Otvory

Termín odevzdání:	**12.11.2017 23:59:59**
Hodnocení:	**2.7291**
Max. hodnocení:	**3.0000** (bez bonusů)
Odevzdaná řešení:	14 / 20 Volné pokusy + 10 Penalizované pokusy (-10 % penalizace za každé odevzdání)
Nápovědy:	2 / 2 Volné nápovědy + 2 Penalizované nápovědy (-10 % penalizace za každou nápovědu)

Úkolem je vytvořit program, který dokáže určit, zda daný objekt (kvádr) projde definovaným otvorem.

V přepážce jsou 3 otvory: čtvercový, kruhový a otvor ve tvaru rovnostranného trojúhelníku. Otvory mají stejný rozměr (strana čtverce, průměr kruhu, strana rovnostranného trojúhelníku), tento rozměr je zadán na vstupu. Chceme rozhodnout, zda kvádr o zadaných rozměrech projde jednotlivými otvory. Velikost kvádru je zadaná jako trojice čísel na vstupu. Kvádr otvorem projde, jestliže jeho rozměr je menší nebo roven rozměru otvoru, kvádr lze libovolně otáčet.

Vstupem programu jsou 4 desetinná čísla: rozměr otvoru a 3 rozměry kvádru. Všechna čísla musí být kladná.

Výstupem programu je rozhodnutí, zda kvádr projde jednotlivými otvory, formát výstupu je v ukázce.

Pokud je vstup neplatný, program to musí detekovat a zobrazit chybové hlášení. Chybové hlášení zobrazujte na standardní výstup (ne na chybový výstup). Za chybu považujte:

- nečíselné zadání libovolného rozměru,
- rozměr je záporný nebo nulový,
- nějaký z rozměrů nebyl zadán.

Ukázka práce programu:

```
Velikost otvoru:
5
Velikost kvadru:
1.5 2.5 3.5
Trojuhelnik: ano
Kruh: ano
Ctverec: ano
```

```
Velikost otvoru:
3
Velikost kvadru:
100 1.45 1.45
Trojuhelnik: ne
Kruh: ano
Ctverec: ano
```

```
Velikost otvoru:
4
Velikost kvadru:
2.8 6 2.9
Trojuhelnik: ne
Kruh: ne
Ctverec: ano
```

```
Velikost otvoru:
2
Velikost kvadru:
1 2.5 3
Trojuhelnik: ne
Kruh: ne
Ctverec: ne
```

```
Velikost otvoru:
4
Velikost kvadru:
```

5 4 3
Trojuhelnik: ne
Kruh: ne
Ctverec: ano

Velikost otvoru:
28.781
Velikost kvadru:
5.019 43.814 28.340
Trojuhelnik: ne
Kruh: ano
Ctverec: ano

Velikost otvoru:
8.5
Velikost kvadru:
4 15 3.897
Trojuhelnik: ano
Kruh: ano
Ctverec: ano

Velikost otvoru:
10
Velikost kvadru:
20 11 2
Trojuhelnik: ne
Kruh: ne
Ctverec: ano

Velikost otvoru:
2
Velikost kvadru:
7 5 -3
Nespravny vstup.

Velikost otvoru:
xyz
Nespravny vstup.

Poznámky:

- Ukázkové běhy zachycují očekávané výpisy Vašeho programu (tučné písmo) a vstupy zadané uživatelem (základní písmo). Zvýraznění tučným písmem je použité pouze zde na stránce zadání, aby byl výpis lépe čitelný. Váš program má za úkol pouze zobrazit text bez zvýrazňování (bez HTML markupu).
- Znak odřádkování (\n) je i za poslední řádkou výstupu (i za případným chybovým hlášením).
- Pro reprezentaci hodnot použijte desetinná čísla typu `double`. Nepoužívejte typ `float`, jeho přesnost nemusí být dostatečná.
- Úlohu lze vyřešit bez použití funkcí. Pokud ale správně použijete funkce, bude program přehlednější a bude se snáze ladit.
- Číselné vstupní hodnoty jsou zadávané tak, aby se vešly do rozsahu datového typu `double`. Referenční řešení si vystačí s číselnými typy `double` a `int`.
- Při programování si dejte pozor na přesnou podobu výpisů. Výstup Vašeho programu kontroluje stroj, který požaduje přesnou shodu výstupů Vašeho programu s výstupy referenčními. Za chybu je považováno, pokud se výpis liší. I chybějící nebo přebývající mezera/odřádkování je považováno za chybu. Abyste tyto problémy rychle vyloučili, použijte přiložený archiv se sadou vstupních a očekávaných výstupních dat. Podívejte se na videotutoriál (edux -> výuková videa), jak testovací data použít a jak testování zautomatizovat.

- Váš program bude spouštěn v omezeném testovacím prostředí. Je omezen dobou běhu (limit je vidět v logu referenčního řešení) a dále je omezena i velikost dostupné paměti (ale tato úloha by ani s jedním omezením neměla mít problém). Testovací prostředí dále zakazuje používat některé "nebezpečné funkce" -- funkce pro spouštění programu, pro práci se sítí, ... Pokud jsou tyto funkce použité, program se nespustí. Možná ve svém programu používáte volání:

```
int main ( int argc, char * argv [] )
{
```

```
  ...

  system ( "pause" ); /* aby se nezavrelo okno programu */
  return 0;
}
```

Toto nebude v testovacím prostředí fungovat - je zakázáno spouštění jiného programu. (I pokud by se program spustil, byl by odmítnut. Nebyl by totiž nikdo, kdo by pauzu "odmáčkl", program by čekal věčně a překročil by tak maximální dobu běhu.) Pokud tedy chcete zachovat pauzu pro testování na Vašem počítači a zároveň chcete mít jistotu, že program poběží správně, použijte následující trik:

```
int main ( int argc, char * argv [] )
{

  ...

#ifndef __PROGTEST__
  system ( "pause" ); /* toto progtest "nevidi" */
#endif /* __PROGTEST__ */
  return 0;
}
```

- Slovní popis struktury platných vstupních dat není zcela exaktní. Proto připojujeme i formální popis vstupního jazyka v EBNF:

```
input      ::= { whiteSpace } decimal { whiteSpace } decimal { whiteSpace }
                decimal { whiteSpace } decimal { whiteSpace }
whiteSpace ::= ' ' | '\t' | '\n' | '\r'
decimal    ::= [ '+' ] integer [ '.' integer [ ( 'e' | 'E' ) [ '+' | '-' ] integer ] ]  |
               [ '+' | '-'  ] '.' integer [ ( 'e' | 'E' ) [ '+' | '-' ] integer ]
integer    ::= digit { digit }
digit      ::= '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```