Zdroj: Andrej Blaho, http://python.input.sk

Grafika

Doteraz sme pracovali len v textovom režime: do textovej plochy (shell) sme z programu zapisovali len pomocou `print()`. V súčasných operačných systémoch (windows, linux, os, …) sú skoro všetky aplikácie grafické. Preto to skúsime aj my.

Budeme pracovať s modulom **tkinter**, ktorý slúži na vytváranie grafických aplikácii. My ho budeme využívať prakticky len na kreslenie na tzv. **plátno**, hoci tento modul zvláda aj iné typy grafických prvkov. Príkladom grafickej aplikácie, ktorá je kompletne napísaná v Pythone a používa **tkinter** je vývojové prostredie **IDLE**.

Základná grafická aplikácia

Minimálna grafická aplikácia je táto:

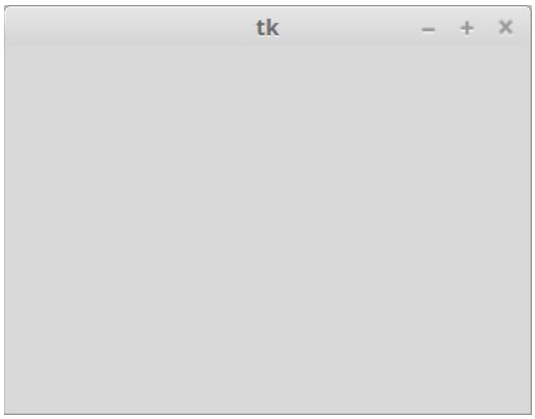
```
import tkinter

canvas = tkinter.Canvas()
canvas.pack()

# tu sa bude kresliť do grafickej plochy

tkinter.mainloop()
```

Príkazy majú postupne takéto vysvetlenie:

- `import tkinter` - vytvorí premennú `tkinter`, pomocou ktorej budeme mať prístup (bodkovou notáciou) k funkciám v module
- `canvas = tkinter.Canvas()` - vytvorí plátno grafickej aplikácie - priradili sme ho do premennej `canvas` (mohlo sa to volať hocijako inak, ale takto budeme lepšie rozumieť aj cudzím programom)
    - týmto vznikne malá grafická aplikácia (malé okno), aj plátno, ktoré ale zatiaľ nie je nikde v tejto aplikácii umiestnené
- `canvas.pack()` - až teraz sa umiestni naše plátno do grafickej aplikácie (do okna) - plátno je teraz pripravené, aby sme do neho mohli kresliť
- `tkinter.mainloop()` - vďaka tomuto príkazu grafická aplikácia v operačnom systéme naozaj *žije*, t.j. reaguje na klikanie, presúvanie, zmenu veľkosti, prekresľovanie, …

Posledný príkaz `tkinter.mainloop()` môžeme pri spustení pod **IDLE** vynechať, lebo práve **IDLE** ho spraví za nás (celý **IDLE** je naprogramovaný v Pythone, využíva `tkinter` a spúšťa na pozadí `mainloop`).

Po spustení tohto programu dostávame malú grafickú aplikáciu:



V tejto aplikácii sa nachádza plátno (šedá plocha vo vnútri okna), ktoré je zatiaľ prázdne, lebo sme ešte nezadali žiaden ďalší grafický príkaz na kreslenie.

Grafické príkazy

Všetky grafické príkazy budú pracovať s plátnom a preto budú začínať slovom `canvas`, za ktorým bude samotný príkaz. Tvar väčšiny príkazov bude

```
canvas.create_<meno útvaru>(x, y, ..., <ďalšie parametre>)
```

Bude to znamenať, že do grafickej plochy `canvas` sa nakreslí uvedený útvar, jeho poloha súvisí so súradnicami `(x, y)` a niekedy budeme okrem ďalších súradníc špecifikovať aj nejaké parametre. Postupne sa zoznámime s týmito grafickými objektmi:

```
canvas.create_text(...)          # vypíše zadaný text
canvas.create_rectangle(...)     # nakreslí obdĺžnik
canvas.create_oval(...)          # nakreslí elipsu
canvas.create_line(...)          # nakreslí lomenú čiaru
canvas.create_polygon(...)       # nakreslí polygón
canvas.create_image(...)         # nakreslí png obrázok
```

Text v grafickej ploche

Ukážme najjednoduchší grafický príkaz, ktorý do plochy (plátna) zapíše nejaký text. Do našej šablóny na vytvorenie grafickej aplikácie pridáme jeden príkaz na „nakreslenie“ (vypísanie) nejakého textu:

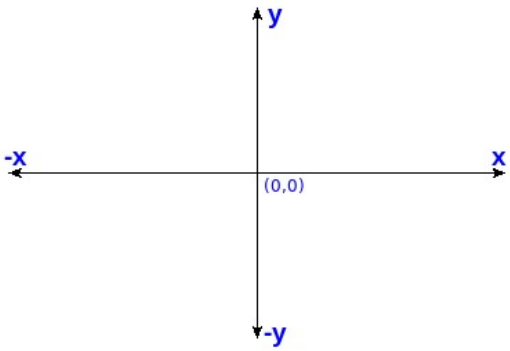
```
import tkinter

canvas = tkinter.Canvas()
canvas.pack()

canvas.create_text(150, 100, text='programujem v Pythone')
```

Po spustení dostávame takýto grafický výstup:

V šedej ploche sa na súradniciach (150, 100) vypísal zadaný text ale dosť malými písmenkami. Potrebujeme ale rozumieť, ako `tkinter` chápe súradnicovú sústavu.

Súradnicová sústava

Zo školskej matematiky vieme, že súradnicové osi **x** a **y** sú na seba kolmé a pretínajú sa v bode (0, 0). Orientácia týchto osí je takáto:



V počítačových programoch sa veľmi často bod (0, 0) presúva do ľavého horného rohu plátna, **x**-ová os prechádza zľava doprava po hornej hrane plátna a **y**-ová os prechádza zhora nadol po ľavej hrane plátna:

Aj takáto súradnicová sústava má záporné **x** a **y** ale, keď budeme kresliť mimo viditeľnú časť plátna, niektoré časti kresby nebudeme vidieť. Zatiaľ budeme pracovať s plátnom, ktorý má rozmery približne 380 x 260. Neskôr sa naučíme nastavovať ľubovoľné rozmery plátna.

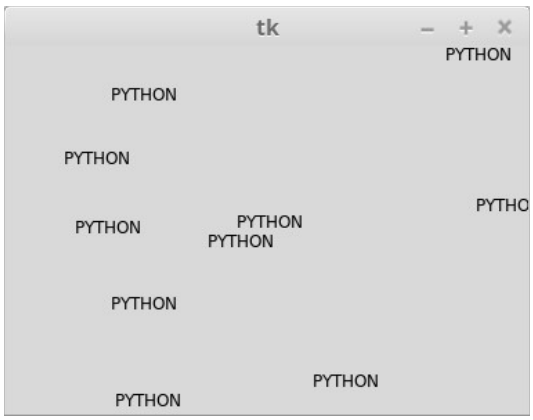
Využime generátor náhodných čísel (modul `random`) na generovanie náhodných bodov v grafickej ploche. Môžeme zapísať:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

x = random.randint(0, 380)
y = random.randint(0, 260)
canvas.create_text(x, y, text='PYTHON')
```

Tento program na náhodnú pozíciu umiestni text `'PYTHON'`. Keď ho zavoláme viackrát, zakaždým vypíše tento istý text ale na inú pozíciu. Môžeme sa o tom presvedčiť tak, že príkazy, ktoré generujú náhodnú pozíciu a vypisujú text, necháme pomocou for-cyklu opakovať 10-krát:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

for i in range(10):
    x = random.randint(0, 380)
    y = random.randint(0, 260)
    canvas.create_text(x, y, text='PYTHON')
```

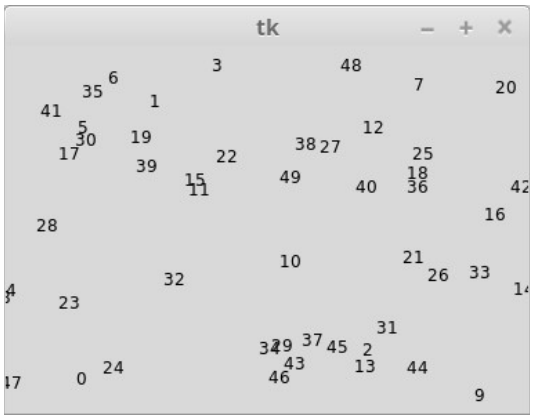
dostávame takýto obrázok:



Stačí namiesto výpisu `'PYTHON'` vypisovať premennú cyklu:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

for i in range(50):
    x = random.randint(0, 380)
    y = random.randint(0, 260)
    canvas.create_text(x, y, text=i)
```

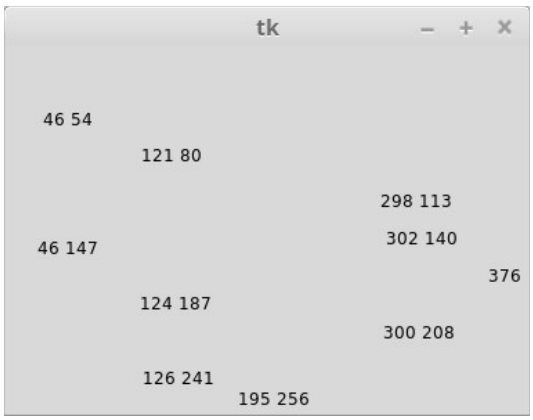
dostávame takýchto 50 náhodne vygenerovaných bodov:



Pri vypisovaní textov pomocou `create_text` môžeme použiť nielen znakové reťazce a celočíselné premenné, ale napríklad aj takéto dvojice:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

for i in range(10):
    x = random.randint(0, 380)
    y = random.randint(0, 260)
    canvas.create_text(x, y, text=(x, y))
```

teraz okrem 10 náhodne vygenerovaných bodov sa dozvedáme aj ich súradnice:



Tu treba poznamenať, že samotné súradnice vypisovaného textu označujú presný stred tohto textu, tak ako to ukazuje nasledovný zväčšený obrázok, v ktorom sa vypísal text `'PYTHON'`. Je tu zobrazená aj poloha (x, y), ktorá sa zadala v príkaze `create_text`:

Kreslenie obdĺžnikov

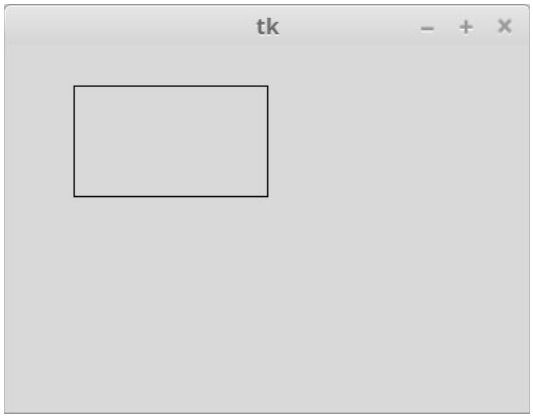
Na nakreslenie obdĺžnikov slúži grafický príkaz:

```
canvas.create_rectangle(x1, y1, x2, y2)
```

Keďže takýto obdĺžnik bude mať strany rovnobežné s osami **x** a **y**, na jednoznačné zadefinovanie obdĺžnika nám budú stačiť ľubovoľné dva protiľahlé vrcholy. Najčastejšie to bude ľavý horný bod `(x1, y1)` a pravý dolný `(x2, y2)`. Zapíšme napríklad:

```
import tkinter

canvas = tkinter.Canvas()
canvas.pack()

canvas.create_rectangle(50, 30, 190, 110)
```

Nakreslí obdĺžnik, ktorého ľavý horný vrchol je (50, 30) a pravý dolný je (190, 110):



Keďže pri kreslení obdĺžnika môžeme určiť jeho ľubovoľné dva protiľahlé vrcholy, všetky nasledovné volania by nakreslili presne ten istý obdĺžnik, ako sa nakreslil v predchádzajúcom programe:

```
canvas.create_rectangle(190, 110, 50, 30)
canvas.create_rectangle(50, 110, 190, 30)
canvas.create_rectangle(190, 30, 50, 110)
```

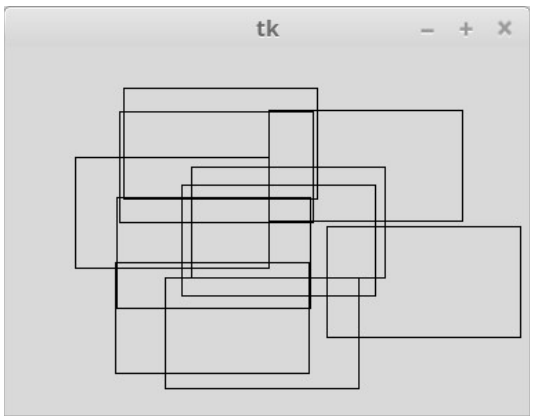
Uvedomte si, že šírka tohto obdĺžnika (veľkosť vodorovnej strany) je `190 - 50`, teda `140` a výška je `110 - 30`, teda `80`. Ak by sme teraz zapísali:

```
import tkinter

canvas = tkinter.Canvas()
canvas.pack()

x, y = 50, 30
sirka, vyska = 140, 80
canvas.create_rectangle(x, y, x + sirka, y + vyska)
```

dostaneme opäť ten istý obdĺžnik. Tento zápis má ale ešte jednu výhodu: veľmi jednoducho vieme nakresliť rovnaký obdĺžnik, ale na inom mieste. Stačí zmeniť `x, y` a obdĺžnik rozmerov `140 x 80` sa nakreslí na novej pozícii. Pomôžeme si podobne ako pri náhodných pozíciách textu aj pri kreslení obdĺžnikov na náhodné pozície pomocou funkcií modulu `random`. Nakreslíme 10 rovnako veľkých obdĺžnikov na náhodné pozície:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

for i in range(10):
    x = random.randint(10, 240)
    y = random.randint(10, 180)
    sirka, vyska = 140, 80
    canvas.create_rectangle(x, y, x + sirka, y + vyska)
```

Všimnite si, že obdĺžniky majú priesvitné vnútro (výplň) a teda je cez obdĺžniky vidieť:

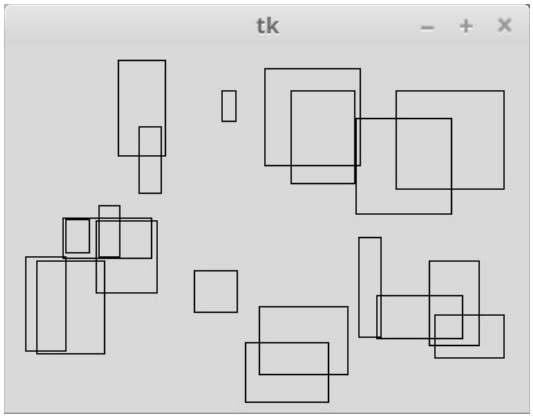


Teraz by sme zvládli zmeniť aj veľkosť obdĺžnikov na nejaké náhodné hodnoty. Preto vo vnútri cyklu vygenerujeme náhodné hodnoty aj pre `sirka` aj `vyska`, napríklad takto:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

for i in range(20):
    sirka = random.randint(10, 80)
```

```
    vyska = random.randint(10, 80)
    x = random.randint(10, 370 - sirka)
    y = random.randint(10, 260 - vyska)
    canvas.create_rectangle(x, y, x + sirka, y + vyska)
```

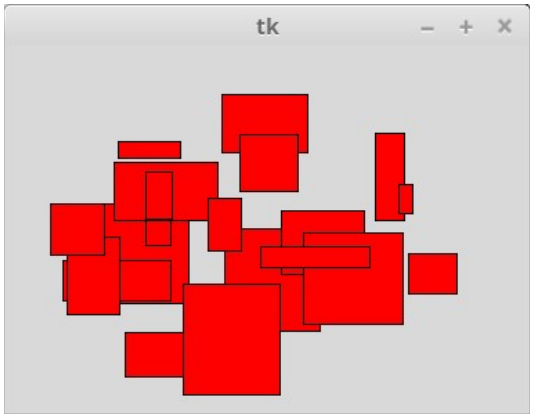
a môžeme dostať:



Veľmi často budem pri kreslení grafických útvarov používať aj nejaké farby. Napríklad, tu by sa veľmi hodilo mať zafarbené vnútro týchto obdĺžnikov. Do predchádzajúceho programu urobíme veľmi malú zmenu: do príkazu `create_rectangle` doplníme jeden parameter, ktorý určí farbu výplne. Zmeňme len tento jeden riadok:

```
canvas.create_rectangle(x, y, x + sirka, y + vyska, fill='red')
```

20 vyfarbených obdĺžnikov teraz vyzerá takto:



Zrejme teraz každý ďalší nakreslený obdĺžnik môže prekrývať (aj úplne zakryť) niektoré už pred tým nakreslené grafické útvary. Aby sme mohli využívať aj iné farby, mali by sme aspoň čiastočne pochopiť princíp, ako `tkinter` používa farby.

Farby v grafických programoch

V prvom rade modul `tkinter` akceptuje najbežnejšie mená farieb v angličtine. Tu môžete vidieť malú ukážku týchto mien aj so zodpovedajúcimi farbami:

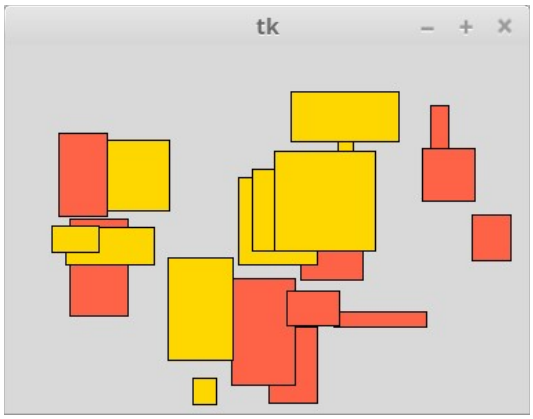
Blue	LightBlue	Cyan	SkyBlue	CornFlowerBlue	DeepSkyBlue	DodgerBlue
RoyalBlue	SlateBlue	SteelBlue	MediumBlue	Navy	Red	SandyBrown
Salmon	Coral	Tomato	Orange	DarkOrange	OrangeRed	IndianRed
Chocolate	Tan	Maroon	Sienna	Brown	SaddleBrown	Pink
Plum	Violet	Orchid	Magenta	Purple	DarkMagenta	Green
PaleGreen	YellowGreen	MediumSeaGreen	LawnGreen	LimeGreen	ForestGreen	DarkGreen
Yellow	Khaki	Gold	Gray	LightGray	Black	White

Tieto mená farieb vo veľkej miere zodpovedajú menám v HTML zápisoch a môžete si o tom niečo pozrieť na stránke HTML Color Names.

Ďalší príklad ilustruje použitie dvoch rôznych farieb. Do premenných `farba1` a `farba2` vložíme mená nejakých dvoch farieb. Potom v cykle použijeme `farba1` na vyfarbenie obdĺžnika a nakoniec navzájom vymeníme obsahy týchto dvoch premenných `farba1` a `farba2`. To znamená, že ďalší obdĺžnik sa vyfarbí už druhou farbou:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

farba1, farba2 = 'tomato', 'gold'
for i in range(20):
    sirka = random.randint(10, 80)
    vyska = random.randint(10, 80)
    x = random.randint(10, 370 - sirka)
    y = random.randint(10, 260 - vyska)
    canvas.create_rectangle(x, y, x + sirka, y + vyska, fill=farba1)
    farba1, farba2 = farba2, farba1
```

Teraz to môže vyzerať takto:



Modul `tkinter` umožňuje zadávanie farieb aj pomocou tzv. **RGB** modelu. Každá farba je namiešaná z troch farebných zložiek: **red** (červená), **green** (zelená) a **blue** (modrá). Intenzita každej zložky je určená číslom medzi 0 a 255: hodnota 0 označuje, že príslušná zložka v danej farbe nie je, hodnota 255 znamená, že sa nachádza v maximálnej intenzite. Čísla medzi tým určujú rôzne odtiene. Ukážme si, akým farbám by zodpovedali niektoré kombinácie **RGB**. Do troch premenných `r, g, b` priradíme tri čísla a zapíšeme zodpovedajúcu farbu:

```
r, g, b = 0, 255, 0       # zelená 'green'
r, g, b = 0, 100, 0       # tmavozelená 'dark green'
r, g, b = 255, 255, 0     # žltá 'yellow'
r, g, b = 190, 190, 190   # šedá 'gray'
r, g, b = 0, 0, 128       # tmavomodrá 'navy'
r, g, b = 255, 255, 255   # biela 'white'
r, g, b = 255, 215, 0     # zlatá 'gold'
r, g, b = 255, 165, 0     # oranžová 'orange'
r, g, b = 255, 192, 203   # ružová 'pink'
```

Takto definované farby sa ale musia zapísať v špeciálnom formáte, ktorý je presne rovnaký ako v HTML. Zapisuje sa ako 7-znakový reťazec v tvare:

```
'#rrggbb'
```

pričom `rr` označuje číslo (od 0 do 255) pre červenú zložku a je zapísané v šestnástkovej (hexadecimálnej) sústave ako dvojciferné číslo, podobne `gg` a `bb` vyjadrujú zelenú a červenú zložku, tiež ako dvojciferné šestnástkové čísla. Napríklad ružovej farbe, ktorá má **rgb** `255, 192, 203`, zodpovedajú šestnástkové zápisy `ff, c0, cb` a preto ružovú farbu môžeme zapísať ako `'#ffc0cb'`. Našťastie nám Python pomôže v prevode ľubovoľných čísel do šestnástkovej sústavy. Môžeme otestovať:

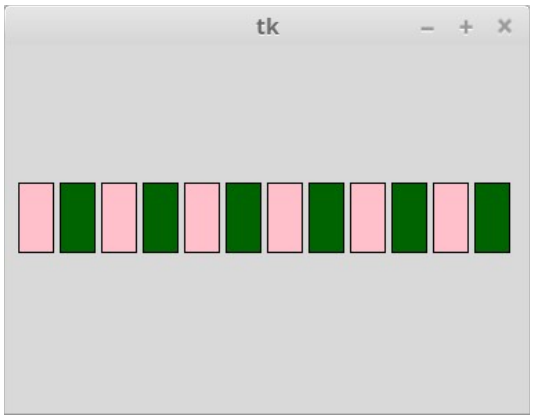
```
>>> r, g, b = 255, 192, 203         # ružová farba
>>> f'#{r:02x}{g:02x}{b:02x}'
'#ffc0cb'
>>> r, g, b = 0, 100, 0             # tmavozelená farba
>>> f'#{r:02x}{g:02x}{b:02x}'
'#006400'
```

Formát `'{r:02x}'` označuje, že zapisujeme hodnotu premennej `r` na šírku `2`, `0` označuje, že číslo bude zľava doplnené nulami a špecifikácia `x` označuje výpis v šestnástkovej sústave.

Použime tieto dve farby na nakreslenie radu obdĺžnikov, ktorým sa striedajú farby výplne:

```
import tkinter

canvas = tkinter.Canvas()
canvas.pack()

y = 100
farba1, farba2 = '#ffc0cb', '#006400'
for x in range(10, 350, 30):
    canvas.create_rectangle(x, y, x + 25, y + 50, fill=farba1)
    farba1, farba2 = farba2, farba1
```

po spustení:



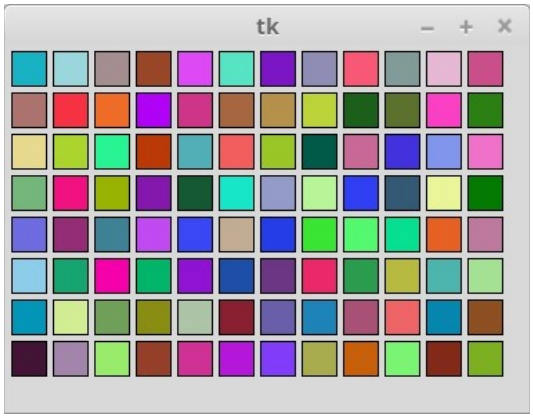
Všimnite si takúto časť programu:

```
r = random.randint(0, 255)          # alebo   r = random.randrange(256)
g = random.randint(0, 255)
b = random.randint(0, 255)
farba = f'#{r:02x}{g:02x}{b:02x}'
```

Najprv sa tu vygenerovali tri náhodné celé čísla z intervalu `<0, 255>` (čo môžu byť tri zložky **RGB**) a z nich sa korektne zostavila šestnástková reprezentácia farby. Využime túto ideu a vygenerujme sieť rôznofarebných štvorcov:

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

for y in range(5, 230, 30):
    for x in range(5, 350, 30):
        r = random.randrange(256)
        g = random.randrange(256)
        b = random.randrange(256)
        farba = f'#{r:02x}{g:02x}{b:02x}'
        canvas.create_rectangle(x, y, x + 25, y + 25, fill=farba)
```

Keď tento program spustíte viackrát, zakaždým dostanete iný výsledok, napríklad:



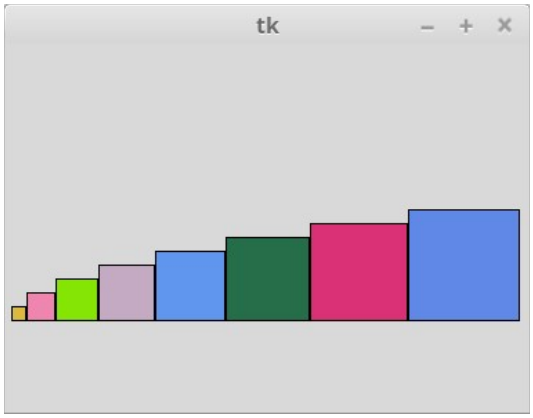
Ak nepotrebujeme generovať tri oddelené zložky **RGB**, ale stačí nám jedna náhodná hodnota, môžeme použiť úspornejší variant generovania náhodnej farby:

```
farba = f'#{random.randrange(256**3):06x}'
```

V tomto prípade sa najprv vygeneruje náhodné číslo z intervalu `<0, 16777215>` (tretia mocnina `256` mínus `1`), toto číslo sa zapíše ako 6-ciferné šestnástkové číslo (3 dvojciferné **RGB** zložky) do znakového reťazca aj so znakom `'#'` na začiatku. Použitie takto generovanej náhodnej farby ukážeme v príklade, v ktorom nakreslíme vedľa seba 8 zväčšujúcich sa štvorcov (so stranami `10`, `20`, `30`, … `80`):

```
import tkinter
import random

canvas = tkinter.Canvas()
canvas.pack()

x, y = 5, 200
for a in range(10, 90, 10):
    farba = f'#{random.randrange(256**3):06x}'
    canvas.create_rectangle(x, y, x + a, y - a, fill=farba)
    x += a+1
```

Každý nakreslený štvorec má inú náhodnú farbu. Všimnite si, že `(x, y)` je v tomto prípade ľavý dolný vrchol a jemu protiľahlý (pravý horný) má súradnice `(x+a, y-a)`, teda veľkosť strany štvorca je zrejme `a`:



Naposledy aktualizované 24. mar. 2020.
Vytvorené pomocou Sphinx 2.4.4.