03 Vlastné funkcie bez parametrov a bez návratovej hodnoty

Pracovný list

Zapojenie

|  |  |
| --- | --- |
| **Úloha 1** | Po viacnásobnom otočení maliarskeho valčeka sa na stenu odtlačil uvedený obrázok s ľudovým motívom:  C:\Users\Šnajder\Desktop\03_cicmiansky_vzor.png  Vyznačte v tomto obrázku vzor, ktorý je nanesený na valčeku. Prípadne tento vzor nakreslite vedľa obrázku.  Uveďte koľkokrát sa otočil valček so vzorom, ktorý na stenu postupne odtlačil uvedený obrázok: .....-krát |

|  |  |
| --- | --- |
| **Úloha 2**  Riešte podľa pokynov učiteľa | Navrhnite vzor pre pečiatku (pečiatky), pomocou ktorej (ktorých) opečiatkujete celý tvar panáka na obrázku.  C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\03_panak2.png      C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\mriezka_stvorcova.png      C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\mriezka_stvorcova.png      C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\mriezka_stvorcova.png  Uveďte, či stačí na opečiatkovanie celého tvaru panáka pečiatka s jedným vzorom: áno – nie  Ak ste prišli aj iné riešenia tejto úlohy, zakreslite ich do prázdnych štvorcových mriežok vedľa tej s panákom.  Prediskutujte so spolužiakmi svoje riešenia z pohľadu veľkosti a zložitosti pečiatky a počtu jej opečiatkovaní pri tvorbe panáka. |

Skúmanie

|  |  |
| --- | --- |
| **Úloha 3** | Na uvedenom obrázku nájdite a (orámčekovaním) vyznačte vzor, pomocou ktorého vieme zostaviť daný obrázok.  C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\dvojgulka_obrazok5x.png  V programe na vykreslenie daného obrázka (orámčekovaním) vyznačte časti, ktoré vykresľujú nájdený vzor.  **import** turtle  pero = turtle.Turtle() tabula = turtle.Screen() pero.penup()  pero.dot(50, **"black"**) pero.forward(50) pero.dot(50, **"lightgray"**) pero.forward(50) pero.dot(50, **"black"**) pero.forward(50) pero.dot(50, **"lightgray"**) pero.forward(50) pero.right(90) pero.dot(50, **"black"**) pero.forward(50) pero.dot(50, **"lightgray"**) pero.forward(50)  tabula.mainloop() |

|  |  |  |  |
| --- | --- | --- | --- |
| **Úloha 4** | Preskúmajte uvedený program **03\_04\_corobim.py**. (Poznámka: Na konci riadku 4 s novým príkazom vzor() je uvedená **dvojbodka**. Riadky 5 až 9 **sú odsadené** dohodnutým počtom medzier, napr. 4.)   |  |  | | --- | --- | | 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19  20 | **import** turtle  **def** vzor():  *# zapis postupu vykreslenia vzoru = definovanie funkcie vzor()* pero.dot(50, **"black"**)  pero.forward(50)  pero.dot(50, **"lightgray"**)  pero.forward(50)   tabula = turtle.Screen() pero = turtle.Turtle() pero.penup()  vzor() *# vykonanie postupu vykreslenia vzoru = volanie funkcie vzor()* pero.right(90) *# otocenie sa o 90 stupnov vpravo* vzor() *# vykonanie postupu vykreslenia vzoru = volanie funkcie vzor()* tabula.mainloop() |  1. Popíšte alebo nakreslite obrázok, ktorý vykreslí tento program: ............................................................... 2. Vysvetlite, aký je rozdiel medzi príkazom def vzor(): na riadkoch 04 až 09 a príkazom vzor() na riadkoch 16 a 18: ................................................................................................................................... 3. Uveďte, akú zmenu treba urobiť v uvedenom programe, aby vykreslil rovnaký obrázok ako v úlohe 3: ..................................................................................................................................................................... |

Vysvetlenie

Prediskutujte a vysvetlite:

1. V čom je upravené riešenie úlohy 4 využívajúce vlastné funkcie lepšie, či horšie ako riešenie úlohy 3 bez vlastných funkcií?
2. V ktorej časti programu vytvárame vlastnú funkciu a v ktorej ju používame? Mohli by sme najprv použiť funkciu a až potom ju vytvoriť?
3. Koľkokrát vytvárame vlastnú funkciu a koľkokrát ju môžeme použiť?
4. Z akých častí pozostáva zápis vytvorenia vlastnej funkcie?
5. Kde končí zápis vytvorenia vlastnej funkcie?
6. Ako sa zmení vykonávanie programu, ak v ňom neuvedieme komentáre?
7. Aký význam pre programátora majú komentáre? Nie je to zbytočná strata času pri ich uvádzaní?

Rozpracovanie

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Úloha 5**  Riešte podľa pokynov učiteľa | Vytvorte programy na vykreslenie uvedených obrázkov, v ktorých použijete vlastnú funkciu pre vykreslenie vzoru.   |  |  |  | | --- | --- | --- | | **A**  C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\stvorgulka_obrazok1x.png | **B**  C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\stvorgulka_obrazok2x.png | **C**  C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\stvorgulka_obrazok3x.png | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Úloha 6** | Preskúmajte, čo robí uvedený program **03\_06\_krabicka.py**.   |  |  | | --- | --- | | 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20  21  22  23 | **import** turtle   **def** obrazok():  pero.fillcolor(**"green"**)  pero.begin\_fill()  pero.forward(100)  pero.left(120)  pero.forward(50)  pero.left(60)  pero.forward(50)  pero.left(60)  pero.forward(50)  pero.left(120)  pero.end\_fill()   tabula = turtle.Screen() pero = turtle.Turtle()  obrazok()  tabula.mainloop() |  1. C:\Users\Ľubomír\AppData\Local\Microsoft\Windows\INetCache\Content.Word\03_06_krabicka.pngVysvetlite význam príkazov fillcolor(), begin\_fill(), end\_fill():  ..................................................................................................................................................................... 2. Uvedený program upravte tak, aby vykreslil nasledovný obrázok: |
| **Úloha 7**  Riešte podľa pokynov učiteľa | Vytvorte program na vykreslenie uvedeného obrázku.  03_stromy4 |

Hodnotenie

Sebahodnotiaci test

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Úloha 8** | Zakrúžkovaním vyberte skupinu (skupiny) príkazov, pomocou ktorej (ktorých) sa vykreslí uvedený obrázok.  03_belgicko_rumunsky_nahrdelnik2   |  |  |  | | --- | --- | --- | | **Skupina príkazov A** | **Skupina príkazov B** | **Skupina príkazov C** | | **def** vzor():  pero.dot(40, **"black"**)  pero.forward(40)  pero.dot(40, **"yellow"**)  pero.forward(40)  pero.dot(40, **"red"**)  pero.forward(40)  pero.dot(40, **"blue"**)  pero.forward(40)  pero.dot(40, **"yellow"**)  pero.forward(40)  pero.dot(40, **"red"**)  pero.forward(40)  vzor() vzor()  vzor() | **def** belgicko():  pero.dot(40, **"black"**)  pero.forward(40)  pero.dot(40, **"yellow"**)  pero.forward(40)  pero.dot(40, **"red"**)  pero.forward(40)  **def** rumunsko():  pero.dot(40, **"blue"**)  pero.forward(40)  pero.dot(40, **"yellow"**)  pero.forward(40)  pero.dot(40, **"red"**)  pero.forward(40)  belgicko() rumunsko() belgicko() rumunsko() | **def** dvojgulka():  pero.dot(40, **"yellow"**)  pero.forward(40)  pero.dot(40, **"red"**)  pero.forward(40)  **def** sestgulka():  pero.dot(40, **"black"**)  pero.forward(40)  dvojgulka()  pero.dot(40, **"blue"**)  pero.forward(40)  dvojgulka()  sestgulka() sestgulka() |   Skupinu (skupiny) príkazov s chybným riešením opravte. Stručne okomentujte uvedené riešenia úlohy:  ...................................................................................................................................................................................  ...................................................................................................................................................................................  ................................................................................................................................................................................... |

|  |  |
| --- | --- |
| Vedomosti v kocke |  |
| Pri riešení problémov môžeme využívať rôzne stratégie, napr. **dekompozíciu** (rozdelíme problém na menšie podproblémy), **hľadanie vzoru** (nájdenie a využitie opakujúceho sa vzoru), **vykreslenie si náčrtu obrázku**, atď. Pri programovaní riešení problémov využívajúcich tieto stratégie sa používajú **vlastné** **funkcie**.  Pri maľovaní obrázkov s opakujúcimi sa vzorom (vzormi) môžeme využiť pečiatku (pečiatky) s daným vzorom (danými vzormi). Postupujeme pri tom tak, že najprv si vyrobíme pečiatky a potom ich namočené odtláčame na maľovaciu plochu. Pri programovaní je to takmer rovnaké. Najprv vytvoríme (=**definujeme**) vlastnú funkciu, ktorá popisuje riešenie podproblému a potom ju použijeme (=**voláme**) pri riešení problému.  V nasledovnej tabuľke uvádzame schematické zápisy dvoch príkladov vlastných funkcií – na vykreslenie obrázku s jedným, resp. dvoma vzormi (pečiatkami).   |  |  | | --- | --- | | Program na vykreslenie obrázku s 1 vzorom | Program na vykreslenie obrázku s 2 vzormi | | **def pečiatka**():  *'''  príkazy na vykreslenie vzoru,  z ktorého pozostáva obrázok  '''*  **def** presun1():  *# príkazy na 1. presun grafického pera*  **def** presun2():  *# príkazy na 2. presun grafického pera*  **pečiatka**() presun1() **pečiatka**() presun2() **pečiatka**() | **def pečiatka1**():  *# príkazy na vykreslenie vzoru 1*  **def pečiatka2**():  *# príkazy na vykreslenie vzoru 2*  **def** presun1():  *# príkazy na 1. presun grafického pera*  **def** presun2():  *# príkazy na 2. presun grafického pera*  **pečiatka1**() presun1() **pečiatka2**() presun2() **pečiatka1**() |   Syntax **definície funkcie** obsahuje kľúčové slovo **def** s priliehavým názvom funkcie za ktorým je dvojica okrúhlych zátvoriek nasledovaná dvojbodkou. Tento prvý riadok funkcie označujeme aj ako hlavička definície funkcie. Po nej nasleduje telo definície funkcie, ktoré obsahuje postupnosť príkazov, ktoré sú v porovnaní s hlavičkou odsadené od ľavého okraja rovnakým počtom medzier (štandardne 4). Ak na niektorom z riadkov nie sú príkazy už odsadené, tieto príkazy nepatria už do tela definície funkcie.  Syntax **volania funkcie** obsahuje samotný názov funkcie nasledovaný dvojicou okrúhlych zátvoriek.  V niektorých prípadoch môžu vzory obsahovať v sebe ešte ďalšie vzory. Napr. text LISTLIPALIST, obsahuje dva vzory LIST a LIPA, pričom každý z nich obsahuje ďalší vzor LI. Program na vypísanie textu by mohol mať nasledovný schematický zápis:   |  | | --- | | Program na vypísanie textu s vnorenými vzormi | | **def li**():  l() *# vypise L*  i() *# vypise I*  **def list**():  li() *# vypise LI*  s() *# vypise S*  t() *# vypise T*  **def lipa**():  li() *# vypise LI*  p() *# vypise P*  a() *# vypise A*  **list**() *# vypise LIST* **lipa**() *# vypise LIPA* **list**() *# vypise LIST* |   **Typickými chybami** začínajúcich programátorov v jazyku Python je> zabúdanie písania okrúhlych zátvoriek za názvom funkcie pri jej definovaní či volaní; zabúdanie písania dvojbodky na konci hlavičky definície funkcie; nedôsledné odsadzovanie príkazov v tele definície funkcie; považovanie prázdneho riadka za ukončenie tela funkcie; používanie nepriliehavých názvov funkcií; nepoužívanie komentárov.  **Používanie komentárov** má **význam** pre sprehľadnenie zápisov funkcií programu:   * v procese návrhu funkcie (ako poznámky, čo treba urobiť – „to do“), * na zdokumentovanie funkcionality funkcie (ako komentáru, čo robí daná funkcia), * na prípadné znefunkčnenie časti kódu.   **Použitia vlastných funkcií** prináša nasledovne **výhody**:   * písanie prehľadnejšieho kódu odpovedajúceho riešenému problému (riešenie každého podproblému bude reprezentovať jedna funkcia), * lepšia lokalizácia a opravovanie chýb v programe, * skrátenie kódu viacnásobnou znovupoužiteľnosťou už raz napísaného programového kódu, * deľba práce medzi viacerých programátorov (každý z nich rieši len vybrané podproblémy). | |