Úkolem je vytvořit program, který vypočte délku časového intervalu.

Na vstupu dostane program čas počátku časového intervalu t1 a čas konce časového intervalu t2. Čas je vždy zadán ve formátu h:m:s,ms (hodina, minuta, sekunda, zlomek sekund). Program vypočte délku tohoto časového intervalu a zobrazí ji ve formátu h:m:s,ms. Očekávané chování je vidět i z ukázek níže.

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 hodnoty,
- hodnoty mimo meze (max. 23h, max. 59 min, max. 59 sec, max. 999 ms),
- chybějící čárka nebo dvojtečka,
- počáteční čas je větší než koncový čas.

Ukázky běhu programu:

Zadejte cas t1:
11:45:20,456
Zadejte cas t2:
12:07:54,349
Doba: 0:22:33,893

Zadejte cas t1:
15:18:34,232
Zadejte cas t2:
15:18:34,293
Doba: 0:00:00,061

Zadejte cas t1:
12:00:00,000
Zadejte cas t2:
12:00:00,000
Doba: 0:00:00,000

Zadejte cas t1:
01:01:01,001
Zadejte cas t2:
07:07:07,007
Doba: 6:06:06,006

Zadejte cas t1:
 1 : 1 : 1 , 001
Zadejte cas t2:
 7 : 7 : 7 , 007
Doba: 6:06:06,006

Zadejte cas t1:
1:1:1,001
Zadejte cas t2:
17:7:7,007
Doba: 16:06:06,006

Zadejte cas t1:
12:30:00,000
Zadejte cas t2:
12:00:00,000
Nespravny vstup.

Zadejte cas t1:
15:30:34,123
Zadejte cas t2:
15:60:34,123
Nespravny vstup.

Zadejte cas t1:
15:40:21.745
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 zobrazit text bez zvýrazňování (bez HTML markupu).
- Znak odřádkování je i za poslední řádkou výstupu (i za případným chybovým hlášením).
- Pro načítání vstupu se hodí funkce scanf. Pomocí funkce scanf lze i snadno kontrolovat přítomnost dvojtečky a čárky.
- Pro načítání vstupu není vhodné používat desetinná čísla (na vstupu je úmyslně desetinná čárka, ne tečka). Dále, vstup není vhodné zpracovávat jako řetězec, je to zbytečně pracné.
- 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 (**Courses → Video tutoriály**), jak testovací data použít a jak testování zautomatizovat.
- Na výstupu zobrazte hodiny na 2 místa bez úvodních nul, minuty a sekundy na 2 místa s úvodními nulami a milisekundy na 3 místa s úvodními nulami.
- Můžete se spolehnout, že na vstupu budou milisekundy zadané pomocí 3 číslic. Výjimkou je poslední (bonusový) test, kde počet milisekund může být zadán pouze pomocí 1, 2 nebo 3 cifer (např. zápis 1:2:3,4 znamená 400 ms). Takový vstup se zpracovává hůře, dá se ale vystačit s funkcí scanf.

- 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 ( void )
  {
      ...
      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 ( void )
  {
      ...
      #ifndef __PROGTEST__
      system ( "pause" ); /* toto Progtest nevidi */
      #endif /* __PROGTEST__ */
      return 0;
  }
  ```