní vykreslování formátu souborů notebooků Jupyter (soubory .ipynb) na platformě GitHub.

Obrázek č. 1 - Zobrazení souborů .ipynb na GitHubu

Zdroj: Můj osobní GitHub

## Project Jupyter

Je nezisková organizace vytvořená s motem "rozvinout otevřený software a služby pro interaktnivní práci na počítači přes desitký programovacích jazyků". Jupyter vznikl oddělením od IPythonu v roce 2014, za kterým stál Fernando Pérez[[29]](#footnote-30). Pod Projekt Jupyter spadají interaktivní výpočetní produkty Jupyter Notebook, JupyterHub a JupyterLab.

## IPython

IPython (Interactive Python) je příkazový shell[[30]](#footnote-31) (příkazový řádek pro práci s počítačem) pro interaktnivní výpočty ve více programovacích jazycích (původně pouze pro Python). Podporuje práci s multimédii[[31]](#footnote-32), introspekci[[32]](#footnote-33) (schopnost programu zkoumat typ a vlastnosti objektu při běhu), nástroje pro paralerní výpočty[[33]](#footnote-34) (výpočet komplikovaných příkladů pomocí rozdělení na menší díly, které se vyřeší jednodušeji) a rozhraní poznámkového bloku[[34]](#footnote-35) s podporou kódu, textu, matematických výrazů a dalších médií (podobně jako Markdown).

## NbViewer

Jedná se o nový produkt Jupyter Notebooku, který vytváří rozhraní, které umí vzít veřejné url[[35]](#footnote-36) notebook dokumentu a převede ho na formát html[[36]](#footnote-37), který zobrazí jako webovou stránku.

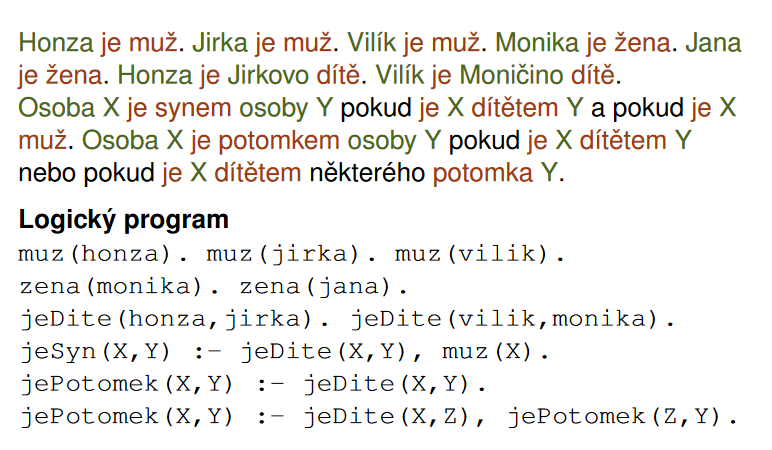
## Elektromagnet ventilObjektově orientované programování

Obrázek č. 2 - webové rozhraní NbViewer

Zdroj: <https://nbviewer.jupyter.org/>

### Programovací paradigmata

OOP se jedná se o tzv. programovací paradigma[[37]](#footnote-38). Jedná se o způsob jak rozdělit programovací jazyky. Paradigma se zabývají implikacemi (prováděním) provádění jazyka (samotné psaní kódu). Paradigmata se rozdělují na:

1. Imperativy[[38]](#footnote-39) - uživatel upravuje/instruuje samotné zařízení/stroj, tím mění jeho stav
   * Procedurální programování[[39]](#footnote-40) (Strukturované programování): nejzákladnější/nejjednodušší paradigma, které seskupuje pokyny do postupů (jednoduše obsahuje řadu výpočtových kroků, které vedou k cílenému výsledku).
   * Objektově orientované programování[[40]](#footnote-41) (OOP): paradigma je založené na konceptu tzv. "objektů" (viz později).
2. Deklarativní[[41]](#footnote-42) - uživatel pouze upravuje vlastnosti požadovaného výsledku, ale ne způsob jeho výpočtu
   * Funkcionální programování[[42]](#footnote-43): programy jsou konstruovány funkcemi, které se skládají do stromů či samotných výrazů, které vracejí hodnotu podle které se mění stav programu.
   * Logické programování[[43]](#footnote-44): je psán sadami vět v logické podobě, které řeší programové problémy.

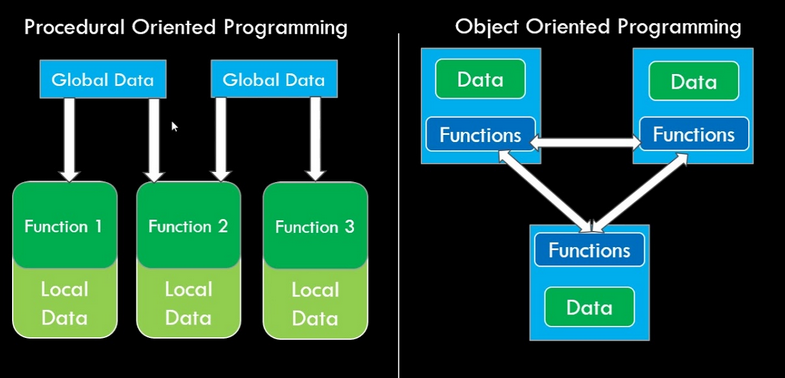
Obrázek č. 3 - Příklad logického programování

Zdroj: prezentace Logické programování - Tomáš Kühr: <https://www.inf.upol.cz/downloads/ruzne/LPproSS.pdf>

* + Matematické programování[[44]](#footnote-45): využívá mnohé způsoby matematických funkcí (lineární, celočíselné, kvadratické, dynamické programování...)

### OOP

OOP paradigma tedy funguje na principů objektů - samotné objekty obsahují buď data[[45]](#footnote-46) (atributy, vlastnosti) nebo funkce[[46]](#footnote-47) (metody).

Funkce objektů spočívá v tom, že objekty mohou přistupovat a upravovat datová pole podle sebe sama (podle jednoho atributu se upraví atribut z jiného objektu). Tudíž objekty mezi sebou interagují.

Obrázek č. 4 - Rozdíl mezi procedurálním „klasickým“ a objektově orientovaným.

Zdroj: obrázek z videa Simple Snippets o OOP v Javě

Odkaz: <https://www.youtube.com/watch?v=Jt13s8RYAVg>

Jazyků OOP je mnoho. Mezi nejoblíbenější však patří ty, které jsou založeny na třídách[[47]](#footnote-48). Objekty jsou poté vytvářeny do tříd[[48]](#footnote-49), kde se stávají tzv. jejich instancemi[[49]](#footnote-50) (rodiči), které určují jejich datové typy.

Mnoho nejrozšířenějších programovacích jazyků (např. C++, Python, Java atd.) jsou založeny na více paradigmech a podporují OOP ve větší či menší míře.

y\Mezi významné objektově orientované jazyky patří: Java, C++, C#, Python, R, PHP(pro weby), Visual Basic(microsoft), JavaScript(pro weby), Ruby, Pearl, Object Pascal, Objective-C, Dart, Swift, Scala, Kotlin, Common Lisp, MATLAB atd…

### Funkcionalita OOP v Pythonu

V Pythonu se využívá tzv. dědičnost[[50]](#footnote-51) pro opětovné použití kódu a rozšířenost ve formě tříd. V konceptu:

* Třídy - definice datového formátu a dostupné postupy pro daný typ objektu nebo třídy objektu
* Objekt - instance tříd (objekty dědí vlastnosti dané třídy)

Funkce v OOP jsou známe jako metody

Proměnné jako pole[[51]](#footnote-52), členy, atributy nebo vlastnosti.

Existují tedy:

* Proměnné třídy - data patří do třídy jako celku
* Proměnné/atributy instance - data patří pouze jednotlivým objektům
* Členské proměnné[[52]](#footnote-53) - označujeme tak proměnné třídy nebo instance, které jsou definovány konkrétní třídou
* Metody třídy - patří do třídy jako celku a mají přístup pouze k proměnným a volaným vstupům třídy
* Metody instance - patří k jednotlivým objektům, přístup pouze k proměnným konkrétního objektu, na který jsou volány vstupy a proměnné třídy

# Praktická část

Praktická část popisuje výhody využití a aplikaci zmíněných nástrojů v teoretické části a samotnou tvorbu tutoriálu.

## Prostředí Pythonu

Při tvorbě tutoriálu byly využity stabilní verze Pythonu 3.8.6 [[53]](#footnote-54)a 3.7.9[[54]](#footnote-55).

Nejnovější verze 3.9.0 [[55]](#footnote-56)není zatím kompatibilní s wxPythonem (rok 2020).

Při studování knihovny wx jsem pracoval ve dvou prostředích a to hlavně v Atomu[[56]](#footnote-57) a příležitostně ve Visual Studio Code[[57]](#footnote-58) (VSCode)

# 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] Wikipedie:

<https://cs.wikipedia.org/wiki/Python>

<https://cs.wikipedia.org/wiki/Programovac%C3%AD_jazyk>

https://en.wikipedia.org/wiki/Guido\_van\_Rossum

# 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 - Zobrazení souborů .ipynb na GitHubu 11](file:///D:\aŠKOLA\maturitní%20práce\Dokumentace\DPM.docx#_Toc63710600)

[Obrázek č. 2 - Rozhraní NbViewer 13](file:///D:\aŠKOLA\maturitní%20práce\Dokumentace\DPM.docx#_Toc63710601)

[Tabulka č. 1: Měření zesílení operačního zesilovače 3](#_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. https://cs.wikipedia.org/wiki/Programovac%C3%AD\_jazyk [↑](#footnote-ref-2)
2. https://en.wikipedia.org/wiki/Guido\_van\_Rossum [↑](#footnote-ref-3)
3. https://en.wikipedia.org/wiki/Centrum\_Wiskunde\_%26\_Informatica [↑](#footnote-ref-4)
4. https://en.wikipedia.org/wiki/Monty\_Python%27s\_Flying\_Circus [↑](#footnote-ref-5)
5. https://en.wikipedia.org/wiki/Modular\_programming [↑](#footnote-ref-6)
6. https://www.w3schools.com/python/python\_lambda.asp [↑](#footnote-ref-7)
7. https://en.wikipedia.org/wiki/Map\_%28higher-order\_function%29 [↑](#footnote-ref-8)
8. https://en.wikipedia.org/wiki/Filter\_%28higher-order\_function%29 [↑](#footnote-ref-9)
9. https://en.wikipedia.org/wiki/Fold\_%28higher-order\_function%29 [↑](#footnote-ref-10)
10. https://en.wikipedia.org/wiki/Higher-order\_function [↑](#footnote-ref-11)
11. https://en.wikipedia.org/wiki/Corporation\_for\_National\_Research\_Initiatives [↑](#footnote-ref-12)
12. https://en.wikipedia.org/wiki/Named\_parameter [↑](#footnote-ref-13)
13. https://cs.qaz.wiki/wiki/Free-software\_license [↑](#footnote-ref-14)
14. https://en.wikipedia.org/wiki/GNU\_General\_Public\_License [↑](#footnote-ref-15)
15. https://en.wikipedia.org/wiki/List\_comprehension [↑](#footnote-ref-16)
16. https://en.wikipedia.org/wiki/Python\_syntax\_and\_semantics#With\_statements [↑](#footnote-ref-17)
17. https://en.wikipedia.org/wiki/Freeze\_%28software\_engineering%29 [↑](#footnote-ref-18)
18. https://en.wikipedia.org/wiki/WxPython [↑](#footnote-ref-19)
19. https://en.wikipedia.org/wiki/Graphical\_user\_interface [↑](#footnote-ref-20)
20. https://wiki.wxpython.org/RobinDunn [↑](#footnote-ref-21)
21. https://cs.qaz.wiki/wiki/SWIG [↑](#footnote-ref-22)
22. https://en.wikipedia.org/wiki/WxWidgets#License [↑](#footnote-ref-23)
23. http://www.anthemion.co.uk/julian.htm [↑](#footnote-ref-24)
24. https://en.wikipedia.org/wiki/University\_of\_Edinburgh [↑](#footnote-ref-25)
25. https://en.wikipedia.org/wiki/Markdown [↑](#footnote-ref-26)
26. https://en.wikipedia.org/wiki/JSON [↑](#footnote-ref-27)
27. https://jupyter4edu.github.io/jupyter-edu-book/jupyter.html [↑](#footnote-ref-28)
28. https://en.wikipedia.org/wiki/GitHub [↑](#footnote-ref-29)
29. https://en.wikipedia.org/wiki/Fernando\_P%C3%A9rez\_%28software\_developer%29 [↑](#footnote-ref-30)
30. https://en.wikipedia.org/wiki/Shell\_%28computing%29 [↑](#footnote-ref-31)
31. https://en.wikipedia.org/wiki/Interactive\_media [↑](#footnote-ref-32)
32. https://en.wikipedia.org/wiki/Type\_introspection [↑](#footnote-ref-33)
33. https://en.wikipedia.org/wiki/Parallel\_computing [↑](#footnote-ref-34)
34. https://en.wikipedia.org/wiki/Notebook\_interface [↑](#footnote-ref-35)
35. https://en.wikipedia.org/wiki/URL [↑](#footnote-ref-36)
36. https://en.wikipedia.org/wiki/HTML [↑](#footnote-ref-37)
37. https://en.wikipedia.org/wiki/Programming\_paradigm [↑](#footnote-ref-38)
38. https://en.wikipedia.org/wiki/Imperative\_programming [↑](#footnote-ref-39)
39. https://en.wikipedia.org/wiki/Procedural\_programming [↑](#footnote-ref-40)
40. https://en.wikipedia.org/wiki/Object-oriented\_programming [↑](#footnote-ref-41)
41. https://en.wikipedia.org/wiki/Declarative\_programming [↑](#footnote-ref-42)
42. https://en.wikipedia.org/wiki/Functional\_programming [↑](#footnote-ref-43)
43. https://en.wikipedia.org/wiki/Logic\_programming [↑](#footnote-ref-44)
44. https://en.wikipedia.org/wiki/Mathematical\_optimization [↑](#footnote-ref-45)
45. https://en.wikipedia.org/wiki/Data\_type [↑](#footnote-ref-46)
46. https://en.wikipedia.org/wiki/Method\_%28computer\_programming%29 [↑](#footnote-ref-47)
47. https://en.wikipedia.org/wiki/Class-based\_programming [↑](#footnote-ref-48)
48. https://en.wikipedia.org/wiki/Class\_%28computer\_programming%29 [↑](#footnote-ref-49)
49. https://en.wikipedia.org/wiki/Instance\_%28computer\_science%29 [↑](#footnote-ref-50)
50. https://en.wikipedia.org/wiki/Inheritance\_%28object-oriented\_programming%29 [↑](#footnote-ref-51)
51. https://en.wikipedia.org/wiki/Field\_%28computer\_science%29 [↑](#footnote-ref-52)
52. https://en.wikipedia.org/wiki/Member\_variable [↑](#footnote-ref-53)
53. https://www.python.org/downloads/release/python-386/ [↑](#footnote-ref-54)
54. https://www.python.org/downloads/release/python-379/ [↑](#footnote-ref-55)
55. https://www.python.org/downloads/release/python-390/ [↑](#footnote-ref-56)
56. https://atom.io/ [↑](#footnote-ref-57)
57. https://code.visualstudio.com/ [↑](#footnote-ref-58)