Programko.sk

Python 3

Polia a ich funkcie

Pripomeňte si prácu s poľami a napíšte nasledujúce programy. Na pripomenutie si prečítajte prehľad funkcií a syntaxe pre prácu s poľami.

- Vytvorenie prázdneho poľa: pole = []
- Prístup k indexu i v poli: pole[i] – Pozor. Ak je index i mimo rozsah poľa, tak dostaneme chybu.
- Pridanie čísla 5 do poľa: pole.append(5)
- Odstránenie prvku s indexom 3 z poľa: pole.pop(3) – Pozor. Ak sa pokúsime odstrániť index, ktorý je mimo rozsah poľa, tak dostaneme chybu.
- Odstránenie posledného prvku z poľa: pole.pop() – Pozor. Ak sa pokúsime odstrániť prvok z prázdneho poľa, tak dostaneme chybu.
- Výpočet dĺžky poľa: len(pole)
- Súčet prvkov v poli: sum(pole)
- Výpočet maximálnej/minimálnej hodnoty v poli: max(pole)/min(pole) – Pozor. Pole z ktorého počítame maximum a minimum nesmie byť prázdne, inak dostaneme chybu.

1.1 Úloha

Vygenerujte náhodné celé číslo x (medzi 1 a 20). Následne pomocou cyklu vytvorte pole, ktoré bude obsahovať x náhodných čísel (každé číslo medzi 1 a 100).

Hint – aby sme vedeli pracovať s náhodnými číslami nesmieme zabudnúť importovať knižnicu random.

1.2 Úloha

Vypíšte dĺžku poľa, ktoré ste vytvorili v predošlom príklade. Následne vypíšte súčet všetkých prvkov vo vašom poli. Na záver vygenerujte náhodný index z vami vytvoreného poľa a odstráňte prvok na danom indexe. Zamyslite sa aké indexy môžu byť v danom poli a podľa toho nastavte hranice do generovacej funkcie.

1.3 Úloha

Zistite hodnoty maximálneho a minimálneho prvku vo vašom poli a odčítajte ich od seba (maximum - minimum). Ak je výsledná hodnota väčšia, nanajvýš rovná ako dĺžka poľa, pridajte túto hodnotu do poľa. V opačnom prípade odstráňte z poľa prvok s indexom rovnajúcim sa vypočítanej hodnote.

1.4 Úloha

Majme nasledujúce **pole = [1, 3, 3, 41, 5, 6, 10, 13, 27, 0, -7, 9]**. Vypíšte z poľa indexy na ktorých sa nachádzajú maximálny a minimálny prvok.
Hint – musíme použiť kombináciu funkcie na výpočet maxima/minima a funkcie index. Funkcia index funguje tak, že vezme nejaké pole a prvok, ktorý sa v ňom nachádza a vráti jeho prvý výskyt (prvý index). Napríklad **pole.index(41)** vráti index 3 a **pole.index(3)** vráti index 1.

1.5 Úloha

Vytvorte dve ľubovoľné polia a potom tieto dve polia spojte do jedného poľa.
Ak je dĺžka výsledného poľa väčšia ako 7 vytlačte z poľa prvé 3 prvky. V opačnom prípade vytlačte z poľa prvú polovicu prvkov.

1.6 Úloha

Majme nasledujúce **pole = [1, 3, 3, 41, 5, 6, 10, 13, 27, 0, -7, 9]**. Vypíšte dané pole pomocou cyklu. Pole najprv vypíšte pomocou for cyklu a potom pomocou while cyklu.

1.7 Úloha

Z poľa z predošlého príkladu odstráňte n prvkov. Číslo n vygenerujte ako náhodné číslo. Uvedomte si aké hranice je potrebné vložiť do funkcie generujúcej n. Prvky odstraňujte buď pomocou for alebo while cyklu.

1.8 Úloha

Majme nasledujúce **pole = [1, 3, 5, 6, 10, 27, 0, 9]**. Postupne doplňte do poľa chýbajúce prvky tak, aby po doplnení pole vyzeralo nasledovne - [1, 3, 3, 41, 5, 6, 10, 13, 27, 0, -7, 9]

Hint – funkcia **append()** nie je správna voľba, pretože pridáva prvky na koniec poľa. Musíme použiť funkciu **insert()**, ktorá potrebuje index kam chceme prvok pridať a samotný prvok ktorý sa pridáva. Ak funkciu zavoláme na kratšie pole zo zadania tak po volaní **pole.insert(4, 13)** bude pole vyzerať nasledovne - [1, 3, 5, 6, 10, 13, 27, 0, 9]. Teda sme pridali číslo 13 za index 4 (prvok s hodnotou 10)

Dvojrozmerné a viacrozmerné polia

2D pole, inak povedané dvojrozmerné alebo dvojdimenzionálne pole je vlastne pole v nejakom inom poli. Je to teda pole polí. V takomto poli pristupujeme k uloženým dátovým hodnotám pomocou dvoch indexov miesto jedného. Prestavte si to ako tabuľku nejakých dát, kde prvý index reprezentuje riadok a druhý index reprezentuje stĺpec. Ako príklad si uveďme nasledovné pole:

pole = [[2, 3, 8, 4], [-3, 0, 5], [7, 6, -10, 7], [1, 0]]

Všimnite si, že toto pole v sebe obsahuje 4 ďalšie polia, ktoré obsahujú rôzne (ale kľudne aj rovnaké) počty prvkov. Prvky poľa vieme indexovať pomocou prvého indexu. Prvky z jednotlivých polí vieme indexovať pomocou druhého indexu.

- Ak chceme vytlačiť číslo -3 z druhého poľa musíme použiť príkaz **print(pole[1][0])**.
- Ak chceme vytlačiť číslo 8 z prvého poľa musíme použiť príkaz **print(pole[0][2])**.
- Ak použijeme príkaz print(pole[4][hocijaký_index]) dostaneme chybu, pretože naše pole nemá uložené žiadne pole na indexe 4. V tomto prípade je jedno aký index dáme na druhú pozíciu.
- Ak použijeme príkaz print(pole[3][2]) dostaneme chybu, pretože posledné pole [1, 0] v našom poli nemá index 2. Toto pole má len dva prvky, takže má iba indexy 0 a 1.
- Ak chceme vypočítať a vytlačiť dĺžku poľa, tak použijeme print(len(pole)).
- Ak chceme vypočítať a vytlačiť dĺžku tretieho poľa v našom poli, tak použijeme print(len(pole[2])).
- Podobne sa dajú aplikovať aj všetky ostatné funkcie pre prácu s poľami.

2.1 Úloha

Majme **pole = [[2, 3, 8, 4], [-3, 0, 5], [7, 6, -10, 7], [1, 0]]**. Pomocou for cyklu vytlačte z tohto poľa všetky polia ktoré sú v ňom uložené. Netlačte prvky polí, ale samotné polia.

2.2 Úloha

Majme **pole = [[2, 3, 8, 4], [-3, 0, 5], [7, 6, -10, 7], [1, 0]]**. Pomocou for cyklu vytlačte postupne prvky všetkých polí vnútri tohto poľa.

Hint – Musíme použiť vnorené for cykly s dvomi rôznymi premennými (napr. i a j). Potom vieme indexovať ako **pole[i][j]**.

2.3 Úloha

Vyberte si z poľa z predošlého príkladu ľubovoľné pole a pridajte do neho prvok. Potom si vyberte nejaké iné z polí a odstráňte z neho prvok.

2.4 Úloha

Majme nejaké pole ktoré obsahuje nejaké názvy krajín, celé čísla, nejaké mená a farby. Každé v samostatnom poli. Takéto pole môže vyzerať napríklad nasledovne.
pole = [[“Slovensko”, “Česko”, “Anglicko”, “Nemecko”, “Kanada”, “Japonsko“], [4, -3, 8, 9, 11, 7], [“Matej“, “Adam“, “Anna“, “Ján“, “Marcela“], [“zelená”, “modrá”, “červená”]].

Z poľa vypíšte celé pole, ktoré obsahuje farby, potom vypíšte súčet prvkov poľa čísel, ďalej vypíšte každý druhý štát a na záver vypíšte počet mien ktoré sú uložené v poli mien.

2.5 Úloha

Spojte polia z úloh 2.1. a 2.4. tak aby vzniklo 3D (trojrozmerné) pole. Následne diskutujte ako by ste z takéhoto poľa vytlačili všetky prvky na najhlbšej úrovni (samotné prvky najhlbších polí).

Slovníky a ich funkcie

Slovník je asociatívny zoznam (alebo mapa) a mapuje kľúče na hodnoty. Je podobný zoznamu, ale jeho prvky nie sú indexované pomocou postupnosti celých čísel, ale pomocou kľúčov. Kľúčom môže byť napríklad číslo, reťazec alebo iný nemeniteľný objekt.

Napríklad zoznam kľúčom byť nemôže, pretože je to meniteľný objekt. Pod kľúčom môžeme pristupovať k uloženým hodnotám, ak sú v slovníku prítomné.

Pripomeňte si príkazy ktoré viete aplikovať na slovníky.

3.1 Úloha

Majme nasledujúci slovník **points = {“Adam“ : 34, “Paul“ : 20, “Anna“ : 45}**. Do slovníka pridajte ďalšieho študenta s ľubovoľným počtom bodov. Následne slovník points vytlačte. Hint – Treba použiť zápis na pridanie nového kľúča **points[key] = value**, kde key je kľúč ktorý pridávate a value je hodnota ktorú priraďujete danému kľúču.

3.2 Úloha

Zo slovníka vypíšte ľubovoľnú hodnotu. Musíme použiť zápis podobný výpisu hodnoty z poľa, ale miesto indexu je treba uviesť správny kľúč.

3.3 Úloha

Pomocou for cyklu vypíšte všetky kľúče zo slovníka. Následne pomocou ďalšieho for cyklu vypíšte všetky hodnoty zo slovníka. Množinu kľúčov slovníka points vieme získať ako **points.keys()**, množinu hodnôt ako **points.values()**.

Ak chceme zo slovníka odstrániť nejaký prvok musíme použiť príkaz dictionary.pop(key), kde dictionary je slovník z ktorého odstraňujeme a key je kľúč ktorý chceme zo slovníka odstrániť (odstráni sa dvojica kľúč : hodnota_na_danom_kľúči).

Alternatívne vieme použiť príkaz dictionary.popitem() ktorý odstráni poslednú pridanú dvojicu kľúč : hodnota.

Ak chceme v slovníku nejakú hodnotu na nejakom kľúči zmeniť, musíme použiť zápis **dictionary[key] = new_value**. Poprípade ak je hodnota číslo a chceme ju zväčšiť o 1 tak použijeme **dictionary[key] += 1**.

3.4 Úloha

Vytvorte program ktorý si od uživateľa postupne vypýta 10 slov. Vytvorte slovník ktorý bude obsahovať štatistiku všetkých písmen ktoré sa v zadaných slovách nachádzajú. Ako príklad si uveďme dve slová “ahoj“ a “jano“. Štatistika z týchto dvoch slov je uložená v slovníku
statistics = {“a“ : 2, “h“ : 1, “o“ : 2, “j“ : 2, “n“ : 1} pretože presne toľko daných písmen sa v týchto dvoch slovách dokopy nachádza.

3.5 Úloha

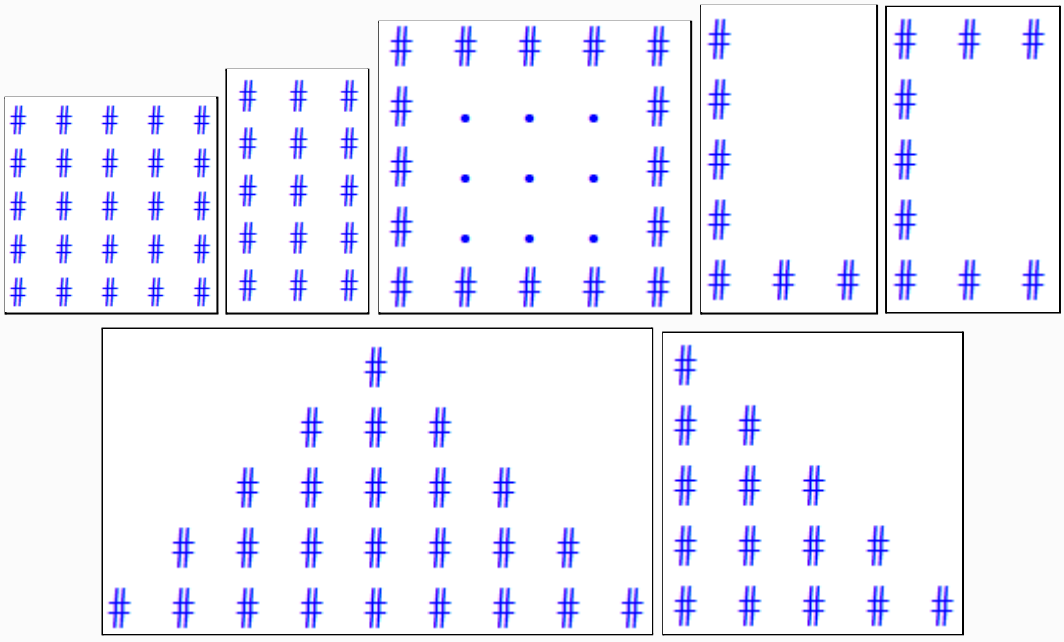
Vytvorte program, ktorý bude pracovať so slovníkom
points = {“Adam“ : 28, “Mary“ : 30, “John“ : 21, “Paul“ : 15, “Anna“ : 24, “Tina“ : 18, “David“ : 12, “Max“ : 25}
Tento slovník obsahuje mená a počty bodov študentov kurzu informatiky za školský rok. Vašou úlohou bude vytvoriť nový slovník grades, ktorý vypočíta koľko ľudí dostane známky A, B, C, D, E, F.
Stupnica pre jednotlivé známky je **A: 30 – 26 | B: 25 – 21 | C: 20 – 16 | D: 15 – 11 | E: 10 – 6 | F: 5 - 0**

3.6 Úloha

Vytvorte program ktorý bude pracovať so slovníkom points z predošlého cvičenia. Vypočítajte koľko bodov dostali študenti kurzu informatiky dohromady.

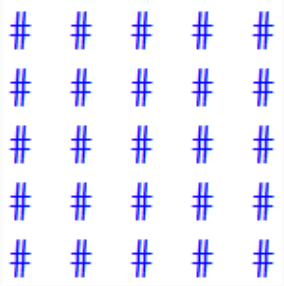
Grafika ASCII art

V programovaní vieme efektívne využiť výstup konzole na vykreslenie viacerých geometrických obrazcov. Tieto môžu byť napr. štvorec, obdĺžnik, trojuholník, kosoštvorec atď... Obrazce sú vo všeobecnosti tvorené zo znakov ASCII (písmená, čísla, špeciálne znaky, ...). Aby sme takéto obrazce mohli vykresliť na výstup musíme vhodne použiť príkaz print() a for cyklus.
Príklady ASCII art:



4.1 Úloha

Napíšte funkciu stvorec(a), ktorá ako parameter dostáva jedno číslo a, ktoré reprezentuje dĺžku strany štvorca. Funkcia stvorec teda vykreslí štvorec o dĺžke strany a. Výstup pre volanie stvorec(5) vyzerá nasledovne:



4.2 Úloha

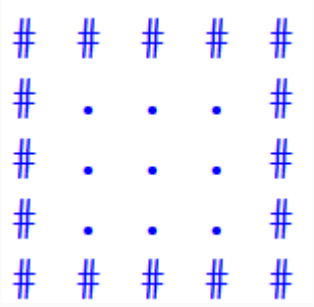
Napíšte funkciu obdlznik(a, b), ktorá ako parametre dostáva dve čísla a a b, ktoré reprezentujú dĺžky strán obdĺžnika. Funkcia obdlznik teda vykreslí obdĺžnik o dĺže strán a a b. Výstup pre volanie obdlznik(5, 3) vyzerá napríklad nasledovne:

4.3 Úloha

Upravte jednu z predošlých dvoch funkcií tak, aby funkcia brala navyše ešte jeden parameter znak, ktorým sa určí z akých znakov bude daný obrazec vytvorený. Príklad volania by mohlo byť volanie obdlznik(5, 3, “*”) pre ktoré výstup vyzerá napríklad nasledovne:

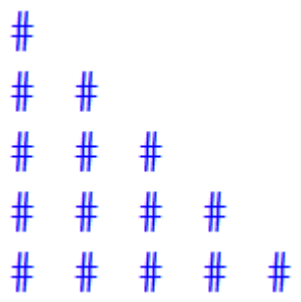
4.4 Úloha

Napíšte funkciu prazdnyStvorec(a), ktorá vypíše štvorec, ktorý je vo vnútri prázdny a zvonka je ohraničený okrajom. Výstup pre volanie prazdnyStvorec(5) vyzerá nasledovne:



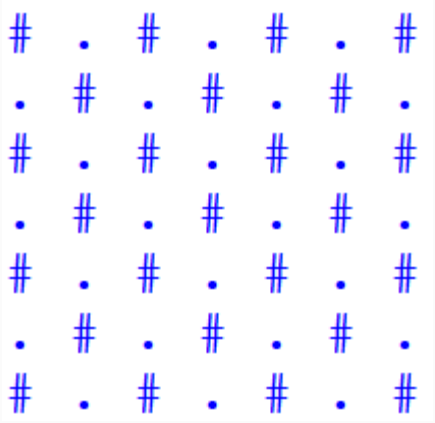
4.5 Úloha

Napíšte funkciu trojuholnik(a), ktorá dostane jeden parameter a, ktorý reprezentuje dĺžku základne pravouhlého trojuholníka. Funkcia trojuholnik teda vykreslí pravouhlý trojuholník s danou dĺžkou základne. Výstup pre volanie trojuholnik(5) vyzerá nasledovne:



BONUS (nepovinné/domáca úloha):

Napíšte funkciu sachovnica(a), ktorá pre dané a vypíše šachovnicu o veľkosti a * a. Výstup pre volanie sachovnica(7) bude vyzerať napríklad nasledovne:



Pokračujeme s ASCII Art

5.1 Úloha

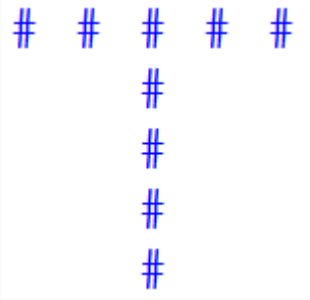
Napíšte funkciu pismenoL(n), ktorá pre dané číslo n vypíše písmeno L, ktorého výška je n. Dĺžku spodnej časti L si vo funkcií vypočítajte podľa vlastného uváženia. Výstup pre volanie pismenoL(5) môže vyzerať napríklad nasledovne:

5.2 Úloha

Napíšte funkciu pismenoC(n), ktorá pre dané číslo n vypíše písmeno C, ktorého výška je n. Dĺžku hornej a spodnej časti C si vo funkcií vypočítajte podľa vlastného uváženia (najlepšie tak ako v predošlej úlohe). Výstup pre volanie pismenoC(5) môže vyzerať napríklad nasledovne:

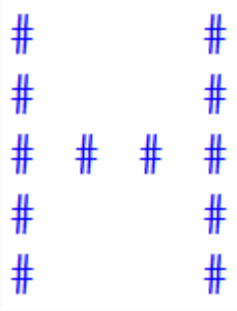
5.3 Úloha

Napíšte funkciu pismenoT(n), ktorá pre dané číslo n vypíše písmeno T, ktorého šírka je n. Dĺžku hornej a časti T si vo funkcií vypočítajte podľa vlastného uváženia (najlepšie tak ako v predošlej úlohe). Pre túto úlohu skontrolujte, či je číslo n nepárne a ak je párne, tak vypíšte správu, že písmeno T sa dá vytlačiť iba pre nepárne n. Výstup pre volanie pismenoT(5) môže vyzerať napríklad nasledovne:



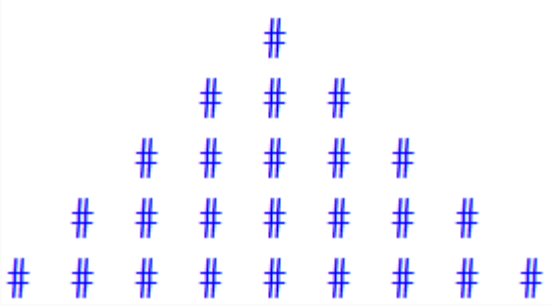
5.4 Úloha

Napíšte funkciu pismenoH(n), ktorá pre dané číslo n vypíše písmeno H, ktorého výška je n. Dĺžku prostrednej časti H si vo funkcií vypočítajte podľa vlastného uváženia (najlepšie tak ako v predošlej úlohe). Pre túto úlohu skontrolujte, či je číslo n nepárne a ak je párne, tak vypíšte správu, že písmeno H sa dá vytlačiť iba pre nepárne n. Výstup pre volanie pismenoH(5) môže vyzerať napríklad nasledovne:



5.5. (BONUS)

Napíšte funkciu pyramida(n), ktorá pre dané číslo n vypíše pyramídu s výškou presne n. Pozor, táto funkcia nie je taká istá ako funkcia trojuholník z predošlej úlohy. Výstup pre volanie pyramida(5) vyzerá nasledovne:



Úvod do knižnice tkinter

Doteraz sme na hodinách pracovali s klasickým príkazom print(), ktorý na obrazovku vypisuje nejaký parameter ktorý mu dáme. Ukázali sme si, že šikovnou voľbou znaku a použitím forcyklu vieme vykreslovať tzv. ASCII art - obrazce tvorené zo znakov abecedy, alebo špeciálnych znakov. Na dnešnej hodine si ukážeme novú knižnicu tkinter, ktorá nám veľmi jednoducho umožní vytvárať geometrické tvary ako štvorec, obdĺžnik a podobne. Okrem vykresľovania obrazcov tkinter umožňuje pracovať aj s textom a tento text rôzne štylizovať (veľkosť, podčiarknutie, hrubý text, nastavenie fontu, ...).

Aby sme mohli niečo kresliť, musíme si najprv importovať knižnicu tkinter a následne vytvoriť kresliacu plochu. Kresliacu plochu vieme vytvoriť pomocou príkazu

Canvas(width = nejaka_sirka, height = nejaka_vyska).

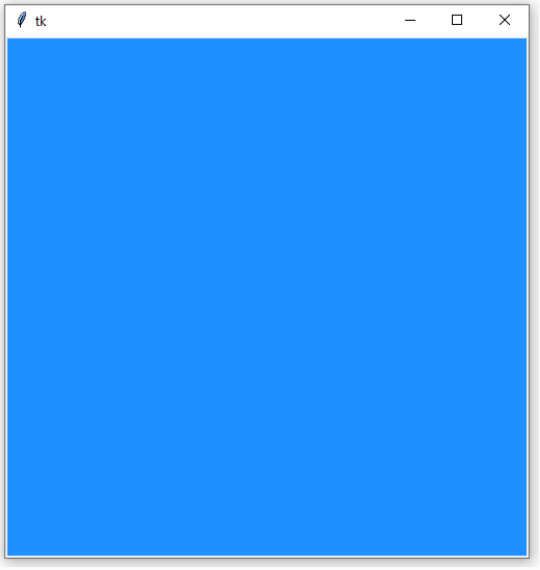
```
import tkinter

canvas = tkinter.Canvas(width = 400, height = 400)
canvas.pack()
```

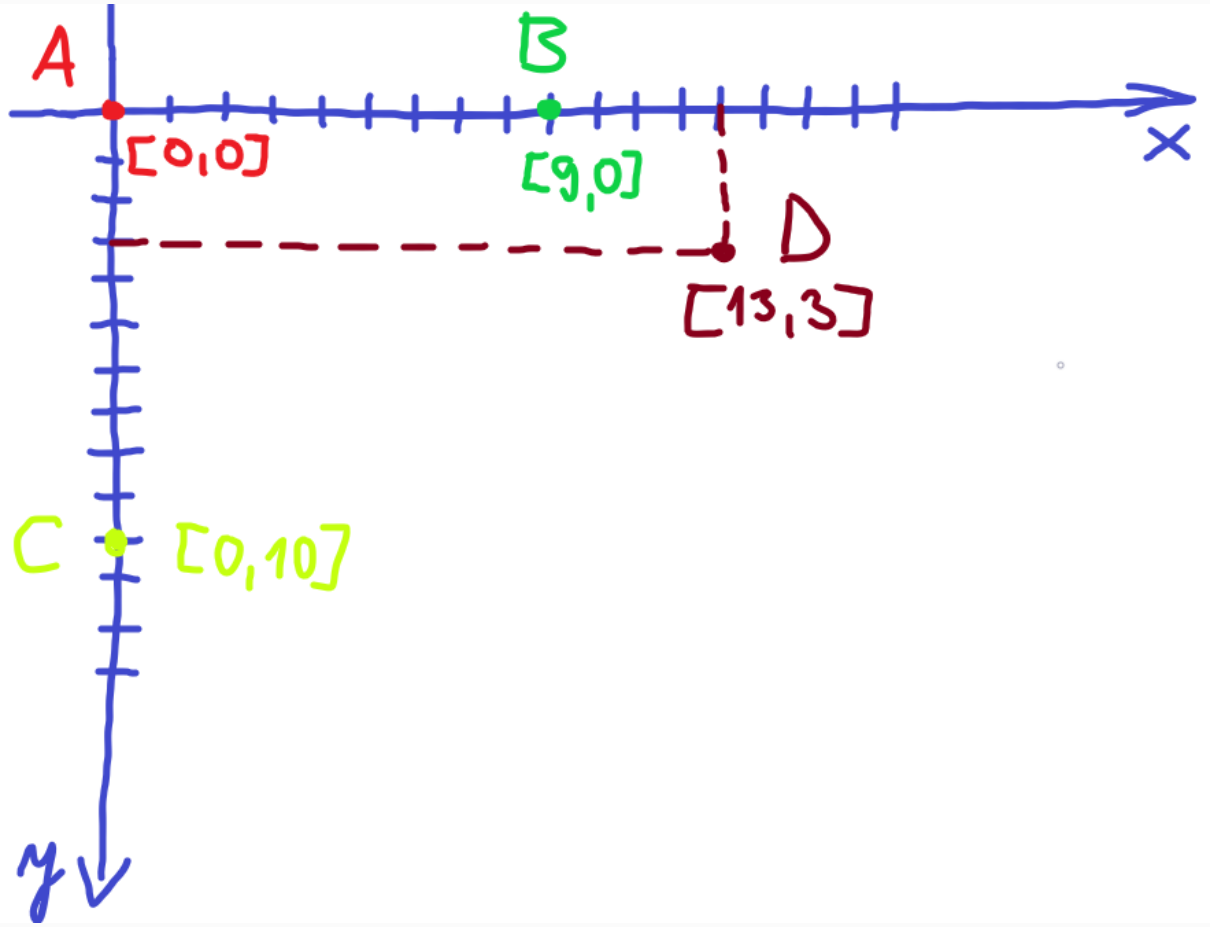
Vytvorili sme si premennú canvas, do ktorej tkinter uložil objekt (zatiaľ sa nezamýšľajte nad tým čo to objekt v Pythone je) nášho kresliaceho plátna. Všimnite si, že sme pri vytváraní špecifikovali šírku a výšku nášho plátna na **400px**. Tieto rozmery si môžeme zvoliť ľubovoľné. Do takéhoto plátna potom môžeme kresliť jednotlivé obrazce ako sú čiary, geometrické tvary a podobne. Príkaz **canvas.pack()** pracuje s naším vytvoreným objektom canvas a zabezpečí správne zobrazenie okna a kresliaceho plátna po spustení programu.

Do riadku, kde vytvárame naše plátno môžeme pridať do zátvoriek vedľa width a height ešte aj parameter **bg = 'nejaka_farba'**, ktorý nastaví nášmu plátnu pozadie na danú farbu. Farby píšeme do úvodzoviek (ako textový reťazec) a ich zoznam si môžete pozrieť na obrázku tu.

Ak teda chceme vytvoriť plátno s pozadím dodgerblue, vytvoríme canvas ako **canvas = Canvas(width = 400, height = 400, bg = 'dodgerblue')**. Naše plátno potom bude vyzerať takto:



Ako sme si už povedali, na plátne vieme vytvárať jednotlivé tvary. Tvary v knižnici tkinter, tak ako aj tvary v skutočnosti sú vlastne tvorené z niekoľkých úsečiek. Úsečky sú tvorené dvoma bodmi a tieto body majú svoje vlastné súradnice. Súradnice sa zapisujú do tzv. súradnicovej sústavy, kde x-ová os je vodorovná a y-ová os je vertikálna. Pozor však na to, že v prípade tkinter sa stred súradnicovej sústavy nenachádza v strede plátna, ale v ľavom hornom rohu plátna. Zároveň platí, že v prípade y-ovej osi sú smerom nadol kladné čísla a smerom nahor záporné čísla.



Poriadne si zopakujte zápis **bodov** do súradnicovej sústavy. Každý bod sa zapisuje ako **[x, y]**, kde v hranatých zátvorkách je na prvom mieste x-ová súradnica a na druhom mieste y-ová súradnica.

Tip: *ak chcete kresliť zo stredu canvas, môžete si vytvoriť dve premenné x a y, ktoré sa budú rovnať polovici z width canvas a polovici z height vášho canvas. Takto potom tieto súradnice môžete používať pri vytváraní vlastných bodov.*

Jednotlivé body potom vieme spájať a vytvoriť **úsečku**, poprípade body zadať ako parametre do jednej z funkcií, ktoré potom vytvoria nejaký geometrický útvar.

Pre vytvorenie úsečky zadáme jednoduchý príkaz **canvas.create_line(bod1, bod2)**, kde bod1 a bod2 sú body v tvare [x, y]. Tento príkaz tieto dva body jednoducho spojí do úsečky a dá jej čiernu farbu a hrúbku 1px. Ak chceme nastaviť úsečke inú farbu, môžeme do funkcie create_line okrem bodov pridať aj parameter **fill = 'farba'**. Ak chceme úsečke nastaviť inú hrúbku ako 1, je treba do funkcie pridať parameter **width = hrubka**.

Ak chceme teda vytvoriť červenú úsečku s dĺžkou 160, hrúbkou 2px a v strede plátna 400x400 s farbou dodgerblue, tak kód môže vyzerať nasledovne:

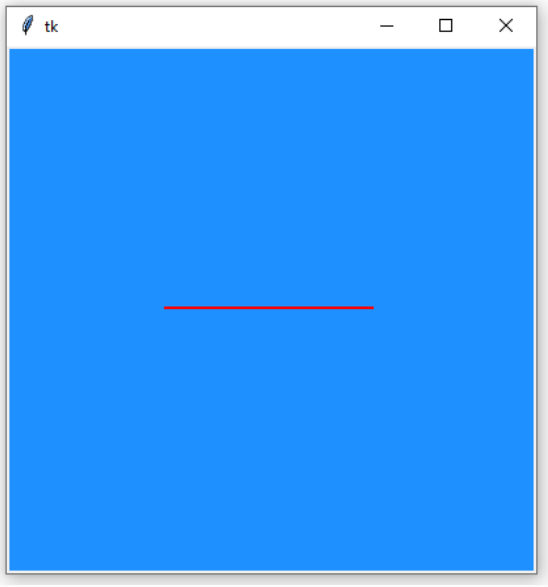
```
import tkinter

canvas = tkinter.Canvas(width = 400, height = 400, bg = "dodgerblue")
canvas.pack()

x = 200
y = 200

canvas.create_line([x - 80, y], [x + 80, y], width = 2, fill = 'red')
```

a po spustení vyzerá takto:



6.1 Úloha

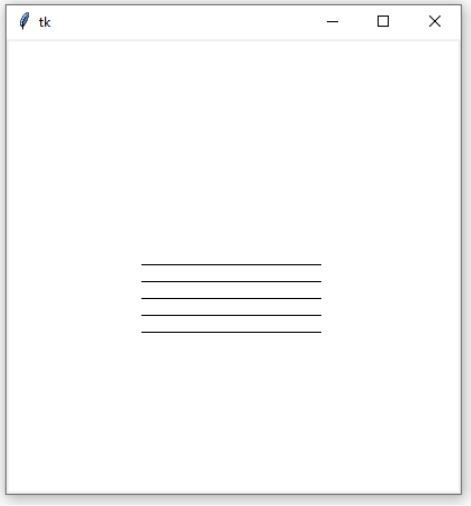
Napíšte program, v ktorom si vytvoríte **plátno**, ktorému nastavíte nejakú **šírku, výšku** a **farbu pozadia**. Farbu si môžete vybrať z tabuľky vyššie. Nezabudnite si do programu importovať knižnicu tkinter a použiť na plátno príkaz **pack()**.

6.2 Úloha

Napíšte program v ktorom do vášho plátna pridáte dve úsečky. Úsečkám zvoľte nejaké body ktoré ich budú tvoriť. Ďalej každej úsečke nastavte nejakú farbu a hrúbku.

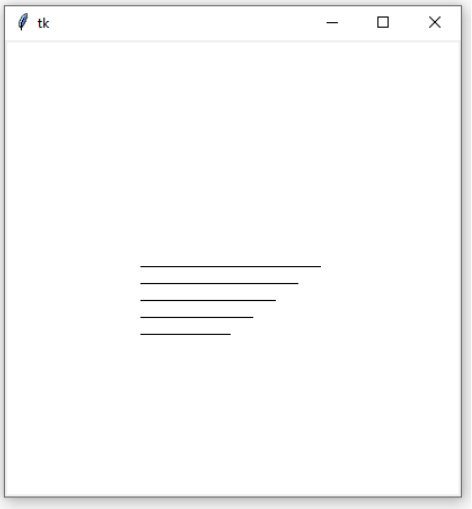
6.3 Úloha

Napíšte program v ktorom si vytvoríte plátno bielej farby (veľkosť je na vás). Do tohto plátna bude vašou úlohou vytvoriť **5 horizontálnych čiar** pod sebou (dĺžku čiar si prispôsobte sami v súradniciach. Program môže vyzerať napríklad takto:



6.4 Úloha

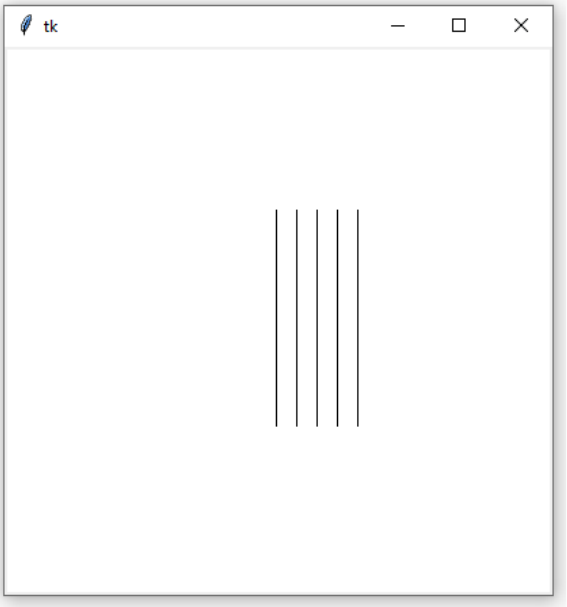
Napíšte program v ktorom si vytvoríte plátno bielej farby (veľkosť je na vás). Do tohto plátna bude vašou úlohou vytvoriť **5 horizontálnych čiar** pod sebou. Dĺžka čiar sa bude každou čiarou **zmenšovať**. To znamená prvá čiara najväčšia, druhá čiara o niečo menšia atď. Program môže vyzerať napríklad takto:



Pokračujeme s tkinter

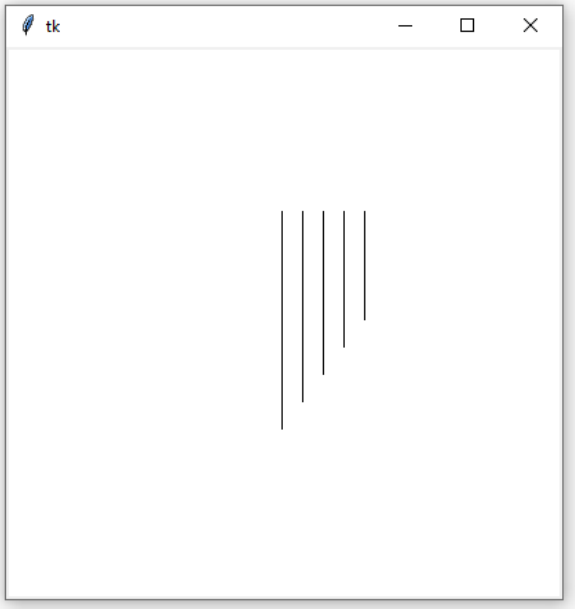
7.1 Úloha

Napíšte program v ktorom si vytvoríte plátno bielej farby (veľkosť je na vás). Do tohto plátna bude vašou úlohou vytvoriť **5 vertikálnych čiar** vedľa seba (dĺžku čiar si prispôsobte sami v súradniciach. Program môže vyzerať napríklad takto:



7.2 Úloha

Napíšte program v ktorom si vytvoríte plátno bielej farby (veľkosť je na vás). Do tohto plátna bude vašou úlohou vytvoriť **5 vertikálnych čiar** vedľa seba. Dĺžka čiar sa bude každou čiarou **zmenšovať**. To znamená prvá čiara najväčšia, druhá čiara o niečo menšia atď. Program môže vyzerať napríklad takto:



Už vieme ako si vytvoriť jednotlivé čiary/úsečky. Teraz si poďme ukázať, ako si vytvoriť **štvorec, obdĺžnik a trojuholník**. Tieto tvary sú tvorené z viacerých úsečiek. Pre štvorec a obdĺžnik by sme si teda mohli vhodne zvoliť 4 body a tieto body pospájať úsečkami. Podobne pri trojuholníku by sme si mohli zvoliť tri body a tieto pospájať úsečkami. Aby sme si však uľahčili prácu, použijeme vstavané funkcie v knižnici tkinter, ktoré toto urobia za nás.

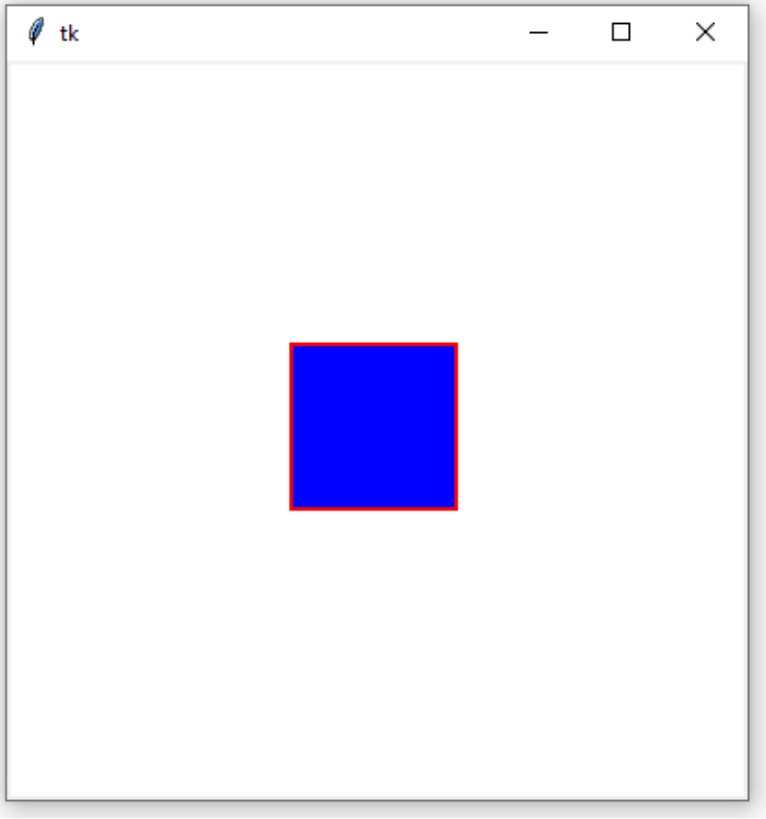
Na vykreslenie štvorca alebo obdĺžnika môžeme použiť príkaz: **canvas.create_rectangle(bod1, bod2)**, kde bod1 a bod2 sú body v tvare [x, y]. Tieto body predstavujú ľavý horný roh a pravý dolný roh daného štvorca/obdĺžnika. Do tejto funkcie opäť môžeme pridať parametre **width = hrubka**, ktorý nastaví hrubku čiary a parameter **fill = 'farba'** ktorý daný štvorec/obdĺžnik vyplní zadanou farbou. Ak chceme zmeniť farbu rámu, pridáme parameter **outline = 'farba'**.

```
import tkinter

canvas = tkinter.Canvas(width = 400, height = 400, bg = "white")
canvas.pack()

x = 200
y = 200

canvas.create_rectangle([x - 45, y - 45], [x + 45, y + 45],
                        width = 2, fill = 'blue', outline = 'red')
```



Na vykreslenie trojuholníka použijeme príkaz:

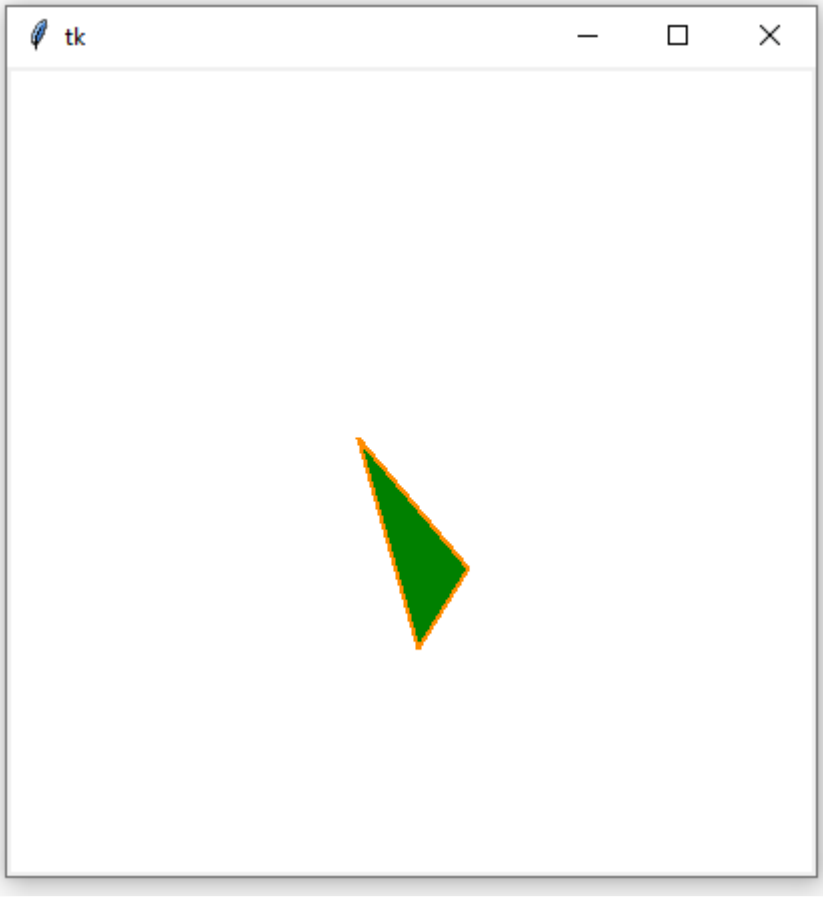
canvas.create_polygon(bod1, bod2, bod3). Zase platí, že vieme zvoliť parameter width, fill a outline a tým si prispôsobiť farbu, orámovanie a jeho hrúbku.

```
import tkinter

canvas = tkinter.Canvas(width = 400, height = 400, bg = "white")
canvas.pack()

x = 200
y = 200

canvas.create_polygon([x - 25, y - 15], [x + 30, y + 50],
                      [x + 5, y + 90],
                        width = 2, fill = 'green', outline = 'darkorange')
```

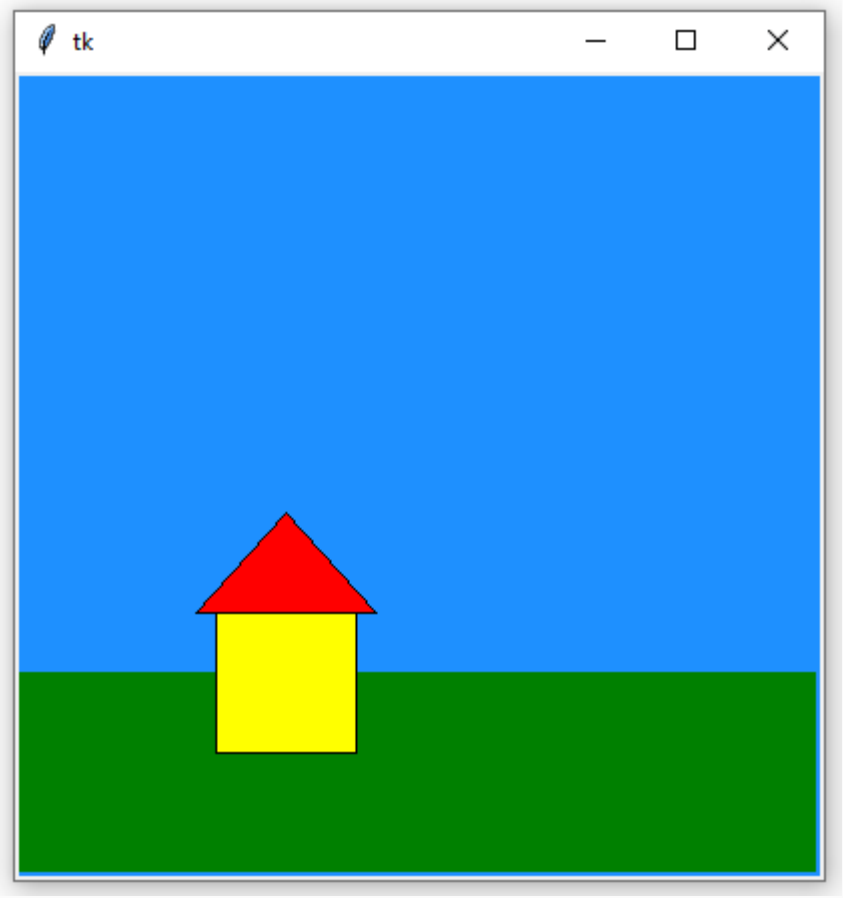


7.3 Úloha

Vašou úlohou bude si vytvoriť plátno, ktorého veľkosť a fraba je na vás. Teraz do tohto plátna pridajte jeden štvorec/obdĺžnik s nejakou farbou výplne a orámovania. Hrúbku orámovania môžete zmeniť tiež. Nakoniec pridajte do plátna aj jeden trojuholník. Farba aj orámovanie je opäť na vás.

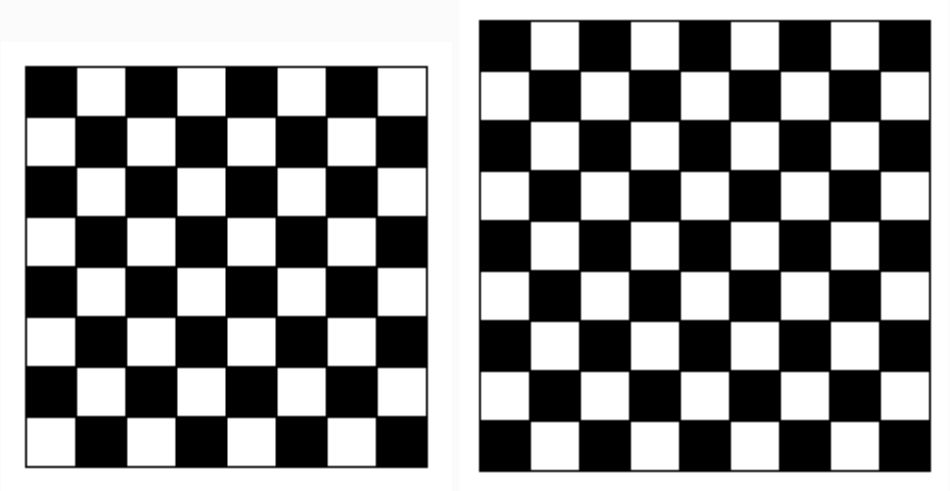
7.4 Úloha

Vytvorte program, ktorý vykreslí dom s oblohou a záhradou podobne ako na obrázku:



BONUS:

Ak vám zostal čas, tak vytvorte program, ktorý si od užívateľa vypýta veľkosť šachovnice a túto šachovnicu potom vykreslí.



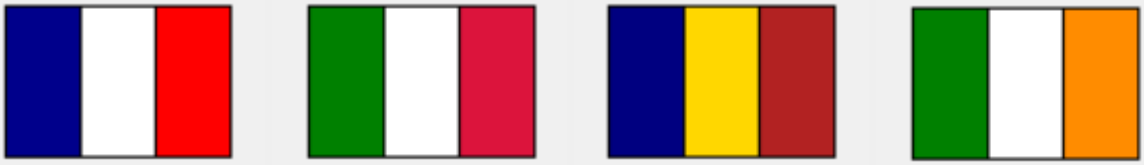
Tkinter - vytváranie vlastných obrázkov - vlajky

8.1 Úloha

Postupne vyvorte porgramy, ktoré vykreslia na plátno tieto vlajky:

8.2 Úloha

Postupne vytvorte programy, ktoré vykreslia na plátno tieto vlajky:



8.3 Úloha

Postupne vytvorte programy, ktoré vykreslia na plátno tieto vlajky. Všimnite si, že tentokrát vlajky nemajú žiadne orámovanie:

8.4 Úloha

Postupne vytvorte programy, ktoré vykreslia na plátno tieto vlajky. Všimnite si, že tentokrát vlajky nemajú žiadne orámovanie:



BONUS:

Ak vám zostal čas, tak môžete pracovať samostatne na ďalších ľubovoľných vlajkách. Vlajky sveta si môžete vyhľadať v prehliadači po zadaní napríklad “flags of the world”. Poprípade ak chcete si môžete vytvoriť vlastnú vlajku podľa svojej fantázie.

Tkinter - Kruh, text a dopravné značky

Na vykreslenie kruhu musíme použiť príkaz **canvas.create_oval(bod1, bod2)**, kde **bod1** a **bod2** sú body v tvare **[x, y]** kde x a y sú jednotlivé súradnice. Kruh nám príkaz vykreslí len ak body zvolíme rovnomerne, podobne ako tomu bolo pri príkaze **canvas.create_rectangle(...)**. Ak totiž zvolíme body nerovnomerne, tak nám príkaz vykreslí ovál, podobne ako create_rentangle vykresľoval obdĺžnik.

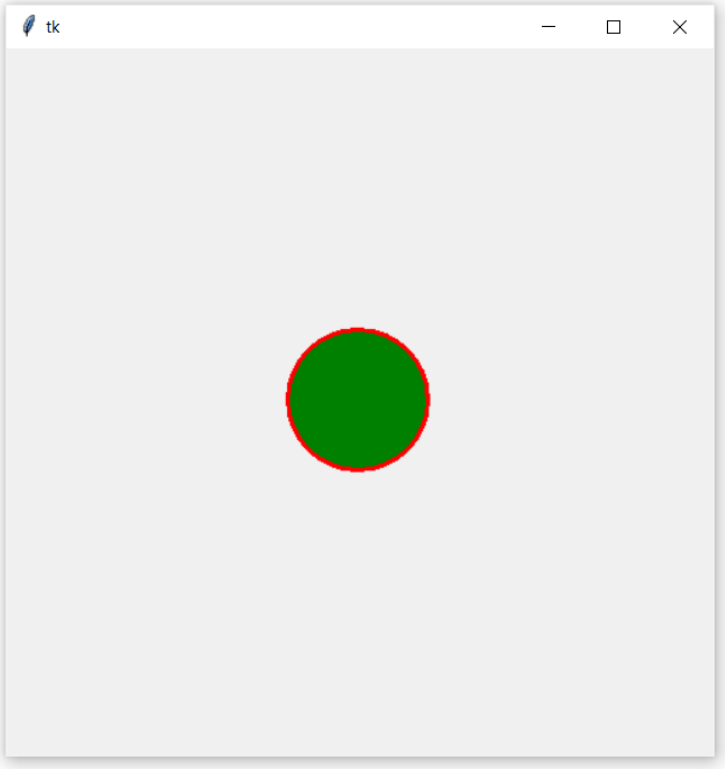
Na obrázku nižšie je kód, v ktorom sme vytvorili kruh s polomerom 50, červeným orámovaním s hrúbkou 3 a so zelenou výplňou.

```
import tkinter

canvas = tkinter.Canvas(width = 500, height = 500)
canvas.pack()

x = int(canvas["width"]) / 2
y = int(canvas["height"]) / 2

canvas.create_oval([x - 50, y - 50], [x + 50, y + 50], outline = "red", fill = "green", width = 3)
```



9.1 Úloha

Vašou pvou úlohou bude nakresliť na plátno japonskú vlajku, ktorá môže vyzerať napríklad takto:

Okrem tvarov vieme v pythone vytvárať aj nejaký text, nastaviť mu farbu, veľkosť a názov fontu. Použijeme príkaz :
canvas.create_text(bod, text = nejakyText, font = (typ, velkost, tucne/sikme/podciarknute), fill = farba)

Ako príklad si uvedieme tento kód:

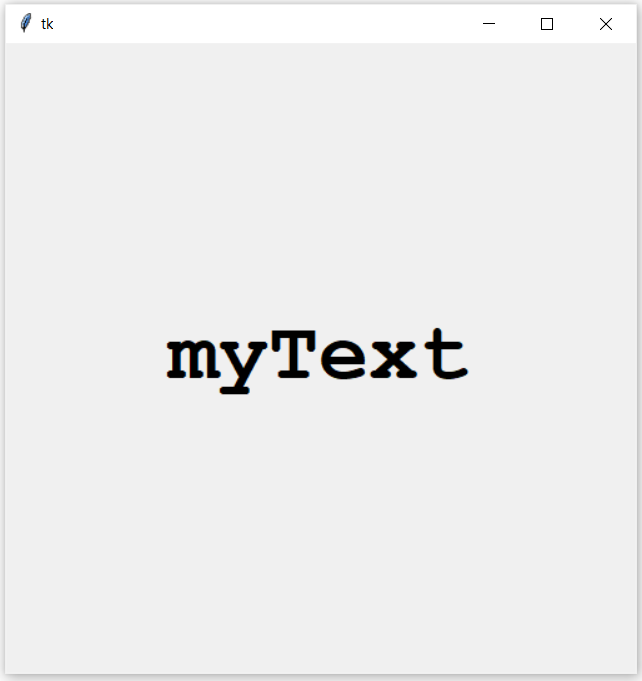
```
import tkinter

canvas = tkinter.Canvas(width = 500, height = 500)
canvas.pack()

x = int(canvas["width"]) / 2
y = int(canvas["height"]) / 2

canvas.create_text([x, y], text = "myText", font = ("Courier", 50, "bold"))
```

Vytvorili sme **text "myText"** ktorý sa nachádza v bode **[x, y]**, typ fontu je Courier, veľkosť fontu je 50 a je bold, čiže hrubý. Keď by sme cheli nastaviť danému textu farbu napríklad na červenú, tak doplníme parameter **fill = "red"** ako v predošlých príkladoch. Kód po spustení vyzerá takto:



9.2 Úloha

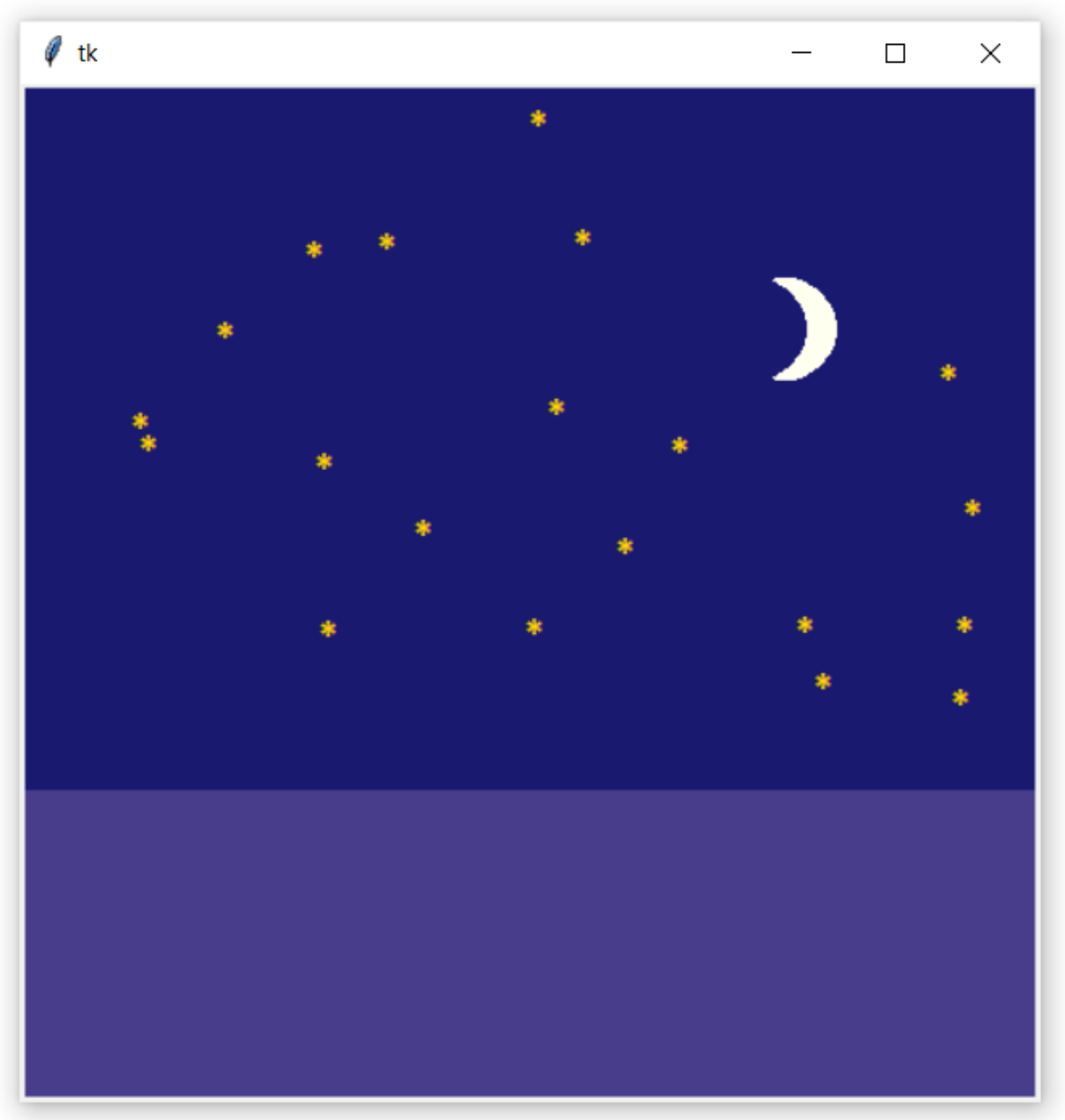
Tvary sa môžu navzájom prekrývať. Vytvorte nasledovné dopravné značky:

9.3 Úloha

Vytvorte značku STOP.

9.4 Úloha

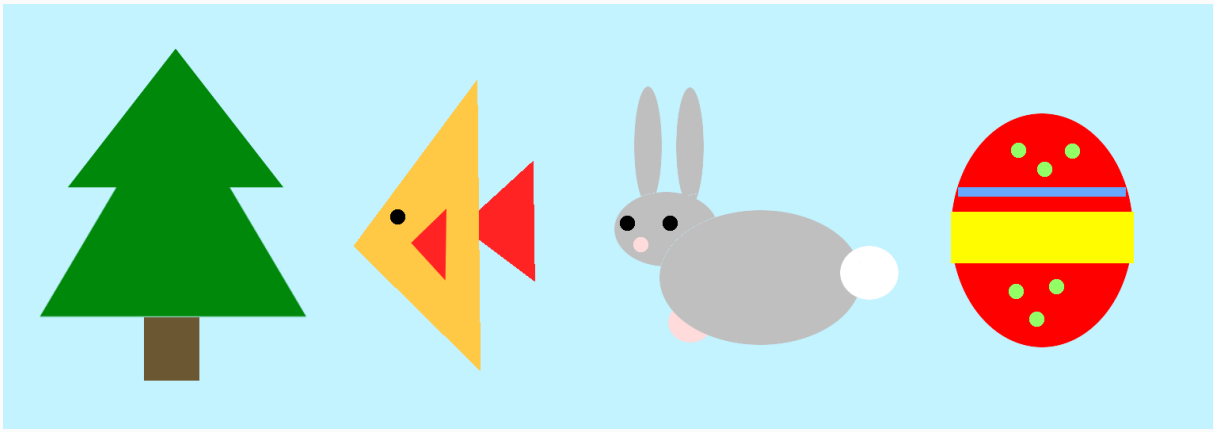
Vytvorte náhodne vygenerovanú nočnú oblohu s hviezdami a mesiacom. Hviezdy môžu byť napr. žlté a mesiac biely. Na generovanie hviezd a mesiaca použite cyklus a funkciu random, ktorá bude generovať súradnice. Hviezdy môžete generovať ako text. Váš program musí fungovať tak, aby zakaždým generovalo inú nočnú oblohu.



Opakovanie

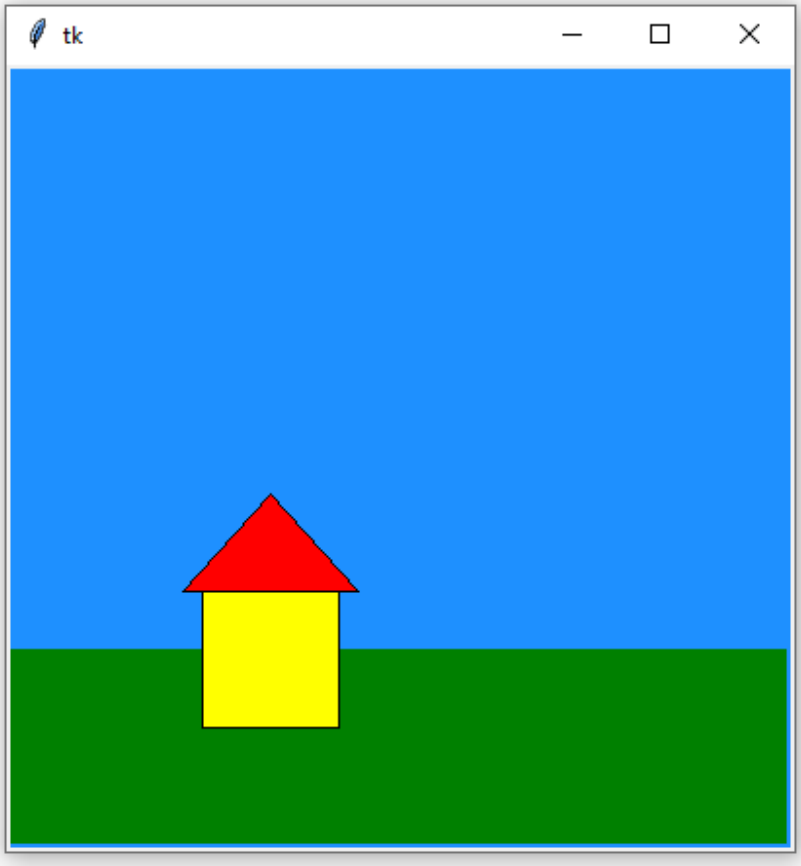
10.1 Úloha

Vašou úlohou bude samostatná práca, kde využijete všetky doteraz nadobudnuté poznatky. Môžete sa inšpirovať úlohami z minula, alebo nakresliť niečo z nasledovného:

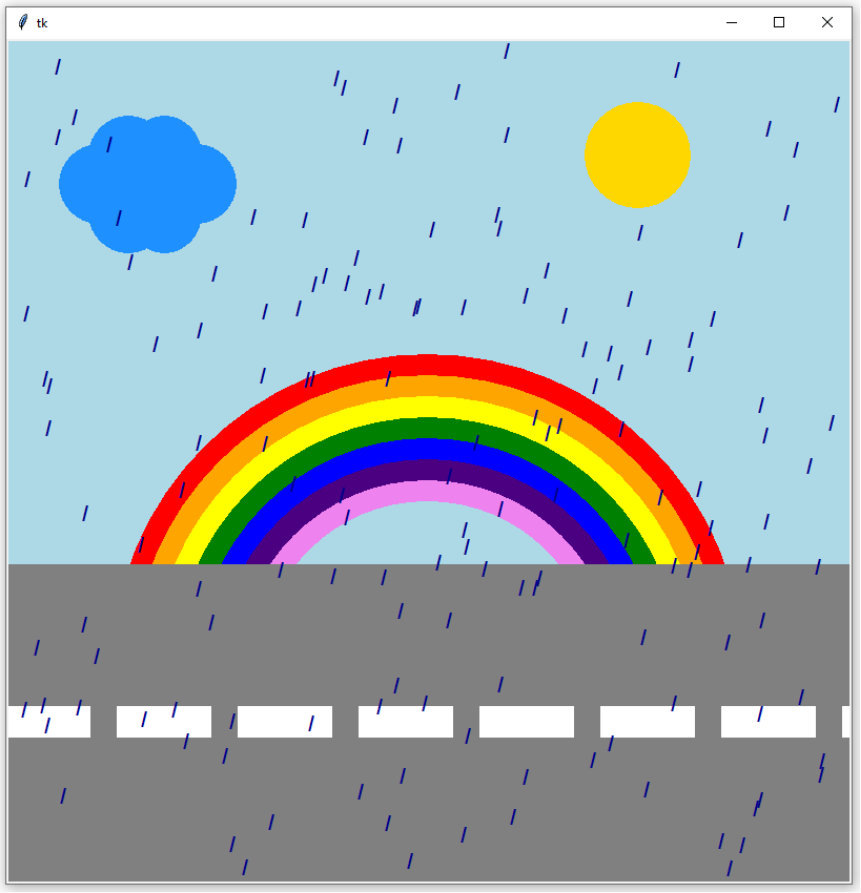


To znamená napríklad vytvorte vianočný stromček a ozdobte ho, pridajte pod stromček darčeky.

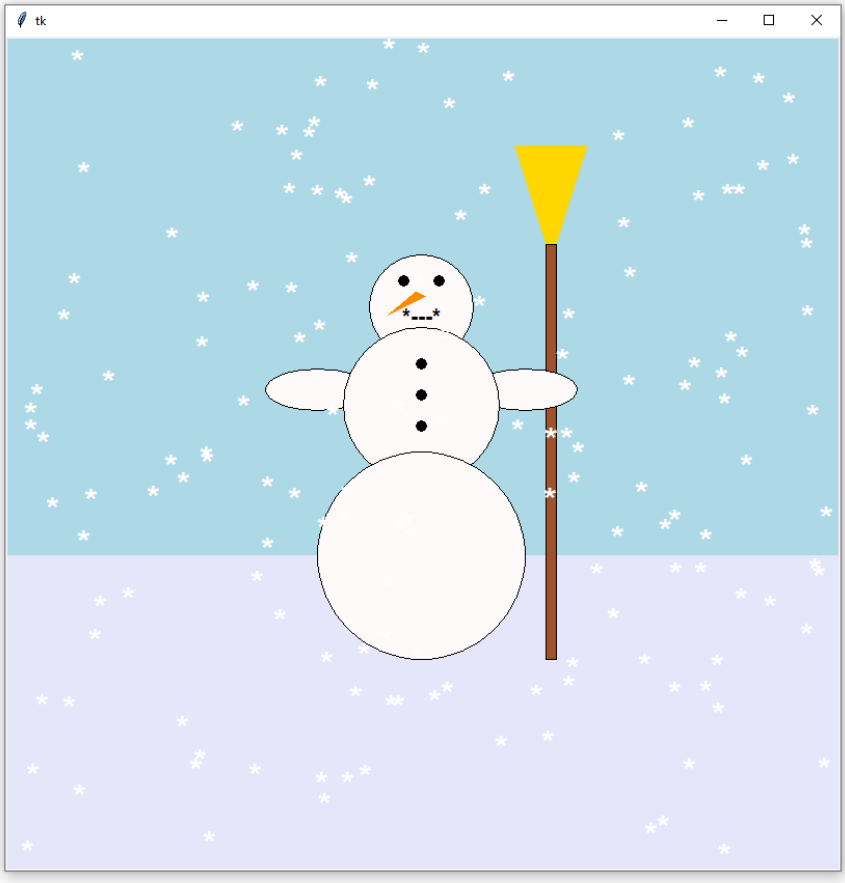
Vytvorte veľkonočné vajíčko a vyfarbite a ozdobte ho, k tomu môžete nakresliť viacerých veľkonočných zajacov s rôznymi farbami. Poprípade môžete skúsiť nakresliť podmorský svet s rybami, kameňmi, riasami a bublinami. Ak chcete, môžete si vylepšiť dom z minula tým, že pridáte oblaky, slnko a podobne. Pokiaľ vás ani jedna aktivita z týchto nezaujíma, stále máte možnosť nakresliť nejaké štátne vlajky, ktoré ste nestihli alebo nerobili vôbec, poprípade nejaké dopravné značky, alebo úlohy týkajúce sa tkinter z minula.



Inšpirovať sa môžete napríklad aj takouto cestou s modrou oblohou so slnkom a oblakom v pozadí, s náhodne genrujúcim sa dážďom, a s dúhou v strede obrázku.

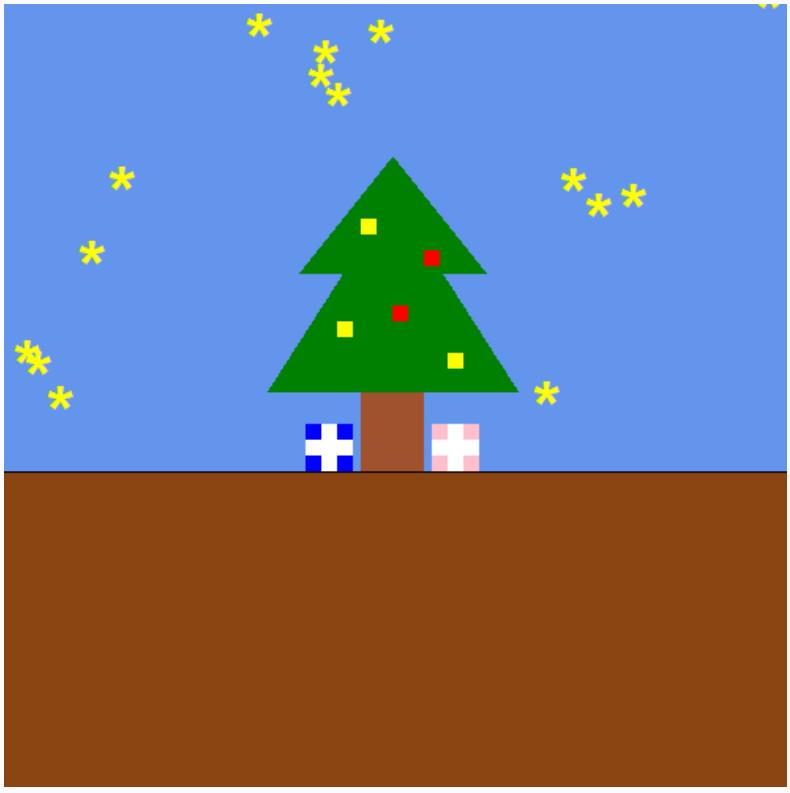


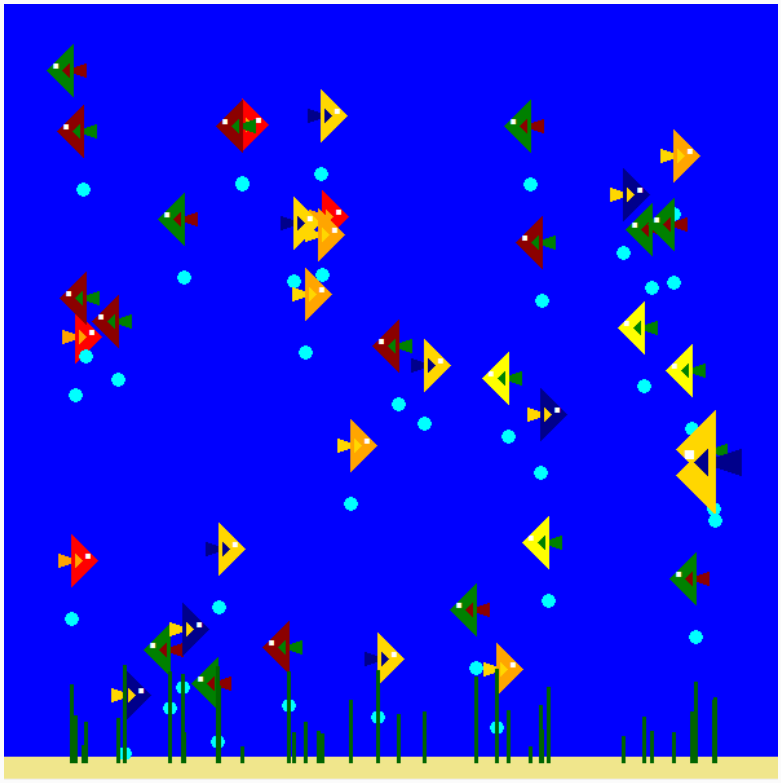
Ďalšia z možných inšpirácií je zasnežená krajina so snehuliakom.



VÝTVORY VAŠICH SPOLUŽIAKOV

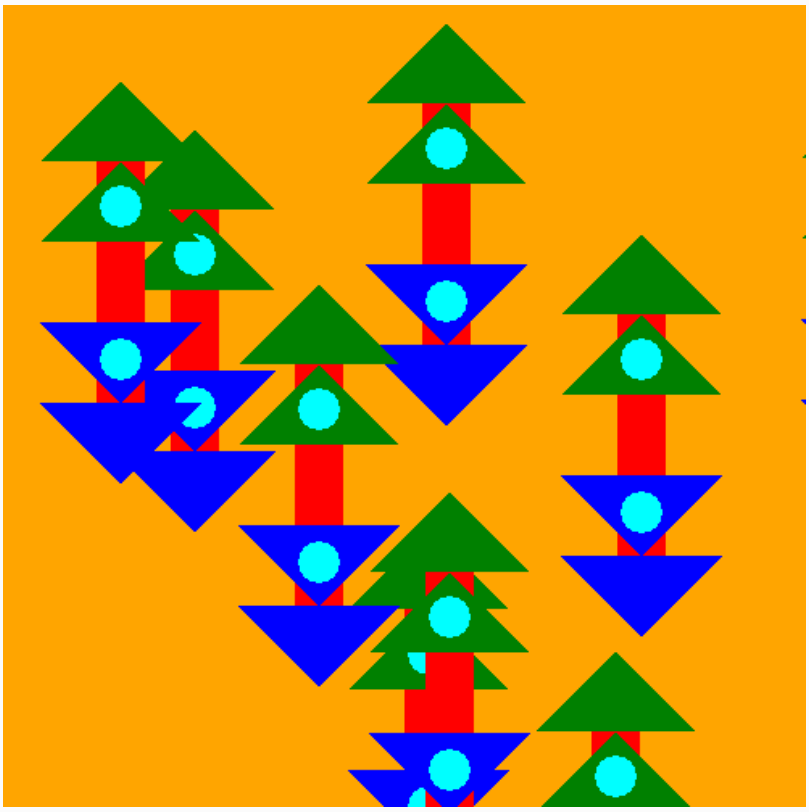
Tomáš - Vianočný stromček s darčekmi a náhodne generujúcimi sa hviezdami

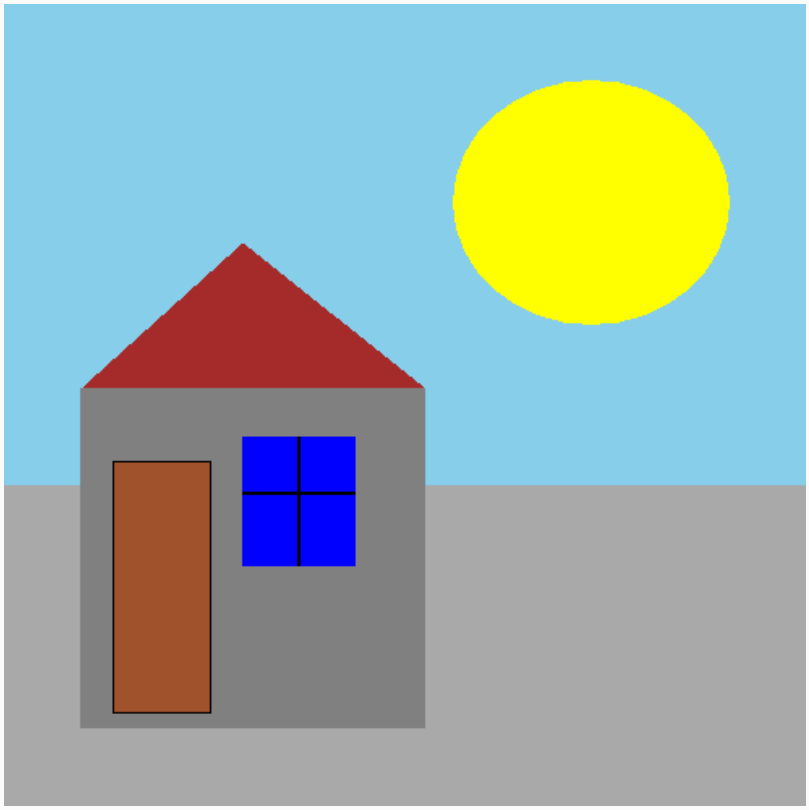




Patrik - Náhodne generujúci sa podmorský svet s rybami, bublinami a riasami

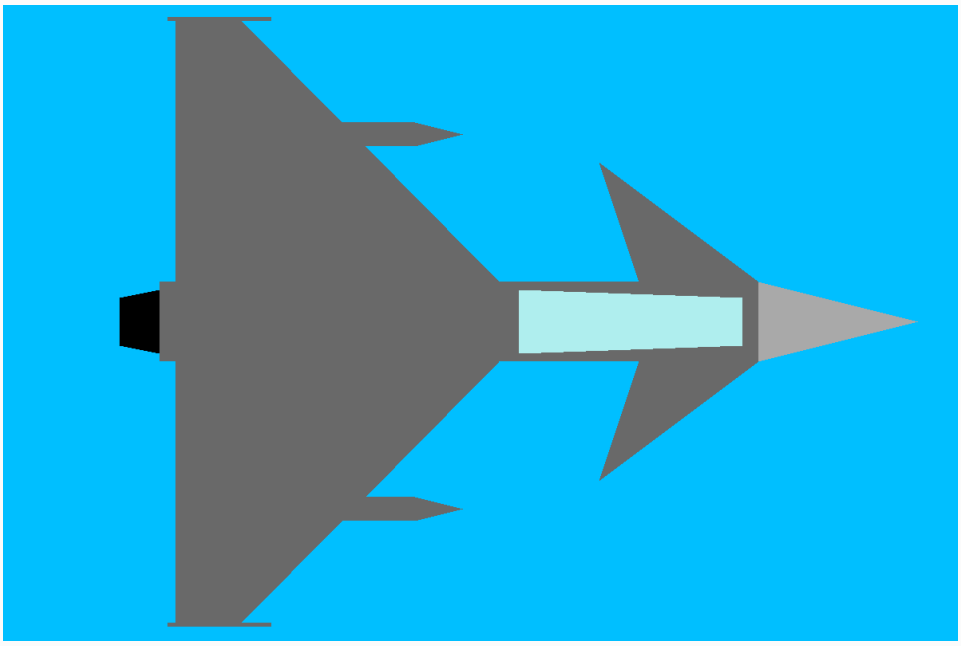
Stano - Generovanie ornamentu

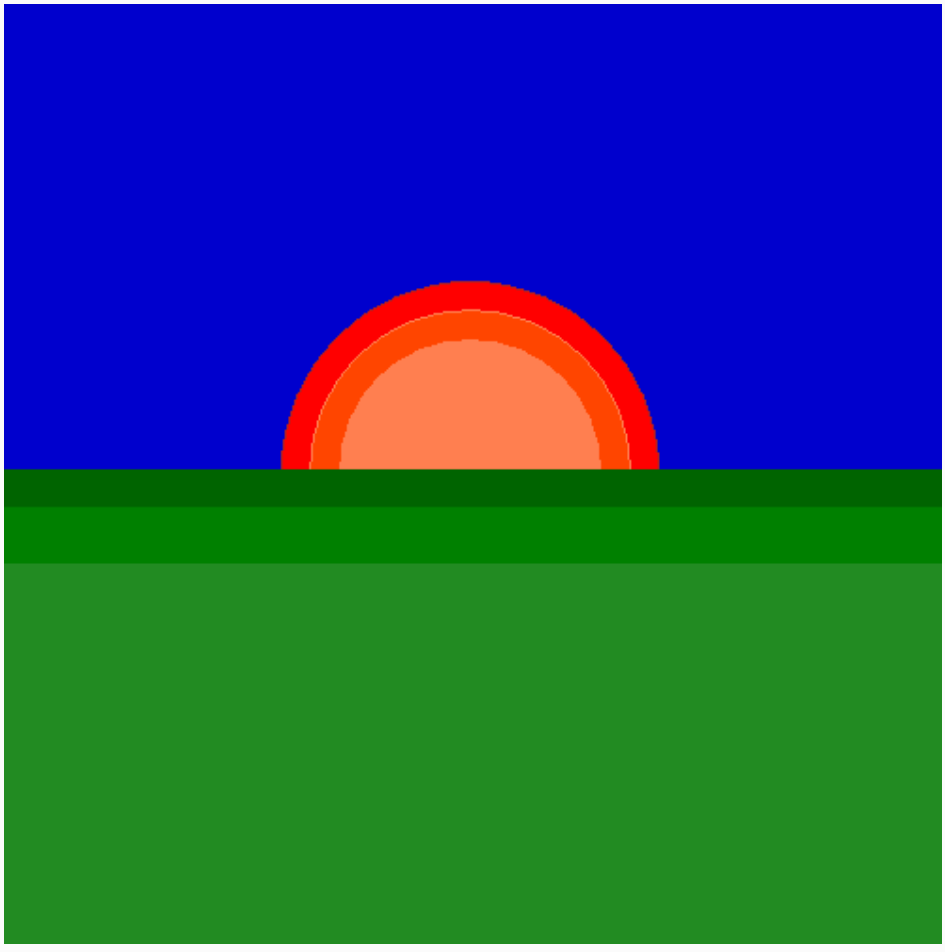




Jakub - Dom v meste s detailmi

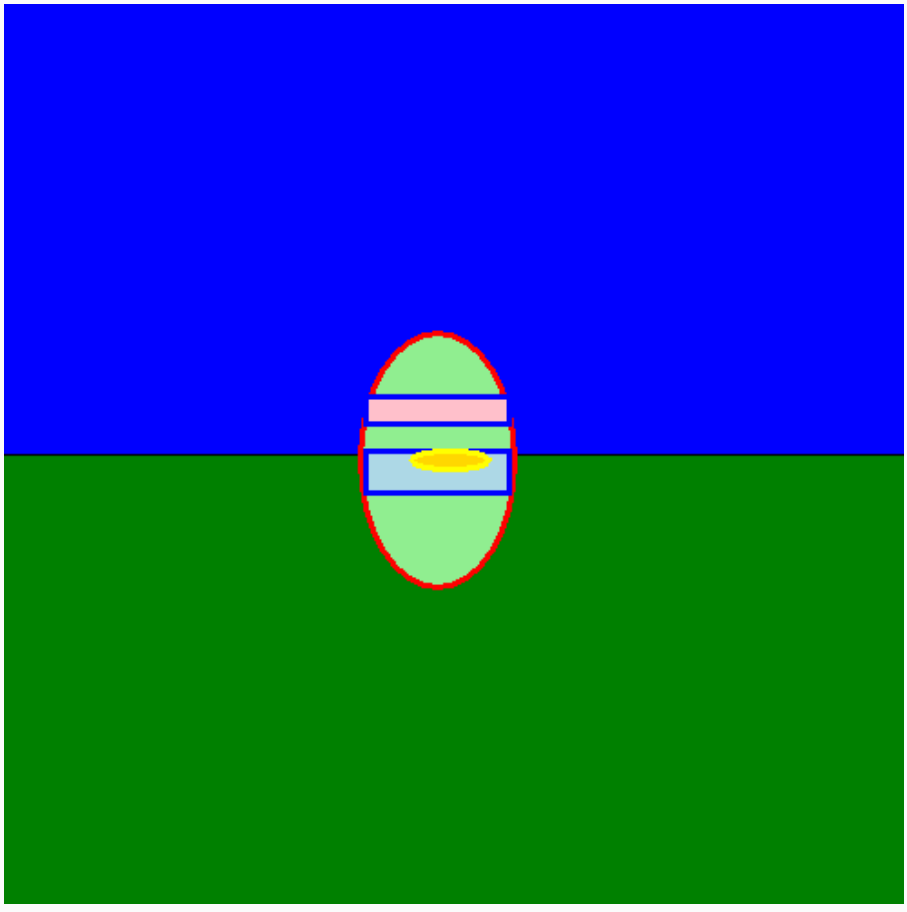
Nicholas - Stíhačka





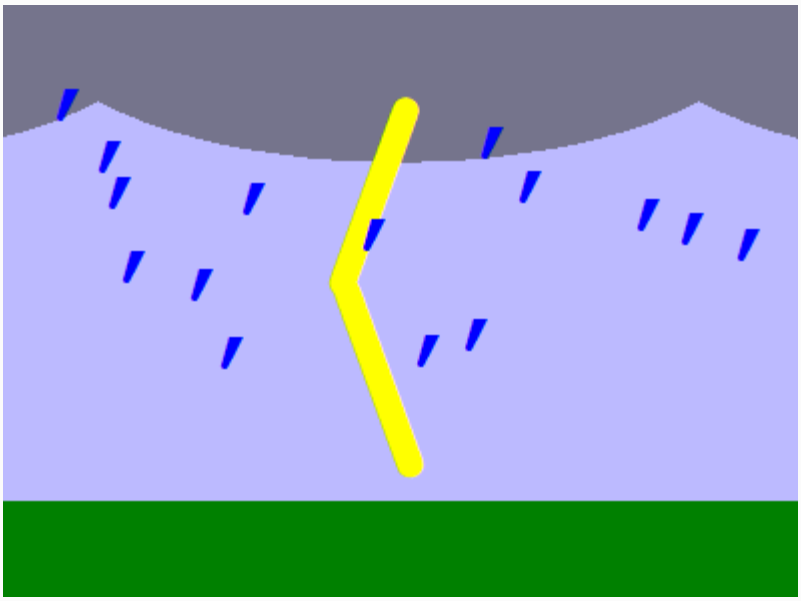
Alica - Západ slnka

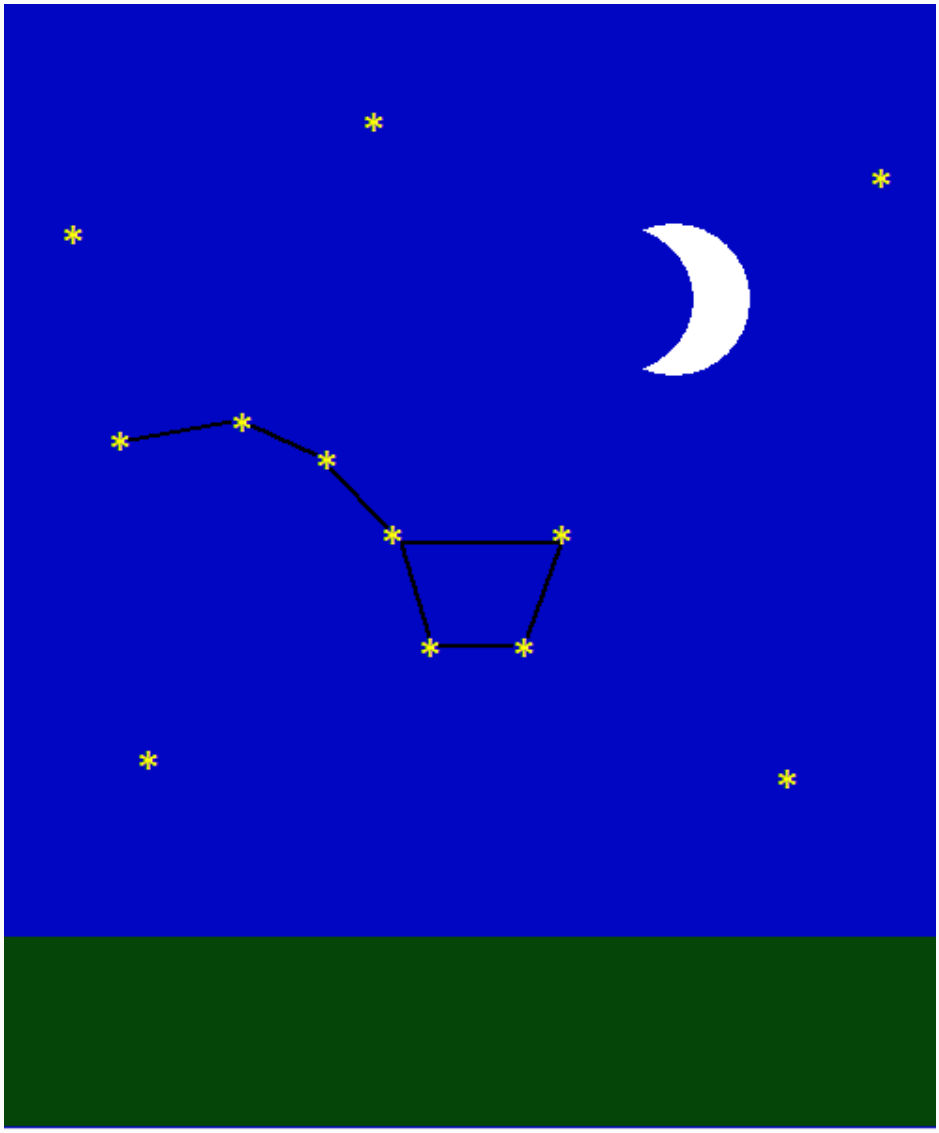
Michal - Lúka s dúhou



Matúš - Veľkonočné vajíčko

Alex - Búrka





Marek - Súhvezdie