
Hodnocení soutěžních úloh

Kategorie mládež

Soutěž dětí a mládeže v programování – 22. ročník
Krajské kolo 2007/2008
18. a 19. dubna 2008

Olympijské logo

Koeficient 1

Vytvořte program, který zobrazí logo olympijských her, tedy pět vzájemně propletených kruhů dle níže uvedeného obrázku. Není nutné, aby místa, kde se kruhy překrývají, byla zobrazena přesně podle vzoru, ale čím více se váš výtvar bude vzoru podobat, tím větší bodové ohodnocení získáte.

Dostanete-li zadání jen jako černobílý výtisk, poradíme vám barvy jednotlivých kruhů – horní řada zleva: modrá, černá, červená; dolní řada zleva: žlutá, zelená.



Hodnocení

funkčnost	2 body	zobrazení pěti mezikruží na správných pozicích
	2 body	barvy odpovídají vzoru
	2 body	správné zobrazení překryvů dle vzorového loga (žádné mezikruží nesmí být opticky navrchu)
efektivita	1 bod	tloušťka mezikruží je větší než 1 pixel
	1 bod	logo není nepřiměřeně malé (=je větší než cca 100px na šířku)
	1 bod	logo není zubaté (=nejedná se o nekvalitní zvětšení malého obrázku)
dokumentace	1 bod	komentáře, přehlednost, výstižné názvy proměnných, ...

Sčítání čísel zapsaných římskými číslicemi

Koeficient 1

Náš starý známý strýček Pompo má kromě jiných zálib i zálibu v historii, protože ale jeho znalosti historie nejsou ještě tak dobré, potřeboval by od vás pomoci. Radost by mu udělal program, který dokáže sčítat čísla zapsaná římskými číslicemi.

Napište program, který dokáže sčítat po sobě jdoucí čísla zapsaná římskými číslicemi. Čísla jsou uložena v textovém souboru, který si uživatel může vybrat. Na každé řádce souboru je pouze jedno číslo zapsané římskými číslicemi. Váš program tento soubor načte a zobrazí celkový součet zapsaný jak arabskými tak římskými číslicemi.

Přestože zápisů čísel římskými číslicemi existuje několik, budeme považovat za správný pouze ten následující.

Číslo 5000 bude pro nás to nejvyšší možné. Římané sice uměli počítat i s většími čísly, ale Pompovi to bude stačit. Římané zapisovali svá čísla pomocí písmen, která vždy symbolizují jeden přesný počet. Zápis čísla začíná vždy písmenem s nejvyšší hodnotou, k tomuto písmenu se pak přiřazují písmena další, která jeho hodnotu zvyšují, všimněte si v následující tabulce čísel 4, 9, 14, 17, 60, 112, atd. Stačí tedy číst čísla zleva doprava a hodnoty jednotlivých písmen sčítat, tak nám vyjde výsledné číslo. Program bude kontrolovat zadané číslo a v případě chybného zápisu vyhlásí chybu.

Použité symboly jsou: I = 1, V = 5, X = 10, L = 50, C = 100, D = 500, M = 1000.

Příklady zápisu:

1 = I	6 = VI	11 = XI	16 = XVI	21 = XXI	40 = XXXX
2 = II	7 = VII	12 = XII	17 = XVII	24 = XXIIII	43 = XXXXIII
3 = III	8 = VIII	13 = XIII	18 = XVIII	26 = XXVI	45 = XXXXV
4 = IIII	9 = VIIII	14 = XIIII	19 = XVIIII	29 = XXVIIII	49 = XXXXVIIII
5 = V	10 = X	15 = XV	20 = XX	30 = XXX	50 = L
55 = LV	67 = LXVII	103 = CIII	129 = CXXVIIII	400 = CCCC	
59 = LVIIII	70 = LXX	112 = CXII	130 = CXXX	450 = CCCCL	
60 = LX	80 = LXXX	115 = CXV	150 = CL	462 = CCCCLXII	
63 = LXIII	90 = LXXXX	120 = CXX	200 = CC	500 = D	
65 = LXV	100 = C	124 = CXXIIII	233 = CCXXXIII	501 = DI	
574 = DLXXIIII	700 = DCC	900 = DCCCC	1300 = MCCC		
599 = DXCVIIII	751 = DCCLI	937 = DCCCCXXXVII	1631 = MDCXXXI		
600 = DC	790 = DCCLXXXX	953 = DCCCCLIII	2200 = MMCC		
650 = DCL	800 = DCCC	978 = DCCCCLXXVIII	2750 = MMDCCCL		
681 = DCLXXXI	825 = DCCCXXV	1000 = M	3000 = MMM		

Hodnocení

funkčnost	0.5 bodu	Program umožní vybrat vstupní soubor
	1 bod	Pro testovací soubor roma01.txt je vrácen součet 45 (XXXXV)
	1 bod	Pro testovací soubor roma02.txt je vrácen součet 1666 (MDCLXVI)
	1 bod	Pro testovací soubor roma03.txt je vrácen součet 2273 (MMCCCLXXIII)
	1 bod	Pro testovací soubor roma04.txt je vrácen součet 2333 (MCCCXXXIII)
	1 bod	Pro testovací soubor roma05.txt je vrácen součet 1320 (MCCCXX)
	1 bod	Pro testovací soubor roma06.txt je vrácen součet 4500 (MMMMD)
	1 bod	Pro testovací soubor roma07.txt je oznámena chyba
	1 bod	Pro testovací soubor roma08.txt je oznámena chyba
	0.5 bodu	Pro testovací soubor roma09.txt je oznámena chyba (součet je větší než 5000)
dokumentace	1 bod	komentáře, přehlednost, výstižné názvy proměnných, ...

Interpret ŠVG

Koeficient 3

Napište program, který dokáže na obrazovce vykreslovat obrázky ve formátu ŠVG (Šmrncovní Vobrázky a Grafika). V programu postupně naimplementujte následující funkce.

1. Výběr souboru s obrázkem k vykreslení.
2. Vykreslení obrázku přes celou plochu okna programu.
3. Zvětšování/zmenšování obrázku.
4. Posouvání obrázku.

Formát ŠVG

ŠVG je vektorový grafický formát. To znamená, že obrázek je složen ze základních geometrických objektů, u kterých lze určovat jejich pozici, rozměry a další vlastnosti jako barvu a výplň.

Formát ŠVG používá syntaxi XML. Celý obrázek je uzavřen v elementu `svg`. Uvnitř něj se pak nachází elementy popisující jednotlivé objekty, ze kterých se skládá celý obrázek. Obrázek se vykresluje vždy na bílé pozadí. Jednotlivé objekty, ze kterých se obrázek skládá, se vykreslují ve stejném pořadí v jakém jsou uvedeny v souboru.

Každý obrázek se vykresluje na plochu, jejíž rozměry jsou určeny pomocí atributů `width` (šířka) a `height` (výška). Rozměry mohou být určeny jako libovolné kladné desetinné číslo. Váš program musí podle potřeby tyto „uživatelské“ souřadnice přepočítávat tak, aby se obrázek vešel na obrazovku. Proporce obrázku přitom musí být zachovány.

Příklad 1. Ukázka kostry ŠVG obrázku

```
<?xml version="1.0" encoding="utf-8"?>
<svg width="320" height="200">
  ...
</svg>
```

Obrázek se může skládat z následujících objektů:

čára

```
<line x1="..." y1="..." x2="..." y2="..." />
```

Čára je určena souřadnicemi svého počátku (x_1, y_1) a konce (x_2, y_2).

obdélník

```
<rect x1="..." y1="..." x2="..." y2="..." />
```

Obdélník je určen souřadnicemi diagonálně protilehlých rohů se souřadnicemi (x_1, y_1) a (x_2, y_2).

obdélník s oblými rohy

```
<rect x1="..." y1="..." x2="..." y2="..." rx="..." ry="..." />
```

Obdélník je určen souřadnicemi diagonálně protilehlých rohů se souřadnicemi (x_1, y_1) a (x_2, y_2) . Jeho rohy nejsou pravouhlé, ale jsou nahrazeny odpovídající čtvrtinovou výsečí elipsy, jejíž poloměr v osách X a Y je určen atributy r_x a r_y .

kružnice

```
<circle cx="..." cy="..." r="..." />
```

Kružnice je určena souřadnicemi svého středu (cx, cy) a svým poloměrem (r).

elipsa

```
<ellipse cx="..." cy="..." rx="..." ry="..." />
```

Elipsa je určena souřadnicemi svého středu (cx, cy) a poloměrem hlavní (r_y) a vedlejší (r_x) osy.

U každého obrázku je možné pomocí následujících atributů určit barvu obrysu, sílu obrysu a barvu výplně:

color Barva obrysu objektu. Barva se určuje podobně jako v jazyce HTML pomocí zápisu $\#RRGGBB$, kde RR je intenzita červené složky, GG je intenzita zelené složky a BB je intenzita modré složky v barevném modelu RGB zapsané v šestnáctkové soustavě.

Příklady některých barev: $\#000000$ (černá), $\#FFFFFF$ (bílá), $\#FF0000$ (červená), $\#FFFF00$ (žlutá).

Pokud barva není určena, předpokládá se černá.

fill Barva výplně objektu (netýká se čár). Barva se určuje stejně jako pro atribut `color`.

Pokud barva výplně není určena, předpokládá se bílá.

stroke Síla čáry obrysu zadaná jako destinné číslo. Pokud není síla určena, předpokládá se 1 (v „uživatelských“ souřadnicích).

Objekty, ze kterých se obrázek skládá, je možné sdružovat pomocí elementu `g` a na něm nastavit pro všechny sdružené objekty barvu obrysu, sílu obrysu a barvu výplně. Pokud je však některý z těchto parametrů nastaven přímo na daném grafickém objektu, má přednost před definicí zděděnou z elementu `g`. Elementy `g` je možné do sebe navzájem zanořovat.

Příklad 2. Ukázka jednoduchého obrázku a jeho zobrazení

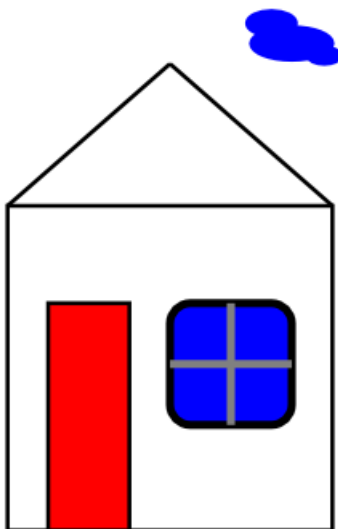
```
<?xml version="1.0" encoding="utf-8"?>
<svg width="100" height="150">
  <!-- Hlavně st domečku -->
  <rect x1="10" y1="10" x2="90" y2="90" color="#000000"/>

  <!-- Střecha -->
  <line x1="10" y1="90" x2="50" y2="125" color="#000000"/>
  <line x1="90" y1="90" x2="50" y2="125" color="#000000"/>

  <!-- Dveře -->
  <rect x1="20" y1="10" x2="40" y2="66" color="#000000" fill="#FF0000"/>

  <!-- Okno -->
  <rect x1="50" y1="36" x2="80" y2="66" rx="5" ry="5" color="#000000" fill="#0000FF"
stroke="2"/>
  <line x1="65" y1="36" x2="65" y2="66" stroke="2" color="#808080"/>
  <line x1="50" y1="51" x2="80" y2="51" stroke="2" color="#808080"/>

  <!-- Mrštek -->
  <g color="#0000FF" fill="#0000FF">
    <ellipse cx="80" cy="130" rx="10" ry="4"/>
    <ellipse cx="75" cy="135" rx="6" ry="3"/>
    <ellipse cx="88" cy="127" rx="4" ry="2"/>
  </g>
</svg>
```



Další ukázky obrázků najdete v adresáři se zadáním úloh.

Hodnocení

funkčnost	0.5 bodu	Lze vybrat název souboru k vykreslení
	2.5 bodu	Podpora základních tvarů – za každý správně vykreslený objekt (tvar) v souboru 01.xml přidělit 0.5 bodu
	1 bod	Podpora přepočtu uživatelských souřadnice – obrázek 02.xml se celý vykreslí přes celé okno programu
	0.5 bodu	Podpora color – obrázek 03.xml se vykreslí se správnými barvami
	0.5 bodu	Podpora fill – obrázek 04.xml se vykreslí se správnou barvou výplně
	1 bod	Podpora stroke – obrázek 05.xml se vykreslí se správnou tloušťkou čar (čára je tenká, obdélník má tlustý obrys a okolo elipsy obrys není vůbec). Body lze přidělovat po třetinách.
	1 bod	Podpora dědění vlastností – obrázek 06.xml se správně vykreslí
	0.5 bodu	Obrázek lze zmenšovat
	0.5 bodu	Obrázek lze zvětšovat
	1 bod	Obrázkem lze plynule rolovat
efektivita	0.5 bodu	pro čtení SVG je použit parser XML
dokumentace	0.5 bodu	komentáře, přehlednost, výstižné názvy proměnných, ...

Pro náhledy testovacích souborů můžete použít buď vyrendrované obrázky (01.png až 05.png). Druhou možností je na začátek SVG obrázku přidat následující instrukci:

```
<?xml version="1.0" encoding="utf-8"?>
<?xml-stylesheet href="svg2svg.xsl" type="text/xsl"?>
<svg ...
```

SVG obrázek pak stačí otevřít přímo v prohlížeči s podporou XSLT a SVG (např. Opera nebo Firefox) a dojde k jeho vykreslení.

Loupežníci

Koeficient 3

Dvě známé loupežnické firmy Rumcajs, s radostí okrádám, a Lotrando, absolutní straka, spojily své síly a společně uloupily velkou kořist. Kořist se skládá z N předmětů, každý předmět má nějakou cenu c_i . Nyní se loupežníci musí o lup rozdělít. Protože ale chtějí i nadále spolupracovat, musí se rozdělít rovným dílem, tedy tak, aby oba dostali přesně polovinu celého lupu. Nemohou se ale dohodnout a už na sebe začínají vytahovat bambitky. Pomůžete jim?

Vaším cílem je napsat program, který na vstupu dostane číslo N , dále N cen jednotlivých předmětů lupu c_1, \dots, c_N a zjistí, zda je možné tyto předměty rozdělít do dvou skupin tak, aby se součet cen předmětů v jedné skupině rovnal součtu cen předmětů v druhé skupině. Žádný z předmětů už není možné dělit, musí být celý buď v jedné nebo druhé skupině. Pokud lze předměty rozdělít, program vypíše, jaké předměty jsou v jedné z těchto dvou skupin. Pokud řešení existuje několik, vypsát můžete libovolné z nich.

Pozor, musíte přesně dodržet popsany formát vstupu a výstupu, protože úloha se bude vyhodnocovat automaticky. Pokud popsany formát nedodržíte, neobdržíte žádné body.

Popis vstupu

Vstup je uložen v souboru `lup.in` v aktuálním adresáři. Na první řádce se nachází jediné číslo $1 \leq N \leq 500$. Na druhé řádce se nachází N přirozených čísel c_1, \dots, c_N – ceny předmětů lupu. Tato čísla jsou oddělena jednou mezerou, každé je větší rovno jedné a součet těchto N čísel je menší rovno 500000.

Řádky mohou být odděleny pomocí znaků CR, LF nebo CR+LF.

Popis výstupu

Výstup musíte vypsát do souboru `lup.out`. Pokud dané předměty nejdou rozdělít na dvě stejně cenné skupiny, musí výstupní soubor obsahovat jedinou řádku, na které se nachází číslo 0. Pokud dané předměty jdou rozdělít, musíte vypsát předměty z jedné skupiny. Výstupní soubor pak musí obsahovat dvě řádky. Na první musí být jediné číslo K – počet předmětů v jedné ze skupin. Na druhé řádce musí být mezerou oddělených K čísel – čísla předmětů ve skupině (počítáno od jedničky). Tato čísla předmětů mohou být vypsána v libovolném pořadí, ale žádné se samozřejmě nesmí opakovat.

Příklad 3.

Pro následující vstupní soubor `lup.in`

```
5
2 5 4 7 2
```

předměty rozdělít nelze. Obsah souboru `lup.out` proto bude

```
0
```


Příklad 4.

Pro vstupní soubor `lup.in`

```
6
7 1 2 3 4 5
```

existují čtyři správná rozdělení. Vypsát můžete libovolné z nich, předměty mohou být vypsány v libovolném pořadí. Soubor `lup.out` může mít jeden z následujících obsahů

```
2
1 5 (7+4=11)
```

```
3
1 2 4 (7+1+3=11)
```

```
3
3 5 6 (2+4+5=11)
```

```
4
2 3 4 6 (1+2+3+5=11)
```

Hodnocení

Za tuto úlohu můžete udělit až deset bodů. Až jeden bod udělte za úpravu zdrojového kódu (pochopitelnost, pojmenování proměnných, odsazení, ...). Zbýlých devět bodů udělte za funkčnost a efektivitu dle následující tabulky:

funkčnost a efektivita	1.5 bodu	program správně vyřeší vstupy m1a a m1b
	1.5 bodu	program správně vyřeší vstupy m2a a m2b
	1.5 bodu	program správně vyřeší vstupy m3a a m3b
	1.5 bodu	program správně vyřeší vstupy m4a a m4b
	1.5 bodu	program správně vyřeší vstupy m5a a m5b
	1.5 bodu	program správně vyřeší vstupy m6a a m6b
dokumentace	1 bod	komentáře, přehlednost, výstižné názvy proměnných, ...

Pokud program vyřeší pouze jeden vstup ze dvojice, nepřiděluje žádné body. Na každý vstup má program 5 vteřin procesorového času. Pokud běží déle, je to stejné, jako by vstup nevyřešil (ohledně měření času viz dále). Testovací vstupy nesmíte dát soutěžícím v průběhu soutěže ani po jejím konci k dispozici.

Protože správných řešení je několik, správnost řešení je třeba testovat dodaným programem. Pro windows jsou tyto programy v archivu `loupeznici_win.zip`, pro linux je v archivu `loupeznici_linux.tgz`. Tento archiv obsahuje také všechny testovací vstupy a zdrojové kódy obou testovacích programů. Programy pro testování řešení jsou dva: `otestuj` a `spust_a_otestuj`. Testovací programy fungují pouze s dodanými testovacími vstupy.

Spust_a_otestuj: testování program, který i spouští řešení a měří čas. Nakopírujte vstupní data a program `spust_a_otestuj` do adresáře, kde je spustitelný soubor `reseni` od soutěže. Programu `spust_a_otestuj` dejte jako první parametr název vstupního souboru bez koncovky a jako druhý parametr příkaz, kterým se spustí řešení od soutěže (tj. třeba `spust_a_otestuj.exe m4a loupeznici.exe` na windows a `./spust_a_otestuj z5b ./loupeznici` na linuxu). Program `spust_a_otestuj` zkopíruje vstupní soubor do souboru `lup.in`, smaže `lup.out`, pokud existuje, dále spustí řešení, změří mu čas (a ukončí ho, pokud běží déle než 5 sekund) a nakonec otestuje, zda soubor `lup.out`, který mělo řešení vytvořit, obsahuje správný výsledek. Výsledek vypíše na konzoli.

Otestuj: testovací program, který pouze testuje správnost řešení. Pokud by vám předchozí způsob z nějakého důvodu nevyhovoval, máte k dispozici ještě testovací program `otestuj`. Tento program pouze načte soubory `lup.in` a `lup.out` z aktuálního adresáře a zjistí, zda je `lup.out` správné řešení `lup.in`. Celé testování pak musí vypadat tak, že vy zkopírujete vstupní soubor do souboru `lup.in`, odstraníte soubor `lup.out`, pokud existuje, dále ručně spustíte řešení soutěže, změříte mu čas, a pokud běželo méně než 5 sekund a vytvořilo soubor `lup.out`, spustíte `otestuj`, abyste zjistili, zda bylo vyprodukované řešení správné.